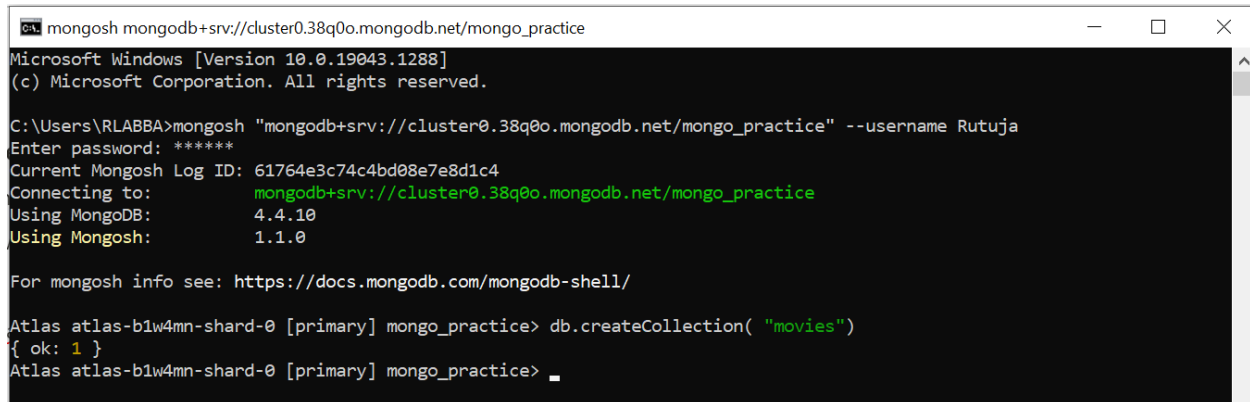


# Mongo Assignment 1

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.



```
mongosh mongodb+srv://cluster0.38q0o.mongodb.net/mongo_practice
Microsoft Windows [Version 10.0.19043.1288]
(c) Microsoft Corporation. All rights reserved.

C:\Users\RLABBA>mongosh "mongodb+srv://cluster0.38q0o.mongodb.net/mongo_practice" --username Rutuja
Enter password: *****
Current Mongosh Log ID: 61764e3c74c4bd08e7e8d1c4
Connecting to:      mongodb+srv://cluster0.38q0o.mongodb.net/mongo_practice
Using MongoDB:      4.4.10
Using Mongosh:      1.1.0

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

Atlas atlas-b1w4mn-shard-0 [primary] mongo_practice> db.createCollection( "movies")
{ ok: 1 }
Atlas atlas-b1w4mn-shard-0 [primary] mongo_practice> _
```

## Insert Documents

Insert the following documents into a movies collection.

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

```
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("617658783e158b93aae9621d"),
    '1': ObjectId("617658783e158b93aae9621e"),
    '2': ObjectId("617658783e158b93aae9621f"),
    '3': ObjectId("617658783e158b93aae96220"),
    '4': ObjectId("617658783e158b93aae96221"),
    '5': ObjectId("617658783e158b93aae96222"),
    '6': ObjectId("617658783e158b93aae96223"),
    '7': ObjectId("617658783e158b93aae96224")
  }
}
Atlas atlas-b1w4mn-shard-0 [primary] mongo_practice>
```

Reference [https://www.tutorialspoint.com/mongodb/mongodb\\_insert\\_document.htm](https://www.tutorialspoint.com/mongodb/mongodb_insert_document.htm)

## Query / Find Documents

query the movies collection to

1. get all documents

```
Atlas atlas-b1w4mn-shard-0 [primary] mongo_practice> db.movies.find()
[
  {
    _id: ObjectId("617658783e158b93aae9621d"),
    title: 'Fight Club',
    writer: 'Chuck Palahniuko',
    year: '1999',
    actors: [ 'Brad Pitt', 'Edward Norton' ]
  },
]
```

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

```
Atlas atlas-b1w4mn-shard-0 [primary] mongo_practice> db.movies.find({ writer: "Quentin Tarantino" })
[
  {
    _id: ObjectId("617658783e158b93aae9621e"),
    title: 'Pulp Fiction',
    writer: 'Quentin Tarantino',
    year: '1994',
    actors: [ 'John Travolta', 'Uma Thurman' ]
  },
  {
    _id: ObjectId("617658783e158b93aae9621f"),
    title: 'Inglorious Basterds',
    writer: 'Quentin Tarantino',
    year: '2009',
    actors: [ 'Brad Pitt', 'Diane Kruger', 'Eli Roth' ]
  }
]
```

