**MongoDB Lab Assignments Day-1**

**MongoDB Exercise in mongo shell**

Connect to a running mongo instance, use a database named mongo\_practice.

use mongo\_practice

**Insert Documents**

Insert the following documents into a movies collection.

* title : Fight Club

writer : Chuck Palahniuk

year : 1999

actors : [

Brad Pitt

Edward Norton

]

MongoDB Enterprise atlas-125c6m-shard-0:PRIMARY> db.movies.insert({title:"Fight Club", writer: "Chuck Palahniuk", year: "1999", actors: ["Brad Pitt", "Edward Norton"]})

WriteResult({ "nInserted" : 1 })

* title : Pulp Fiction

writer : Quentin Tarantino

year : 1994

actors : [

John Travolta

Uma Thurman

]

MongoDB Enterprise atlas-125c6m-shard-0:PRIMARY> db.movies.insert({title:"Pulp Fiction", writer:"Quentin Tarantino", year:"1994", actors:["John Travolta", "Uma Thurman"]})

WriteResult({ "nInserted" : 1 })

* title : Inglorious Basterds

writer : Quentin Tarantino

year : 2009

actors : [

Brad Pitt

Diane Kruger

Eli Roth

]

MongoDB Enterprise atlas-125c6m-shard-0:PRIMARY> db.movies.insert({title:"Inglorious Basterds", writer:"Quentin Tarantino", year:"2009", actors:["Brad Pitt", "Diane Kruger", "Eli Roth"]})

WriteResult({ "nInserted" : 1 })

* title : The Hobbit: An Unexpected Journey

writer : J.R.R. Tolkein

year : 2012

franchise : The Hobbit

MongoDB Enterprise atlas-125c6m-shard-0:PRIMARY> db.movies.insert({title:"The Hobbit: An unexpected Journey", writer:"J.R.R. Tolkein", year:"2012",franchise:"The Hobbit"})

WriteResult({ "nInserted" : 1 })

* title : The Hobbit: The Desolation of Smaug

writer : J.R.R. Tolkein

year : 2013

franchise : The Hobbit

MongoDB Enterprise atlas-125c6m-shard-0:PRIMARY> db.movies.insert({title:"The Hobbit: The Desolation of Smaug", writer:"J.R.R Tolkien", year:"2013", franchise:"The Hobbit"})

WriteResult({ "nInserted" : 1 })

* 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.

MongoDB Enterprise atlas-125c6m-shard-0:PRIMARY> db.movies.insert({title:"The Hobbit: The Battle of the Five Armies", writer:"J.R.R Tolkien", 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."})

WriteResult({ "nInserted" : 1 })

* title : Pee Wee Herman's Big Adventure

MongoDB Enterprise atlas-125c6m-shard-0:PRIMARY> db.movies.insert({title:"Pee Wee Herman's Big Adventures"})

WriteResult({ "nInserted" : 1 })

* title : Avatar

MongoDB Enterprise atlas-125c6m-shard-0:PRIMARY> db.movies.insert({title: "Avatar"})

WriteResult({ "nInserted" : 1 })

**Query / Find Documents**

query the movies collection to

1. get all documents

MongoDB Enterprise atlas-125c6m-shard-0:PRIMARY> db.movies.find()

{ "\_id" : ObjectId("60e6db6028280e9b364d23f3"), "title" : "Fight Club", "writer" : "Chuck Palahniuk", "year" : "1999", "actors" : [ "Brad Pitt", "Edward Norton" ] }

{ "\_id" : ObjectId("60e6dbd028280e9b364d23f4"), "title" : "Pulp Fiction", "writer" : "Quentin Tarantino", "year" : "1994", "actors" : [ "John Travolta", "Uma Thurman" ] }

{ "\_id" : ObjectId("60e6dbed28280e9b364d23f5"), "title" : "Inglorious Basterds", "writer" : "Quentin Tarantino", "year" : "2009", "actors" : [ "Brad Pitt", "Diane Kruger", "Eli Roth" ] }

