**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:\WINDOWS\system32>mongosh "mongodb+srv://mongodb-course.ailhn.mongodb.net/myFirstDatabase" --username diksharai

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

Current Mongosh Log ID: 61eea4887e6dce089c1d0600

Connecting to: mongodb+srv://mongodb-course.ailhn.mongodb.net/myFirstDatabase?appName=mongosh+1.1.9

Using MongoDB: 4.4.11

Using Mongosh: 1.1.9

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

Atlas atlas-68i9f5-shard-0 [primary] myFirstDatabase> show dbs

admin 336 kB

local 4.1 GB

Atlas atlas-68i9f5-shard-0 [primary] myFirstDatabase> use mongo\_practice

switched to db mongo\_practice

**Insert Documents**

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

Atlas atlas-68i9f5-shard-0 [primary] mongo\_practice> db.movies.insertMany([{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("61eeaa90180ec4a4ff6a70ee"),

'1': ObjectId("61eeaa90180ec4a4ff6a70ef"),

'2': ObjectId("61eeaa90180ec4a4ff6a70f0"),

'3': ObjectId("61eeaa90180ec4a4ff6a70f1"),

'4': ObjectId("61eeaa90180ec4a4ff6a70f2"),

'5': ObjectId("61eeaa90180ec4a4ff6a70f3"),

'6': ObjectId("61eeaa90180ec4a4ff6a70f4"),

'7': ObjectId("61eeaa90180ec4a4ff6a70f5")

}

}

**Query / Find Documents**

query the **movies** collection to:

1. get all documents

Atlas atlas-68i9f5-shard-0 [primary] mongo\_practice> db.movies.find()

[

{

\_id: ObjectId("61eeaa90180ec4a4ff6a70ee"),

title: 'Fight Club',

writer: 'Chuck Palahniuko',

year: 1999,

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

},

{

\_id: ObjectId("61eeaa90180ec4a4ff6a70ef"),

title: 'Pulp Fiction',

writer: 'Quentin Tarantino',

year: 1994,

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

},

{

\_id: ObjectId("61eeaa90180ec4a4ff6a70f0"),

title: 'Inglorious Basterds',

writer: 'Quentin Tarantino',

year: 2009,

actors: [ 'Brad Pitt', 'Diane Kruger', 'Eli Roth' ]

},

{

\_id: ObjectId("61eeaa90180ec4a4ff6a70f1"),

title: 'The Hobbit:An Unexpected Journey',

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

year: 2012,

franchise: 'The Hobbit'

},

{

\_id: ObjectId("61eeaa90180ec4a4ff6a70f2"),

title: 'The Hobbit: The Desolation of Smaug',

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

year: 2013,

franchise: 'The Hobbit'

},

{

\_id: ObjectId("61eeaa90180ec4a4ff6a70f3"),

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.'

},

{

\_id: ObjectId("61eeaa90180ec4a4ff6a70f4"),

title: "Pee Wee Herman's Big Adventure"

},

{ \_id: ObjectId("61eeaa90180ec4a4ff6a70f5"),

title: 'Avatar' }

]

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

Atlas atlas-68i9f5-shard-0 [primary] mongo\_practice> db.movies.find({writer:"Quentin Tarantino"})

[

{

\_id: ObjectId("61eeaa90180ec4a4ff6a70ef"),

title: 'Pulp Fiction',

writer: 'Quentin Tarantino',

year: 1994,

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

},

{

\_id: ObjectId("61eeaa90180ec4a4ff6a70f0"),

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-68i9f5-shard-0 [primary] mongo\_practice> db.movies.find({actors:"Brad Pitt"})

[

{

\_id: ObjectId("61eeaa90180ec4a4ff6a70ee"),

title: 'Fight Club',

writer: 'Chuck Palahniuko',

year: 1999,

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

},

{

\_id: ObjectId("61eeaa90180ec4a4ff6a70f0"),

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"

Atlas atlas-68i9f5-shard-0 [primary] mongo\_practice> db.movies.find({franchise:"The Hobbit"})

[

{

\_id: ObjectId("61eeaa90180ec4a4ff6a70f1"),

title: 'The Hobbit:An Unexpected Journey',

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

year: 2012,

franchise: 'The Hobbit'

},

{

\_id: ObjectId("61eeaa90180ec4a4ff6a70f2"),

title: 'The Hobbit: The Desolation of Smaug',

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

year: 2013,

franchise: 'The Hobbit'

},

{

\_id: ObjectId("61eeaa90180ec4a4ff6a70f3"),

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.'