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

```
Atlas atlas-b1w4mn-shard-0 [primary] mongo_practice> db.movies.find({ actors: "Brad Pitt" })
[
  {
    _id: ObjectId("617658783e158b93aae9621d"),
    title: 'Fight Club',
    writer: 'Chuck Palahniuko',
    year: '1999',
    actors: [ 'Brad Pitt', 'Edward Norton' ]
  },
  {
    _id: ObjectId("617658783e158b93aae9621f"),
    title: 'Inglorious Basterds',
    writer: 'Quentin Tarantino',
    year: '2009',
    actors: [ 'Brad Pitt', 'Diane Kruger', 'Eli Roth' ]
  }
]
```

4. get all documents with franchise set to "The Hobbit" [ Case sensitivity matters ]

```
Atlas atlas-b1w4mn-shard-0 [primary] mongo_practice> db.movies.find({ franchise: "The Hobbit" })
[
  {
    _id: ObjectId("617658783e158b93aae96220"),
    title: 'The Hobbit: An Unexpected Journey',
    writer: 'J.R.R. Tolkein',
    year: '2012',
    franchise: 'The Hobbit'
  },
  {
    _id: ObjectId("617658783e158b93aae96221"),
    title: 'The Hobbit: The Desolation of Smaug',
    writer: 'J.R.R. Tolkein',
    year: '2013',
    franchise: 'The Hobbit'
  },
  {
    _id: ObjectId("617658783e158b93aae96222"),
    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
  },
]
Atlas atlas-b1w4mn-shard-0 [primary] mongo_practice> _
```

5. get all movies released in the 90s

```
Atlas atlas-b1w4mn-shard-0 [primary] mongo_practice> db.movies.find(
... { $and: [ { year: { $gt: 1900 } }, { year: { $lt: 2000 } } ] } )
```

6. get all movies released before the year 2000 or after 2010

```
mongo_practice> db.movies.find( { $or: [{ year: { $lt: 2000 } }, { year: { $gt: 2010 } }] })
```

Reference: [https://www.tutorialspoint.com/mongodb/mongodb\\_query\\_document.htm](https://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-b1w4mn-shard-0 [primary] mongo_practice> db.movies.update({title: "The Hobbit: An Unexpected Journey"},{$set: {synopsis: "A reluctant hobbit, Bilbo Baggins, set 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
}
```

2. 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-b1w4mn-shard-0 [primary] mongo_practice> db.movies.updateOne({
... 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
}
```

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

```
Atlas atlas-b1w4mn-shard-0 [primary] mongo_practice> db.movies.updateOne( {title: "Pulp Fiction"},{ $push:{actors: "Samuel L. Jackson"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

Reference: [https://www.tutorialspoint.com/mongodb/mongodb\\_update\\_document.htm](https://www.tutorialspoint.com/mongodb/mongodb_update_document.htm)

## Text Search

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

```
Atlas atlas-b1w4mn-shard-0 [primary] mongo_practice> db.movies.createIndex({synopsis:"text"})
synopsis_text
Atlas atlas-b1w4mn-shard-0 [primary] mongo_practice> db.movies.find({$text:{$search:"Bilbo"}}).pretty()
[
```

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

```
Atlas atlas-b1w4mn-shard-0 [primary] mongo_practice> db.movies.find({$text:{$search:"Gandalf"}}).pretty()
[
  {
```

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

```
Atlas atlas-b1w4mn-shard-0 [primary] mongo_practice> db.movies.find({$text:{$search:"Bilbo -Gandalf"}}).pretty()
[
  {
    _id: ObjectId("617658783e158b93aae96222"),
```

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

```
Atlas atlas-b1w4mn-shard-0 [primary] mongo_practice> db.movies.find({$or: [{ synopsis: /dwarves/}, { synopsis: /hobbit/}]})
[
  {
    _id: ObjectId("617658783e158b93aae96220"),
    title: 'The Hobbit: An Unexpected Journey',
    writer: 'J.R.R. Tolkein',
    year: '2012',
    franchise: 'The Hobbit',
    synopsis: 'A reluctant hobbit, Bilbo Baggins, sets out to the Lonely Mountain with a spirited group of dwarves to reclaim the dragon Smaug.'
```

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

```
Atlas atlas-b1w4mn-shard-0 [primary] mongo_practice> db.movies.find({$and: [{ synopsis: /gold/}, { synopsis: /dragon/}]})
[
  {
    _id: ObjectId("617658783e158b93aae96220"),
    title: 'The Hobbit: An Unexpected Journey',
    writer: 'J.R.R. Tolkein',
    year: '2012',
    franchise: 'The Hobbit',
    synopsis: 'A reluctant hobbit, Bilbo Baggins, sets out to the Lonely Mountain with a spirited group of dwarves to reclaim the dragon Smaug.'
```

Reference: [https://www.tutorialspoint.com/mongodb/mongodb\\_text\\_search.htm](https://www.tutorialspoint.com/mongodb/mongodb_text_search.htm)

## Delete Documents

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