{ "\_id" : ObjectId("60e6dc1728280e9b364d23f6"), "title" : "The Hobbit: An unexpected Journey", "writer" : "J.R.R. Tolkein", "year" : "2012", "franchise" : "The Hobbit" }

{ "\_id" : ObjectId("60e6dc3528280e9b364d23f7"), "title" : "The Hobbit: The Desolation of Smaug", "writer" : "J.R.R Tolkien", "year" : "2013", "franchise" : "The Hobbit" }

{ "\_id" : ObjectId("60e6dc5e28280e9b364d23f8"), "title" : "The Hobbit: The Battle of the Five Armies", "writer" : "J.R.R Tolkien", "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("60e6dc7b28280e9b364d23f9"), "title" : "Pee Wee Herman's Big Adventures" }

{ "\_id" : ObjectId("60e6dcb428280e9b364d23fa"), "title" : "Avatar" }

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

MongoDB Enterprise atlas-125c6m-shard-0:PRIMARY> db.movies.find({writer:"Quentin Tarantino"})

{ "\_id" : ObjectId("60e6dbd028280e9b364d23f4"), "title" : "Pulp Fiction", "writer" : "Quentin Tarantino", "year" : "1994", "actors" : [ "John Travolta", "Uma Thurman" ] }

{ "\_id" : ObjectId("60e6dbed28280e9b364d23f5"), "title" : "Inglorious Basterds", "writer" : "Quentin Tarantino", "year" : "2009", "actors" : [ "Brad Pitt", "Diane Kruger", "Eli Roth" ] }

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

MongoDB Enterprise atlas-125c6m-shard-0:PRIMARY> db.movies.find({actors:"Brad Pitt"})

{ "\_id" : ObjectId("60e6db6028280e9b364d23f3"), "title" : "Fight Club", "writer" : "Chuck Palahniuk", "year" : "1999", "actors" : [ "Brad Pitt", "Edward Norton" ] }

{ "\_id" : ObjectId("60e6dbed28280e9b364d23f5"), "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"

MongoDB Enterprise atlas-125c6m-shard-0:PRIMARY> db.movies.find({franchise:"The Hobbit"})

{ "\_id" : ObjectId("60e6dc1728280e9b364d23f6"), "title" : "The Hobbit: An unexpected Journey", "writer" : "J.R.R. Tolkein", "year" : "2012", "franchise" : "The Hobbit" }

{ "\_id" : ObjectId("60e6dc3528280e9b364d23f7"), "title" : "The Hobbit: The Desolation of Smaug", "writer" : "J.R.R Tolkien", "year" : "2013", "franchise" : "The Hobbit" }

{ "\_id" : ObjectId("60e6dc5e28280e9b364d23f8"), "title" : "The Hobbit: The Battle of the Five Armies", "writer" : "J.R.R Tolkien", "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." }

1. get all movies released in the 90s

MongoDB Enterprise atlas-125c6m-shard-0:PRIMARY> db.movies.find({year:{$gte:"1990", $lt:"2000"}})

{ "\_id" : ObjectId("60e6db6028280e9b364d23f3"), "title" : "Fight Club", "writer" : "Chuck Palahniuk", "year" : "1999", "actors" : [ "Brad Pitt", "Edward Norton" ] }

{ "\_id" : ObjectId("60e6dbd028280e9b364d23f4"), "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

MongoDB Enterprise atlas-125c6m-shard-0:PRIMARY> db.movies.find({$or: [{year:{$gt: "2010"}},{year: {$lt: "2000"}}]})

{ "\_id" : ObjectId("60e6db6028280e9b364d23f3"), "title" : "Fight Club", "writer" : "Chuck Palahniuk", "year" : "1999", "actors" : [ "Brad Pitt", "Edward Norton" ] }

{ "\_id" : ObjectId("60e6dbd028280e9b364d23f4"), "title" : "Pulp Fiction", "writer" : "Quentin Tarantino", "year" : "1994", "actors" : [ "John Travolta", "Uma Thurman" ] }