}

]

5. get all movies released in the 90s

Atlas atlas-68i9f5-shard-0 [primary] mongo\_practice> db.movies.find({year:{$gte:1990,$lt:2000}})

[

{

\_id: ObjectId("61eeaa90180ec4a4ff6a70ee"),

title: 'Fight Club',

writer: 'Chuck Palahniuko',

year: 1999,

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

},

{

\_id: ObjectId("61eeaa90180ec4a4ff6a70ef"),

title: 'Pulp Fiction',

writer: 'Quentin Tarantino',

year: 1994,

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

}

]

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

Atlas atlas-68i9f5-shard-0 [primary] mongo\_practice> db.movies.find({$or:[{year:{$lt:2000}},{year:{$gt:2010}}]})

[

{

\_id: ObjectId("61eeaa90180ec4a4ff6a70ee"),

title: 'Fight Club',

writer: 'Chuck Palahniuko',

year: 1999,

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

},

{

\_id: ObjectId("61eeaa90180ec4a4ff6a70ef"),

title: 'Pulp Fiction',

writer: 'Quentin Tarantino',

year: 1994,

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

},

{

\_id: ObjectId("61eeaa90180ec4a4ff6a70f1"),

title: 'The Hobbit:An Unexpected Journey',

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

year: 2012,

franchise: 'The Hobbit'

},

{

\_id: ObjectId("61eeaa90180ec4a4ff6a70f2"),

title: 'The Hobbit: The Desolation of Smaug',

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

year: 2013,

franchise: 'The Hobbit'

},

{

\_id: ObjectId("61eeaa90180ec4a4ff6a70f3"),

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.'

}

]

**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-68i9f5-shard-0 [primary] mongo\_practice> db.movies.update({\_id:ObjectId("61eeaa90180ec4a4ff6a70f1")}, {$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."}})

{

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-68i9f5-shard-0 [primary] mongo\_practice> db.movies.update({\_id:ObjectId("61eeaa90180ec4a4ff6a70f2")}, {$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-68i9f5-shard-0 [primary] mongo\_practice> db.movies.update({\_id:ObjectId("61eeaa90180ec4a4ff6a70ef")}, {$push:{actors:"Samuel L. Jackson"}})

{

acknowledged: true,

insertedId: null,

matchedCount: 1,

modifiedCount: 1,

upsertedCount: 0

}

**Text Search**

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

Atlas atlas-68i9f5-shard-0 [primary] mongo\_practice> db.movies.find({synopsis:{$regex:"Bilbo"}})

[

{

\_id: ObjectId("61eeaa90180ec4a4ff6a70f1"),

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 their mountain home - and the gold within it - from the dragon Smaug.'

},

{

\_id: ObjectId("61eeaa90180ec4a4ff6a70f2"),

title: 'The Hobbit: The Desolation of Smaug',

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

year: 2013,

franchise: 'The Hobbit',

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.'

},

{

\_id: ObjectId("61eeaa90180ec4a4ff6a70f3"),

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.'

}

]

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

Atlas atlas-68i9f5-shard-0 [primary] mongo\_practice> db.movies.find({synopsis:{$regex:"Gandalf"}})

[

{

\_id: ObjectId("61eeaa90180ec4a4ff6a70f2"),

title: 'The Hobbit: The Desolation of Smaug',

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

year: 2013,

franchise: 'The Hobbit',

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.'

}

]

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

Atlas atlas-68i9f5-shard-0 [primary] mongo\_practice> db.movies.find({$and:[{synopsis:{$regex:"Bilbo"}}, {synopsis:{$not:/Gandalf/}}]})

[

{

\_id: ObjectId("61eeaa90180ec4a4ff6a70f1"),

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 their mountain home - and the gold within it - from the dragon Smaug.'

},

{

\_id: ObjectId("61eeaa90180ec4a4ff6a70f3"),

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.'

}

]

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

Atlas atlas-68i9f5-shard-0 [primary] mongo\_practice> db.movies.find({$or:[{synopsis:{$regex:"dwarves"}}, {synopsis:{$regex:"hobbit"}}]})

[

{

\_id: ObjectId("61eeaa90180ec4a4ff6a70f1"),

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 their mountain home - and the gold within it - from the dragon Smaug.'

