**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\ROHITK>mongosh "mongodb+srv://cluster0.iz5wh.mongodb.net/cluster0" --username Rohit

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

Current Mongosh Log ID: 61dbc91ee76899d0a8812dc1

Connecting to: mongodb+srv://cluster0.iz5wh.mongodb.net/cluster0

Using MongoDB: 4.4.10.

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

**Insert Documents**

Insert the following documents into a movies collection.

Atlas atlas-2v2ryx-shard-0 [primary] mongodb\_practice> db.createCollection('movies')

{ ok: 1 }

Atlas atlas-2v2ryx-shard-0 [primary] mongodb\_practice> show collections

movies

The documents that are to be inserted;

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

//Code

Atlas atlas-2v2ryx-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("61da79d7c4f4f44d993c57d6"),

'1': ObjectId("61da79d7c4f4f44d993c57d7"),

'2': ObjectId("61da79d7c4f4f44d993c57d8"),

'3': ObjectId("61da79d7c4f4f44d993c57d9"),

'4': ObjectId("61da79d7c4f4f44d993c57da"),

'5': ObjectId("61da79d7c4f4f44d993c57db"),

'6': ObjectId("61da79d7c4f4f44d993c57dc"),

'7': ObjectId("61da79d7c4f4f44d993c57dd")

}

}

**Query/ Find Documents**

query the movies collection to

1. get all documents

Atlas atlas-2v2ryx-shard-0 [primary] mongodb\_practice> db.movies.find()

[

{

\_id: ObjectId("61da79d7c4f4f44d993c57d6"),

title: 'Fight Club',

writer: 'Chuck Palahniuko',

year: 1999,

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

},

{

\_id: ObjectId("61da79d7c4f4f44d993c57d7"),

title: 'Pulp Fiction',

writer: 'Quentin Tarantino',

year: 1994,

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

},

{

\_id: ObjectId("61da79d7c4f4f44d993c57d8"),

title: 'Inglorious Basterds',

writer: 'Quentin Tarantino',

year: 2009,

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

},

{

\_id: ObjectId("61da79d7c4f4f44d993c57d9"),

title: 'The Hobbit: An Unexpected Journey',

writer: 'J.R.R Tolkein',

year: 2012,

franchise: 'The Hobbit'

},

{

\_id: ObjectId("61da79d7c4f4f44d993c57da"),

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

writer: 'J.R.R Tolkein',

year: 2013,

franchise: 'The Hobbit'

},

{

\_id: ObjectId("61da79d7c4f4f44d993c57db"),

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'

},

{

\_id: ObjectId("61da79d7c4f4f44d993c57dc"),

title: "Pee Wee Herman's Big Adventure"

},

{ \_id: ObjectId("61da79d7c4f4f44d993c57dd"), title: 'Avatar' }

]

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

Atlas atlas-2v2ryx-shard-0 [primary] mongodb\_practice> db.movies.find({writer:"Quentin Tarantino"})

[

{

\_id: ObjectId("61da79d7c4f4f44d993c57d7"),

title: 'Pulp Fiction',

writer: 'Quentin Tarantino',

year: 1994,

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

},

{

\_id: ObjectId("61da79d7c4f4f44d993c57d8"),

title: 'Inglorious Basterds',

writer: 'Quentin Tarantino',

year: 2009,

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

}

]

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

Atlas atlas-2v2ryx-shard-0 [primary] mongodb\_practice> db.movies.find({actors:"Brad Pitt"})

[

{

\_id: ObjectId("61da79d7c4f4f44d993c57d6"),

title: 'Fight Club',

writer: 'Chuck Palahniuko',

year: 1999,

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

},

{

\_id: ObjectId("61da79d7c4f4f44d993c57d8"),

title: 'Inglorious Basterds',

writer: 'Quentin Tarantino',

year: 2009,

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

}

]

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

Atlas atlas-2v2ryx-shard-0 [primary] mongodb\_practice> db.movies.find({franchise:"The Hobbit"})

[

{

\_id: ObjectId("61da79d7c4f4f44d993c57d9"),

title: 'The Hobbit: An Unexpected Journey',

writer: 'J.R.R Tolkein',

year: 2012,

franchise: 'The Hobbit'

},

{

\_id: ObjectId("61da79d7c4f4f44d993c57da"),

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

writer: 'J.R.R Tolkein',

year: 2013,

franchise: 'The Hobbit'