{ "\_id" : ObjectId("60e6dc1728280e9b364d23f6"), "title" : "The Hobbit: An unexpected Journey", "writer" : "J.R.R. Tolkein", "year" : "2012", "franchise" : "The Hobbit" }

{ "\_id" : ObjectId("60e6dc3528280e9b364d23f7"), "title" : "The Hobbit: The Desolation of Smaug", "writer" : "J.R.R Tolkien", "year" : "2013", "franchise" : "The Hobbit" }

{ "\_id" : ObjectId("60e6dc5e28280e9b364d23f8"), "title" : "The Hobbit: The Battle of the Five Armies", "writer" : "J.R.R Tolkien", "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."

MongoDB Enterprise atlas-125c6m-shard-0:PRIMARY> db.movies.update({\_id:ObjectId("60e6dc1728280e9b364d23f6")}, {$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."}})

WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

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."

MongoDB Enterprise atlas-125c6m-shard-0:PRIMARY> db.movies.update({\_id:ObjectId("60e6dc3528280e9b364d23f7")}, {$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."}})

WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

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

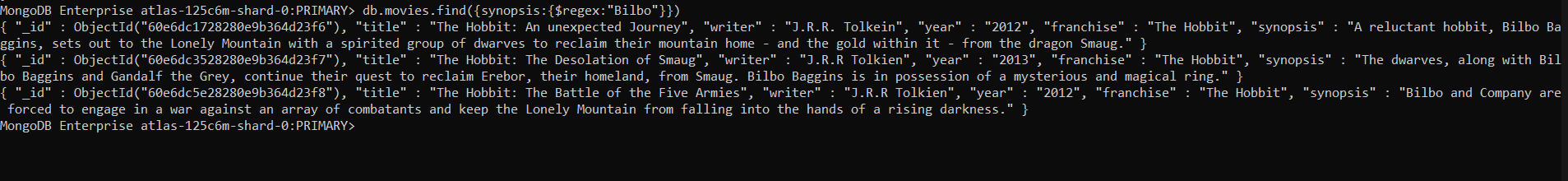
MongoDB Enterprise atlas-125c6m-shard-0:PRIMARY> db.movies.update({\_id:ObjectId("60e6dbd028280e9b364d23f4")}, {$push:{actors:"Samuel L. Jackson"}})

WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

**Text Search**

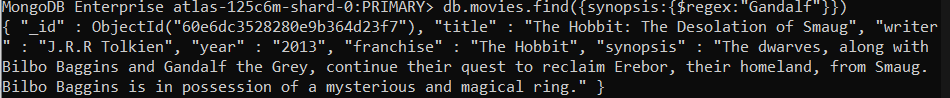
1. find all movies that have a synopsis that contains the word "Bilbo"

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



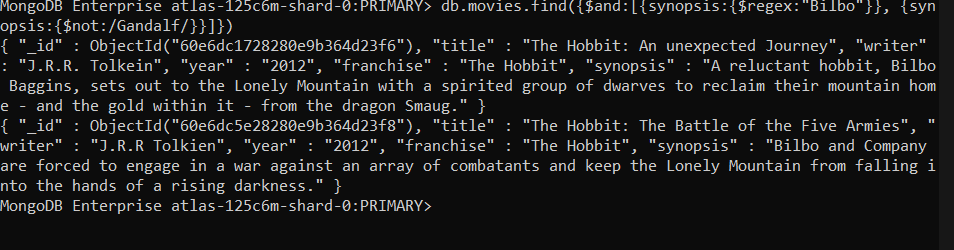
1. find all movies that have a synopsis that contains the word "Gandalf"