},

{

\_id: ObjectId("61eeaa90180ec4a4ff6a70f2"),

title: 'The Hobbit: The Desolation of Smaug',

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

year: 2013,

franchise: 'The Hobbit',

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.'

}

]

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

Atlas atlas-68i9f5-shard-0 [primary] mongo\_practice> db.movies.find({$and:[{synopsis:{$regex:"gold"}}, {synopsis:{$regex:"dragon"}}]})

[

{

\_id: ObjectId("61eeaa90180ec4a4ff6a70f1"),

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 their mountain home - and the gold within it - from the dragon Smaug.'

}

]

**Delete Documents**

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

Atlas atlas-68i9f5-shard-0 [primary] mongo\_practice> db.movies.remove({\_id:ObjectId("61eeaa90180ec4a4ff6a70f4")})

{ acknowledged: true, deletedCount: 1 }

2. delete the movie "Avatar"

Atlas atlas-68i9f5-shard-0 [primary] mongo\_practice> db.movies.remove({\_id:ObjectId("61eeaa90180ec4a4ff6a70f5")})

{ acknowledged: true, deletedCount: 1 }

**Relationships**

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

username : GoodGuyGreg

first\_name : "Good Guy"

last\_name : "Greg"

Atlas atlas-68i9f5-shard-0 [primary] mongo\_practice> db.users.insert({username:"GoodGuyGreg", first\_name:"Good Guy", last\_name:"Greg"})

{

acknowledged: true,

insertedIds: { '0': ObjectId("61eeb573180ec4a4ff6a70f6") }

}

username : ScumbagSteve

full\_name :

first : "Scumbag"

last : "Steve"

Atlas atlas-68i9f5-shard-0 [primary] mongo\_practice> db.users.insert({username:"ScumbagSteve", fullname:{first: "Scumbag", last:"Steve"}})

{

acknowledged: true,

insertedIds: { '0': ObjectId("61eeb5c0180ec4a4ff6a70f7") }

}

Atlas atlas-68i9f5-shard-0 [primary] mongo\_practice> db.users.find()

[

{

\_id: ObjectId("61eeb573180ec4a4ff6a70f6"),

username: 'GoodGuyGreg',

first\_name: 'Good Guy',

last\_name: 'Greg'

},

{

\_id: ObjectId("61eeb5c0180ec4a4ff6a70f7"),

username: 'ScumbagSteve',

fullname: { first: 'Scumbag', last: 'Steve' }

}

]

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

username : GoodGuyGreg

title : Passes out at party

body : Wakes up early and cleans house

Atlas atlas-68i9f5-shard-0 [primary] mongo\_practice> db.posts.insert({username:"GoodGuyGreg", title:"Passes out at Party", body:"Raises your credit score"})

{

acknowledged: true,

insertedIds: { '0': ObjectId("61eeb666180ec4a4ff6a70f8") }

}

username : GoodGuyGreg

title : Steals your identity

body : Raises your credit score

Atlas atlas-68i9f5-shard-0 [primary] mongo\_practice> db.posts.insert({ username:"GoodGuyGreg", title:"Steals your identity", body:"Raises your credit score"})

{

acknowledged: true,

insertedIds: { '0': ObjectId("61eeb674180ec4a4ff6a70f9") }

}

username : GoodGuyGreg

title : Reports a bug in your code

body : Sends you a Pull Request

Atlas atlas-68i9f5-shard-0 [primary] mongo\_practice> db.posts.insert({username:"GoodGuyGreg", title:"Reports a bug in your code", body:"Sends you a pull request"})

{

acknowledged: true,

insertedIds: { '0': ObjectId("61eeb681180ec4a4ff6a70fa") }

}

username : ScumbagSteve

title : Borrows something

body : Sells it

Atlas atlas-68i9f5-shard-0 [primary] mongo\_practice> db.posts.insert({ username:"ScumbagSteve", title:"Borrows something", body:"Sells it"})

{

acknowledged: true,

insertedIds: { '0': ObjectId("61eeb692180ec4a4ff6a70fb") }

}

username : ScumbagSteve

title : Borrows everything

body : The end

Atlas atlas-68i9f5-shard-0 [primary] mongo\_practice> db.posts.insert({ username:"ScumbagSteve", title:"Borrows everything", body:"The end"})