},

{

\_id: ObjectId("61da79d7c4f4f44d993c57db"),

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'

}

]

1. get all movies released in the 90s

Atlas atlas-2v2ryx-shard-0 [primary] mongodb\_practice> db.movies.find({year:{$gt:1990, $lt:2000}})

[

{

\_id: ObjectId("61da79d7c4f4f44d993c57d6"),

title: 'Fight Club',

writer: 'Chuck Palahniuko',

year: 1999,

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

},

{

\_id: ObjectId("61da79d7c4f4f44d993c57d7"),

title: 'Pulp Fiction',

writer: 'Quentin Tarantino',

year: 1994,

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

}

]

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

Atlas atlas-2v2ryx-shard-0 [primary] mongodb\_practice> db.movies.find({$or:[{year:{$gt:2010}},{year: {$lt:2000}}]})

[

{

\_id: ObjectId("61da79d7c4f4f44d993c57d6"),

title: 'Fight Club',

writer: 'Chuck Palahniuko',

year: 1999,

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

},

{

\_id: ObjectId("61da79d7c4f4f44d993c57d7"),

title: 'Pulp Fiction',

writer: 'Quentin Tarantino',

year: 1994,

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

},

{

\_id: ObjectId("61da79d7c4f4f44d993c57d9"),

title: 'The Hobbit: An Unexpected Journey',

writer: 'J.R.R Tolkein',

year: 2012,

franchise: 'The Hobbit'

},

{

\_id: ObjectId("61da79d7c4f4f44d993c57da"),

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

writer: 'J.R.R Tolkein',

year: 2013,

franchise: 'The Hobbit'

},

{

\_id: ObjectId("61da79d7c4f4f44d993c57db"),

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'

}

]

**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-2v2ryx-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-2v2ryx-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-2v2ryx-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

}

**Text Search**

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

Atlas atlas-2v2ryx-shard-0 [primary] mongodb\_practice> db.movies.find({synopsis:{$regex:"Bilbo"}})

[

{

\_id: ObjectId("61da79d7c4f4f44d993c57d9"),

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("61da79d7c4f4f44d993c57da"),

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("61da79d7c4f4f44d993c57db"),

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'

}

]

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

Atlas atlas-2v2ryx-shard-0 [primary] mongodb\_practice> db.movies.find({synopsis:{$regex:"Gandalf"}})

[

{

\_id: ObjectId("61da79d7c4f4f44d993c57da"),

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.'

}

]

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

Atlas atlas-2v2ryx-shard-0 [primary] mongodb\_practice> db.movies.find({$and:[{synopsis:{$regex:"Bilbo"}}, {synopsis:{$not:/Gandalf/}}]})

[

{

\_id: ObjectId("61da79d7c4f4f44d993c57d9"),

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("61da79d7c4f4f44d993c57db"),

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'

}

]

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

Atlas atlas-2v2ryx-shard-0 [primary] mongodb\_practice> db.movies.find({$or:[{synopsis:{$regex:"dwarves"}}, {synopsis:{$r$regex:"hobbit"}}]})

[

{

\_id: ObjectId("61da79d7c4f4f44d993c57d9"),

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("61da79d7c4f4f44d993c57da"),

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.'

}

]

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

Atlas atlas-2v2ryx-shard-0 [primary] mongodb\_practice> db.movies.find({$and:[{synopsis:{$regex:"gold"}}, {synopsis:{$reg$regex:"dragon"}}]})

[

{

\_id: ObjectId("61da79d7c4f4f44d993c57d9"),

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-2v2ryx-shard-0 [primary] mongodb\_practice> db.movies.remove({title:"Pee Wee Herman's Big Adventure"})

DeprecationWarning: Collection.remove() is deprecated. Use deleteOne, deleteMany, findOneAndDelete, or bulkWrite.