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



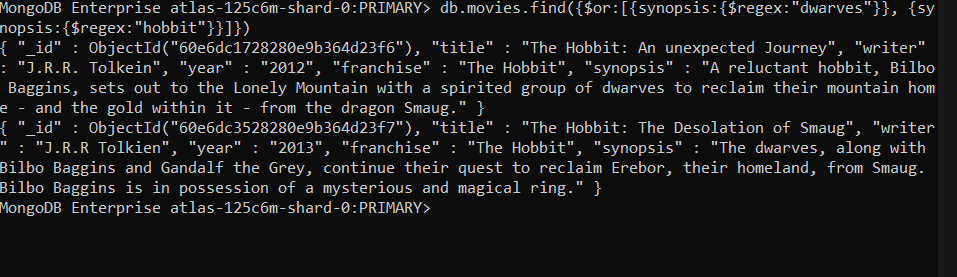
1. find all movies that have a synopsis that contains the word "Bilbo" and not the word "Gandalf"

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



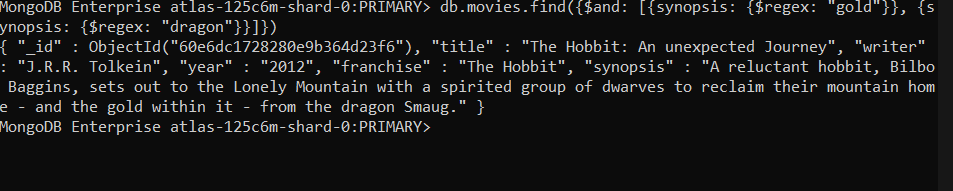
1. find all movies that have a synopsis that contains the word "dwarves" or "hobbit"

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

****

**Delete Documents**

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

MongoDB Enterprise atlas-125c6m-shard-0:PRIMARY> db.movies.remove({\_id:ObjectId("60e6dc7b28280e9b364d23f9")})

WriteResult({ "nRemoved" : 1 })

1. delete the movie "Avatar"

MongoDB Enterprise atlas-125c6m-shard-0:PRIMARY> db.movies.remove({\_id:ObjectId("60e6dcb428280e9b364d23fa")})

WriteResult({ "nRemoved" : 1 })

**Relationships**

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

* username : GoodGuyGreg

first\_name : "Good Guy"

last\_name : "Greg"

MongoDB Enterprise atlas-125c6m-shard-0:PRIMARY> db.users.insert({username:"GoodGuyGreg", first\_name:"Good Guy", last\_name:"Greg"})

WriteResult({ "nInserted" : 1 })

* username : ScumbagSteve

full\_name :

first : "Scumbag"

last : "Steve"

MongoDB Enterprise atlas-125c6m-shard-0:PRIMARY> db.users.insert({username:"ScumbagSteve", fullname:{first: "Scumbag", last:"Steve"}})

WriteResult({ "nInserted" : 1 })

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

* username : GoodGuyGreg

title : Passes out at party

body : Wakes up early and cleans house

MongoDB Enterprise atlas-125c6m-shard-0:PRIMARY> db.posts.insert({username:"GoodGuyGreg", title:"Passes out at Party", body:"Raises your credit score"})

WriteResult({ "nInserted" : 1 })

* username : GoodGuyGreg

title : Steals your identity

body : Raises your credit score

MongoDB Enterprise atlas-125c6m-shard-0:PRIMARY> db.posts.insert({ username:"GoodGuyGreg", title:"Steals your identity", body:"Raises your credit score"})

WriteResult({ "nInserted" : 1 })

* username : GoodGuyGreg

title : Reports a bug in your code

body : Sends you a Pull Request

MongoDB Enterprise atlas-125c6m-shard-0:PRIMARY> db.posts.insert({username:"GoodGuyGreg", title:"Reports a bug in your code", body:"Sends you a pull request"})

WriteResult({ "nInserted" : 1 })

* username : ScumbagSteve

title : Borrows something

body : Sells it

MongoDB Enterprise atlas-125c6m-shard-0:PRIMARY> db.posts.insert({ username:"ScumbagSteve", title:"Borrows something", body:"Sells it"})

WriteResult({ "nInserted" : 1 })