```
db.movies.deleteOne({title: "Pee Wee Herman's Big Adventure"})
```

```
Atlas atlas-b1w4mn-shard-0 [primary] mongo_practice> _  
{ acknowledged: true, deletedCount: 1 }
```

2. delete the movie "Avatar"

```
Atlas atlas-b1w4mn-shard-0 [primary] mongo_practice> db.movies.deleteOne({ title: "Avatar" })  
{ acknowledged: true, deletedCount: 1 }
```

Reference: [https://www.tutorialspoint.com/mongodb/mongodb\\_delete\\_document.htm](https://www.tutorialspoint.com/mongodb/mongodb_delete_document.htm)

## Relationships

1. 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-b1w4mn-shard-0 [primary] mongo_practice> db.users.insertMany([  
... username: "GoodGuyGreg",  
... first_name: "Good Guy",  
... last_name: "Greg"  
... },  
... {  
.... username: "ScumbagSteve",  
.... full_name: {  
..... first: "Scumbag",  
..... last: "Steve"  
..... }  
.... }  
... ])  
{  
  acknowledged: true,  
  insertedIds: {  
    '0': ObjectId("6176b8803e158b93aae96225"),  
    '1': ObjectId("6176b8803e158b93aae96226")  
  }  
}
```

2. 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-b1w4mn-shard-0 [primary] mongo_practice> db.posts.insertMany([{\n... username : "GoodGuyGreg",\n... title : "Passes out at party",\n... body : "Wakes up early and cleans house"\n... },\n... {\n..... username : "GoodGuyGreg",\n..... title : "Steals your identity",\n..... body : "Raises your credit score "\n..... },\n..... ])
```

### 3. Insert the following documents into a comments collection

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"

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"

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"

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"

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-b1w4mn-shard-0 [primary] mongo_practice> db.comments.insertMany([ { username: "GoodGuyGreg", comment: "Hope you got a good deal!", post: ObjectId("6176ba7b3e158b93aae9622b") }, { username: "GoodGuyGreg", comment: "What's mine is yours!", post: ObjectId("6176ba7b3e158b93aae9622c") }, { username: "GoodGuyGreg", comment: "Don't violate the licensing agreement!", post: ObjectId("6176ba7b3e158b93aae9622d") }, { username: "ScumbagSteve", comment: "It still isn't clean", post: ObjectId("6176ba7b3e158b93aae9622e") }, { username: "ScumbagSteve", comment: "Denied your PR cause I found a hack", post: ObjectId("6176ba7b3e158b93aae9622f") } ] )
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("61813da34a9c8dddeb3ddb0f4"),
    '1': ObjectId("61813da34a9c8dddeb3ddb0f5"),
    '2': ObjectId("61813da34a9c8dddeb3ddb0f6"),
    '3': ObjectId("61813da34a9c8dddeb3ddb0f7"),
    '4': ObjectId("61813da34a9c8dddeb3ddb0f8")
  }
}

```

#### 4. Querying related collections

##### a. find all users

```

Atlas atlas-b1w4mn-shard-0 [primary] mongo_practice> db.users.find()
[
  {
    _id: ObjectId("6176b8803e158b93aae96225"),
    username: 'GoodGuyGreg',
    first_name: 'Good Guy',
    last_name: 'Greg'
  },
  {
    _id: ObjectId("6176b8803e158b93aae96226"),
    username: 'ScumbagSteve',
    full_name: { first: 'Scumbag', last: 'Steve' }
  }
]

```

##### b. find all posts

```

Atlas atlas-b1w4mn-shard-0 [primary] mongo_practice> db.posts.find()
[
  {
    _id: ObjectId("6176ba7b3e158b93aae96227"),
    username: 'GoodGuyGreg',
    title: 'Passes out at party',
    body: 'Wakes up early and cleans house'
  },
  {
    _id: ObjectId("6176ba7b3e158b93aae96228"),
    username: 'GoodGuyGreg',
    title: 'Steals your identity',
    body: 'Raises your credit score '
  },
  {
    _id: ObjectId("6176ba7b3e158b93aae96229"),
    username: 'GoodGuyGreg',
    title: 'Reports a bug in your code',
    body: 'Sends you a Pull Request'
  },
  {
    _id: ObjectId("6176ba7b3e158b93aae9622b"),