{ acknowledged: true, deletedCount: 1 }

1. delete the movie "Avatar"

Atlas atlas-2v2ryx-shard-0 [primary] mongodb\_practice> db.movies.remove({title:"Avatar"})

{ acknowledged: true, deletedCount: 1 }

**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"

//creating collection users

Atlas atlas-2v2ryx-shard-0 [primary] mongodb\_practice> db.createCollection('users')

{ ok: 1 }

//insertion

Atlas atlas-2v2ryx-shard-0 [primary] mongodb\_practice> db.users.insertMany([

... {username: 'GoodGuyGreg',first\_name: 'GoodGuy', last\_name: 'Greg'},

... {username: 'ScumbagSteve', fullname:{first:'Scumbag', last:'Steve'}}])

{

acknowledged: true,

insertedIds: {

'0': ObjectId("61da85b9f1e258629fdd6f5a"),

'1': ObjectId("61da85b9f1e258629fdd6f5b")

}

}

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

//create posts collection

Atlas atlas-2v2ryx-shard-0 [primary] mongodb\_practice> db.createCollection('posts')

{ ok: 1 }

//insertion

Atlas atlas-2v2ryx-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("61da896cf1e258629fdd6f5c"),

'1': ObjectId("61da896cf1e258629fdd6f5d"),

'2': ObjectId("61da896cf1e258629fdd6f5e"),

'3': ObjectId("61da896cf1e258629fdd6f5f"),

'4': ObjectId("61da896cf1e258629fdd6f60"),

'5': ObjectId("61da896cf1e258629fdd6f61")

}

}

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"

//create comments collection

Atlas atlas-2v2ryx-shard-0 [primary] mongodb\_practice> db.createCollection('comments')

{ ok: 1 }

//insertion with the post\_id

Atlas atlas-2v2ryx-shard-0 [primary] mongodb\_practice> db.comments.insert(

... {username:'GoodGuyGreg', comment: 'Hope you got a good deal!', post:ObjectId("61da896cf1e258629fdd6f5f")})

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

{

acknowledged: true,

insertedIds: { '0': ObjectId("61da8b2ef1e258629fdd6f62") }

}

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-2v2ryx-shard-0 [primary] mongodb\_practice> db.comments.insert(

... {username: 'GoodGuyGreg', comment: "What's mine is yours!", post:ObjectId("61da896cf1e258629fdd6f60")})

{

acknowledged: true,

insertedIds: { '0': ObjectId("61da8cd5f1e258629fdd6f64") }

}

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-2v2ryx-shard-0 [primary] mongodb\_practice> db.comments.insert(

... {username: 'GoodGuyGreg', comment: "Don't violate the licensing agreement!", post:ObjectId("61da896cf1e258629fdd6f61")})

{

acknowledged: true,

insertedIds: { '0': ObjectId("61da8d6ef1e258629fdd6f65") }

}

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-2v2ryx-shard-0 [primary] mongodb\_practice> db.comments.insert(

... {username: 'ScumbagSteve', comment: "It still isn't clean", post:ObjectId("61da896cf1e258629fdd6f5c")})

{

acknowledged: true,

insertedIds: { '0': ObjectId("61da8dfcf1e258629fdd6f66") }

}

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-2v2ryx-shard-0 [primary] mongodb\_practice> db.comments.insert(

... {username: 'ScumbagSteve', comment: "Denied your PR cause I found a hack",

..... post:ObjectId("61da896cf1e258629fdd6f5e")})

{

acknowledged: true,

insertedIds: { '0': ObjectId("61da8e8bf1e258629fdd6f67") }

}

**Querying related Collections**

1. find all users

Atlas atlas-2v2ryx-shard-0 [primary] mongodb\_practice> db.users.find().pretty()

[

{

\_id: ObjectId("61da85b9f1e258629fdd6f5a"),

username: 'GoodGuyGreg',

first\_name: 'GoodGuy',

last\_name: 'Greg'

},

{

\_id: ObjectId("61da85b9f1e258629fdd6f5b"),

username: 'ScumbagSteve',

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

}

]

1. find all posts

Atlas atlas-2v2ryx-shard-0 [primary] mongodb\_practice> db.posts.find().pretty()

[

{

\_id: ObjectId("61da896cf1e258629fdd6f5c"),

username: 'GoodGuyGreg',

title: 'Passes out at party',

body: 'Wakes up early and cleans house'

},

{

\_id: ObjectId("61da896cf1e258629fdd6f5d"),

username: 'GoodGuyGreg',

title: 'Steals your identity',

body: 'Raises your credit score'

},

{

\_id: ObjectId("61da896cf1e258629fdd6f5e"),

username: 'GoodGuyGreg',

title: 'Reports a bug in your code',

body: 'Sends you a pull request'

},

{

\_id: ObjectId("61da896cf1e258629fdd6f5f"),

username: 'ScumbagSteve',

title: 'Borrows something',

body: 'Sells it'

},

{

\_id: ObjectId("61da896cf1e258629fdd6f60"),

username: 'ScumbagSteve',

title: 'Borrows everything',

body: 'The end'