* username : ScumbagSteve

title : Borrows everything

body : The end

MongoDB Enterprise atlas-125c6m-shard-0:PRIMARY> db.posts.insert({ username:"ScumbagSteve", title:"Borrows everything", body:"The end"})

WriteResult({ "nInserted" : 1 })

* username : ScumbagSteve

title : Forks your repo on github

body : Sets to private

MongoDB Enterprise atlas-125c6m-shard-0:PRIMARY> db.posts.insert({username:"ScumbagSteve", title:"Forks your repo on github", body:"Sets to private"})

WriteResult({ "nInserted" : 1 })

**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"

MongoDB Enterprise atlas-125c6m-shard-0:PRIMARY> db.comments.insert({ username:"GoodGuyGreg", comment:"Hope you got a good deal!", post:ObjectId("60e6fa0728280e9b364d2401")})

WriteResult({ "nInserted" : 1 })

* 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"

MongoDB Enterprise atlas-125c6m-shard-0:PRIMARY> db.comments.insert({username:"GoodGuyGreg", comment:"What's mine is yours!", post:ObjectId("60e6fa3528280e9b364d2402")})

WriteResult({ "nInserted" : 1 })

* 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

MongoDB Enterprise atlas-125c6m-shard-0:PRIMARY> db.comments.insert({username:"GoodGuyGreg", comment:"Don't violate the licensing agreement!", post:ObjectId("60e6fa4828280e9b364d2403")})

WriteResult({ "nInserted" : 1 }

* 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"

MongoDB Enterprise atlas-125c6m-shard-0:PRIMARY> db.comments.insert({username:"ScumbagSteve", comment:"It still isn't clean", post:ObjectId("60e6f9c728280e9b364d23fe")})

WriteResult({ "nInserted" : 1 })

* 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"

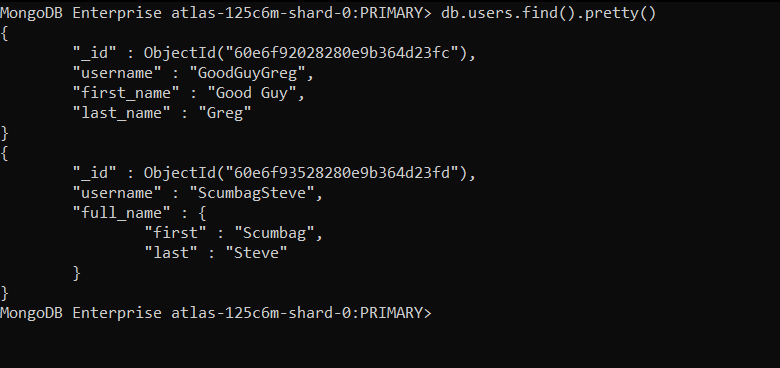
MongoDB Enterprise atlas-125c6m-shard-0:PRIMARY> db.comments.insert({username:"ScumbagSteve", comment:"Denied your PR cause I found a hack", post:ObjectId("60e6f9f028280e9b364d2400")})

WriteResult({ "nInserted" : 1 })