{

acknowledged: true,

insertedIds: { '0': ObjectId("61eeb6a1180ec4a4ff6a70fc") }

}

username : ScumbagSteve

title : Forks your repo on github

body : Sets to private

Atlas atlas-68i9f5-shard-0 [primary] mongo\_practice> db.posts.insert({username:"ScumbagSteve", title:"Forks your repo on github", body:"Sets to private"})

{

acknowledged: true,

insertedIds: { '0': ObjectId("61eeb6b3180ec4a4ff6a70fd") }

}

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"

Atlas atlas-68i9f5-shard-0 [primary] mongo\_practice> db.comments.insert({ username:"GoodGuyGreg", comment:"Hope you got a good deal!", post:ObjectId("61eeb692180ec4a4ff6a70fb")})

{

acknowledged: true,

insertedIds: { '0': ObjectId("61eeb79b180ec4a4ff6a70fe") }

}

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-68i9f5-shard-0 [primary] mongo\_practice> db.comments.insert({username:"GoodGuyGreg", comment:"What's mine is yours!", post:ObjectId("61eeb6a1180ec4a4ff6a70fc")})

{

acknowledged: true,

insertedIds: { '0': ObjectId("61eeb832180ec4a4ff6a70ff") }

}

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-68i9f5-shard-0 [primary] mongo\_practice> db.comments.insert({username:"GoodGuyGreg", comment:"Don't violate the licensing agreement!", post:ObjectId("61eeb6b3180ec4a4ff6a70fd")})

{

acknowledged: true,

insertedIds: { '0': ObjectId("61eeb883180ec4a4ff6a7100") }

}

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-68i9f5-shard-0 [primary] mongo\_practice> db.comments.insert({username:"ScumbagSteve", comment:"It still isn't clean", post:ObjectId("61eeb666180ec4a4ff6a70f8")})

{

acknowledged: true,

insertedIds: { '0': ObjectId("61eeb8cc180ec4a4ff6a7101") }

}

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-68i9f5-shard-0 [primary] mongo\_practice> db.comments.insert({username:"ScumbagSteve", comment:"Denied your PR cause I found a hack", post:ObjectId("61eeb681180ec4a4ff6a70fa")})

{

acknowledged: true,

insertedIds: { '0': ObjectId("61eeb931180ec4a4ff6a7102") }

}

**Querying related collections**

1. find all users

Atlas atlas-68i9f5-shard-0 [primary] mongo\_practice> db.users.find()

[

{

\_id: ObjectId("61eeb573180ec4a4ff6a70f6"),

username: 'GoodGuyGreg',

first\_name: 'Good Guy',

last\_name: 'Greg'

},

{

\_id: ObjectId("61eeb5c0180ec4a4ff6a70f7"),

username: 'ScumbagSteve',

fullname: { first: 'Scumbag', last: 'Steve' }

}

]

2. find all posts

Atlas atlas-68i9f5-shard-0 [primary] mongo\_practice> db.posts.find()

[

{

\_id: ObjectId("61eeb666180ec4a4ff6a70f8"),

username: 'GoodGuyGreg',

title: 'Passes out at Party',

body: 'Raises your credit score'

},

{

\_id: ObjectId("61eeb674180ec4a4ff6a70f9"),

username: 'GoodGuyGreg',

title: 'Steals your identity',

body: 'Raises your credit score'

},

{

\_id: ObjectId("61eeb681180ec4a4ff6a70fa"),

username: 'GoodGuyGreg',

title: 'Reports a bug in your code',

body: 'Sends you a pull request'

},

{

\_id: ObjectId("61eeb692180ec4a4ff6a70fb"),

username: 'ScumbagSteve',

title: 'Borrows something',

body: 'Sells it'

},

{

\_id: ObjectId("61eeb6a1180ec4a4ff6a70fc"),

username: 'ScumbagSteve',

title: 'Borrows everything',

body: 'The end'

},

{

\_id: ObjectId("61eeb6b3180ec4a4ff6a70fd"),

username: 'ScumbagSteve',

title: 'Forks your repo on github',

body: 'Sets to private'

}

]

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

Atlas atlas-68i9f5-shard-0 [primary] mongo\_practice> db.posts.find({username:"GoodGuyGreg"})

[

{

\_id: ObjectId("61eeb666180ec4a4ff6a70f8"),

username: 'GoodGuyGreg',

title: 'Passes out at Party',

body: 'Raises your credit score'