},

{

\_id: ObjectId("61da896cf1e258629fdd6f61"),

username: 'ScumbagSteve',

title: 'Forks your repo in github',

body: 'Sets to private'

}

]

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

Atlas atlas-2v2ryx-shard-0 [primary] mongodb\_practice> db.posts.find({username:"GoodGuyGreg"})

[

{

\_id: ObjectId("61da896cf1e258629fdd6f5c"),

username: 'GoodGuyGreg',

title: 'Passes out at party',

body: 'Wakes up early and cleans house'

},

{

\_id: ObjectId("61da896cf1e258629fdd6f5d"),

username: 'GoodGuyGreg',

title: 'Steals your identity',

body: 'Raises your credit score'

},

{

\_id: ObjectId("61da896cf1e258629fdd6f5e"),

username: 'GoodGuyGreg',

title: 'Reports a bug in your code',

body: 'Sends you a pull request'

}

]

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

Atlas atlas-2v2ryx-shard-0 [primary] mongodb\_practice> db.posts.find({username:"ScumbagSteve"})

[

{

\_id: ObjectId("61da896cf1e258629fdd6f5f"),

username: 'ScumbagSteve',

title: 'Borrows something',

body: 'Sells it'

},

{

\_id: ObjectId("61da896cf1e258629fdd6f60"),

username: 'ScumbagSteve',

title: 'Borrows everything',

body: 'The end'

},

{

\_id: ObjectId("61da896cf1e258629fdd6f61"),

username: 'ScumbagSteve',

title: 'Forks your repo in github',

body: 'Sets to private'

}

]

1. find all comments

Atlas atlas-2v2ryx-shard-0 [primary] mongodb\_practice> db.comments.find().pretty()

[

{

\_id: ObjectId("61da8b2ef1e258629fdd6f62"),

username: 'GoodGuyGreg',

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

post: ObjectId("61da896cf1e258629fdd6f5f")

},

{

\_id: ObjectId("61da8cd5f1e258629fdd6f64"),

username: 'GoodGuyGreg',

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

post: ObjectId("61da896cf1e258629fdd6f60")

},

{

\_id: ObjectId("61da8d6ef1e258629fdd6f65"),

username: 'GoodGuyGreg',

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

post: ObjectId("61da896cf1e258629fdd6f61")

},

{

\_id: ObjectId("61da8dfcf1e258629fdd6f66"),

username: 'ScumbagSteve',

comment: "It still isn't clean",

post: ObjectId("61da896cf1e258629fdd6f5c")

},

{

\_id: ObjectId("61da8e8bf1e258629fdd6f67"),

username: 'ScumbagSteve',

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

post: ObjectId("61da896cf1e258629fdd6f5e")

}

]

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

Atlas atlas-2v2ryx-shard-0 [primary] mongodb\_practice> db.comments.find({username:"GoodGuyGreg"})

[

{

\_id: ObjectId("61da8b2ef1e258629fdd6f62"),

username: 'GoodGuyGreg',

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

post: ObjectId("61da896cf1e258629fdd6f5f")

},

{

\_id: ObjectId("61da8cd5f1e258629fdd6f64"),

username: 'GoodGuyGreg',

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

post: ObjectId("61da896cf1e258629fdd6f60")

},

{

\_id: ObjectId("61da8d6ef1e258629fdd6f65"),

username: 'GoodGuyGreg',

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

post: ObjectId("61da896cf1e258629fdd6f61")

}

]

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

Atlas atlas-2v2ryx-shard-0 [primary] mongodb\_practice> db.comments.find({username:'ScumbagSteve'})

[

{

\_id: ObjectId("61da8dfcf1e258629fdd6f66"),

username: 'ScumbagSteve',

comment: "It still isn't clean",

post: ObjectId("61da896cf1e258629fdd6f5c")

},

{

\_id: ObjectId("61da8e8bf1e258629fdd6f67"),

username: 'ScumbagSteve',

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

post: ObjectId("61da896cf1e258629fdd6f5e")

}

]

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

Atlas atlas-2v2ryx-shard-0 [primary] mongodb\_practice> db.comments.find({post:ObjectId("61da896cf1e258629fdd6f5e")})

[

{

\_id: ObjectId("61da8e8bf1e258629fdd6f67"),

username: 'ScumbagSteve',

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

post: ObjectId("61da896cf1e258629fdd6f5e")

}

]

## \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*