**Querying related collections**

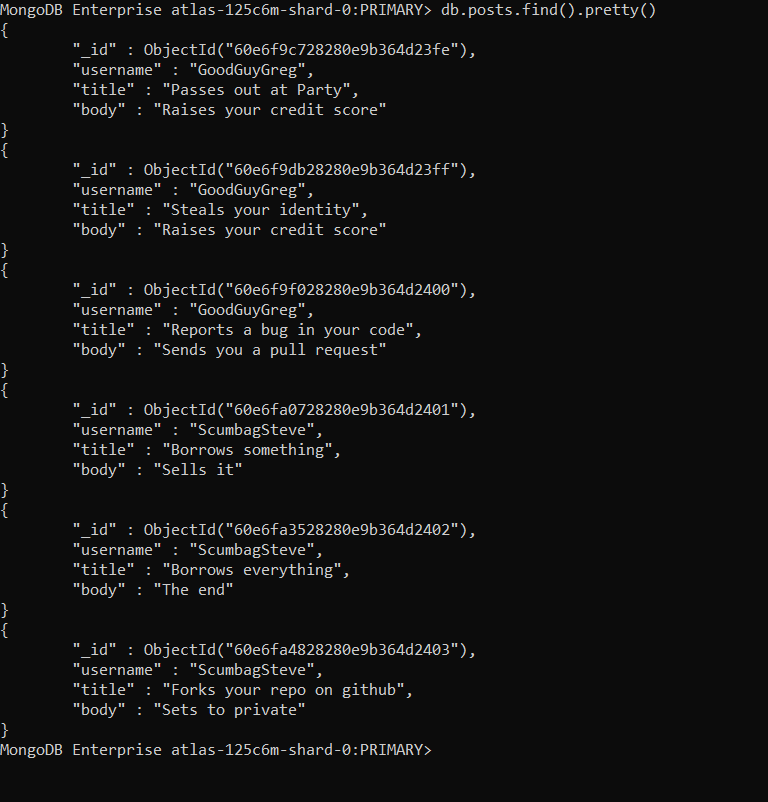
1. find all users

db.users.find().pretty()



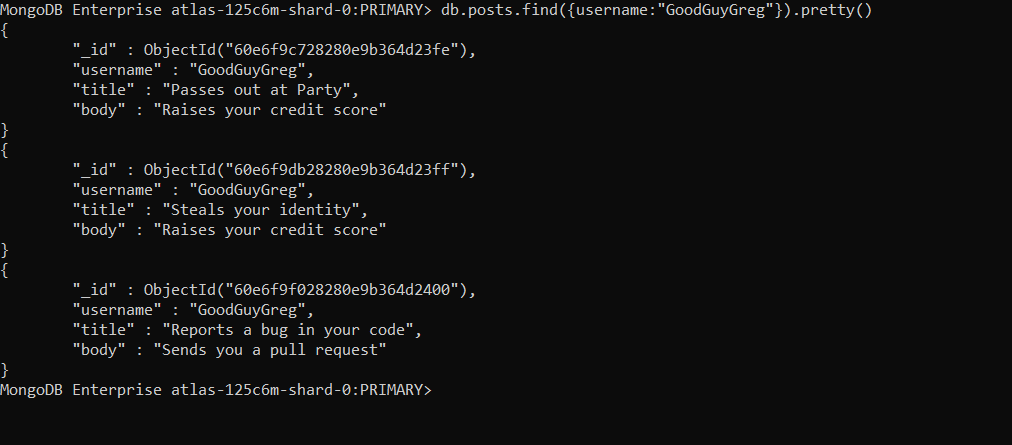
1. find all posts

db.posts.find().pretty()



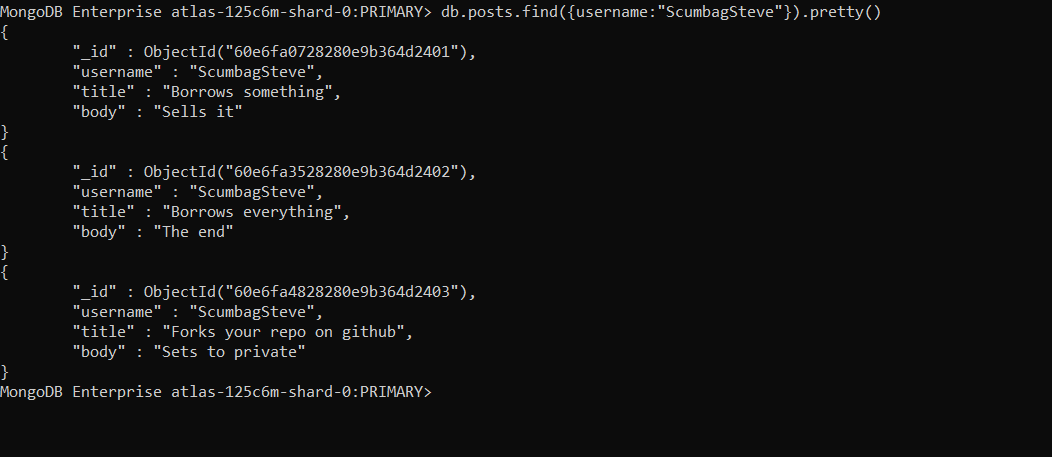
1. find all posts that was authored by "GoodGuyGreg"