},

{

\_id: ObjectId("61eeb674180ec4a4ff6a70f9"),

username: 'GoodGuyGreg',

title: 'Steals your identity',

body: 'Raises your credit score'

},

{

\_id: ObjectId("61eeb681180ec4a4ff6a70fa"),

username: 'GoodGuyGreg',

title: 'Reports a bug in your code',

body: 'Sends you a pull request'

}

]

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

Atlas atlas-68i9f5-shard-0 [primary] mongo\_practice> db.posts.find({username:"ScumbagSteve"})

[

{

\_id: ObjectId("61eeb692180ec4a4ff6a70fb"),

username: 'ScumbagSteve',

title: 'Borrows something',

body: 'Sells it'

},

{

\_id: ObjectId("61eeb6a1180ec4a4ff6a70fc"),

username: 'ScumbagSteve',

title: 'Borrows everything',

body: 'The end'

},

{

\_id: ObjectId("61eeb6b3180ec4a4ff6a70fd"),

username: 'ScumbagSteve',

title: 'Forks your repo on github',

body: 'Sets to private'

}

]

5. find all comments

Atlas atlas-68i9f5-shard-0 [primary] mongo\_practice> db.comments.find()

[

{

\_id: ObjectId("61eeb79b180ec4a4ff6a70fe"),

username: 'GoodGuyGreg',

comment: 'Hope you got a good deal!',

post: ObjectId("61eeb692180ec4a4ff6a70fb")

},

{

\_id: ObjectId("61eeb832180ec4a4ff6a70ff"),

username: 'GoodGuyGreg',

comment: "What's mine is yours!",

post: ObjectId("61eeb6a1180ec4a4ff6a70fc")

},

{

\_id: ObjectId("61eeb883180ec4a4ff6a7100"),

username: 'GoodGuyGreg',

comment: "Don't violate the licensing agreement!",

post: ObjectId("61eeb6b3180ec4a4ff6a70fd")

},

{

\_id: ObjectId("61eeb8cc180ec4a4ff6a7101"),

username: 'ScumbagSteve',

comment: "It still isn't clean",

post: ObjectId("61eeb666180ec4a4ff6a70f8")

},

{

\_id: ObjectId("61eeb931180ec4a4ff6a7102"),

username: 'ScumbagSteve',

comment: 'Denied your PR cause I found a hack',

post: ObjectId("61eeb681180ec4a4ff6a70fa")

}

]

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

Atlas atlas-68i9f5-shard-0 [primary] mongo\_practice> db.comments.find({username:"GoodGuyGreg"})

[

{

\_id: ObjectId("61eeb79b180ec4a4ff6a70fe"),

username: 'GoodGuyGreg',

comment: 'Hope you got a good deal!',

post: ObjectId("61eeb692180ec4a4ff6a70fb")

},

{

\_id: ObjectId("61eeb832180ec4a4ff6a70ff"),

username: 'GoodGuyGreg',

comment: "What's mine is yours!",

post: ObjectId("61eeb6a1180ec4a4ff6a70fc")

},

{

\_id: ObjectId("61eeb883180ec4a4ff6a7100"),

username: 'GoodGuyGreg',

comment: "Don't violate the licensing agreement!",

post: ObjectId("61eeb6b3180ec4a4ff6a70fd")

}

]

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

Atlas atlas-68i9f5-shard-0 [primary] mongo\_practice> db.comments.find({username:"ScumbagSteve"})

[

{

\_id: ObjectId("61eeb8cc180ec4a4ff6a7101"),

username: 'ScumbagSteve',

comment: "It still isn't clean",

post: ObjectId("61eeb666180ec4a4ff6a70f8")

},

{

\_id: ObjectId("61eeb931180ec4a4ff6a7102"),

username: 'ScumbagSteve',

comment: 'Denied your PR cause I found a hack',

post: ObjectId("61eeb681180ec4a4ff6a70fa")

}

]

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

Atlas atlas-68i9f5-shard-0 [primary] mongo\_practice> db.comments.find({post:ObjectId("61eeb681180ec4a4ff6a70fa")})

[

{

\_id: ObjectId("61eeb931180ec4a4ff6a7102"),

username: 'ScumbagSteve',

comment: 'Denied your PR cause I found a hack',

post: ObjectId("61eeb681180ec4a4ff6a70fa")

}

]