```



- c. find all posts that was authored by "GoodGuyGreg"

```
Atlas atlas-b1w4mn-shard-0 [primary] mongo_practice> db.posts.find( {username: "GoodGuyGreg"} )
[
  {
    _id: ObjectId("6176ba7b3e158b93aae96227"),
    username: 'GoodGuyGreg',
    title: 'Passes out at party',
    body: 'Wakes up early and cleans house'
  },
  {
    _id: ObjectId("6176ba7b3e158b93aae96228"),
    username: 'GoodGuyGreg',
    title: 'Steals your identity',
    body: 'Raises your credit score '
  },
  {
    _id: ObjectId("6176ba7b3e158b93aae96229"),
    username: 'GoodGuyGreg',
    title: 'Reports a bug in your code',
    body: 'Sends you a Pull Request'
  }
]
```

- d. find all posts that was authored by "ScumbagSteve"

```
Atlas atlas-b1w4mn-shard-0 [primary] mongo_practice> db.posts.find( {username: "ScumbagSteve"} )
[
  {
    _id: ObjectId("6176ba7b3e158b93aae9622b"),
    username: 'ScumbagSteve',
    title: 'Borrows something',
    body: 'Sells it'
  },
  {
    _id: ObjectId("6176ba7b3e158b93aae9622c"),
    username: 'ScumbagSteve',
    title: 'Borrows everything',
    body: 'The end'
  },
  {
    _id: ObjectId("6176ba7b3e158b93aae9622d"),
    username: 'ScumbagSteve',
    title: 'Forks your repo on github',
    body: 'Sets to private'
  }
]
```

- e. find all comments

```

Atlas atlas-b1w4mn-shard-0 [primary] mongo_practice> db.comments.find()
[
  {
    _id: ObjectId("61813da34a9c0ddeb3ddb0f4"),
    username: 'GoodGuyGreg',
    comment: 'Hope you got a good deal!',
    post: ObjectId("6176ba7b3e158b93aae9622b")
  },
  {
    _id: ObjectId("61813da34a9c0ddeb3ddb0f5"),
    username: 'GoodGuyGreg',
    comment: "What's mine is yours!",
    post: ObjectId("6176ba7b3e158b93aae9622c")
  },
  {
    _id: ObjectId("61813da34a9c0ddeb3ddb0f6"),
    username: 'GoodGuyGreg',
    comment: "Don't violate the licensing agreement!",
    post: ObjectId("6176ba7b3e158b93aae9622d")
  },
]

```

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

```

Atlas atlas-b1w4mn-shard-0 [primary] mongo_practice> db.comments.find( {username: "GoodGuyGreg"} )
[
  {
    _id: ObjectId("61813da34a9c0ddeb3ddb0f4"),
    username: 'GoodGuyGreg',
    comment: 'Hope you got a good deal!',
    post: ObjectId("6176ba7b3e158b93aae9622b")
  },
  {
    _id: ObjectId("61813da34a9c0ddeb3ddb0f5"),
    username: 'GoodGuyGreg',
    comment: "What's mine is yours!",
    post: ObjectId("6176ba7b3e158b93aae9622c")
  },
  {
    _id: ObjectId("61813da34a9c0ddeb3ddb0f6"),
    username: 'GoodGuyGreg',
    comment: "Don't violate the licensing agreement!",
    post: ObjectId("6176ba7b3e158b93aae9622d")
  },
]

```

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

```

Atlas atlas-b1w4mn-shard-0 [primary] mongo_practice> db.comments.find( {username: "ScumbagSteve"} )
[
  {
    _id: ObjectId("61813da34a9c0ddeb3ddb0f7"),
    username: 'ScumbagSteve',
    comment: "It still isn't clean",
    post: ObjectId("6176ba7b3e158b93aae96227")
  },
  {
    _id: ObjectId("61813da34a9c0ddeb3ddb0f8"),
    username: 'ScumbagSteve',
    comment: 'Denied your PR cause I found a hack',
    post: ObjectId("6176ba7b3e158b93aae96229")
  },
]

```

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

```
Atlas atlas-b1w4mn-shard-0 [primary] mongo_practice> db.comments.find(
... { post: db.posts.findOne(
.... { title: "Reports a bug in your code" } )._id})
[
  {
    _id: ObjectId("61813da34a9c0ddeb3ddb0f8"),
    username: 'ScumbagSteve',
    comment: 'Denied your PR cause I found a hack',
    post: ObjectId("6176ba7b3e158b93aae96229")
  }
]
```

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