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



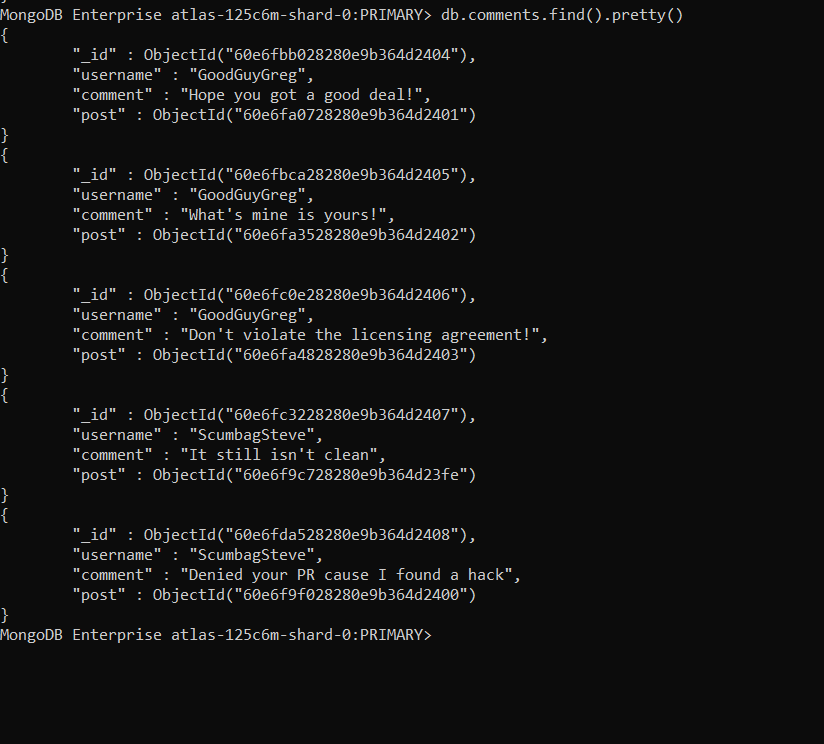
1. find all posts that was authored by "ScumbagSteve"

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



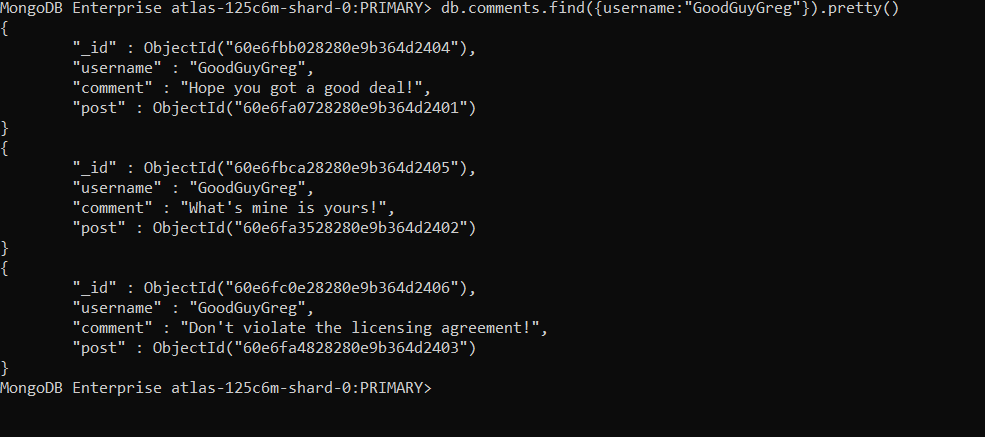
1. find all comments

db.comments.find().pretty()



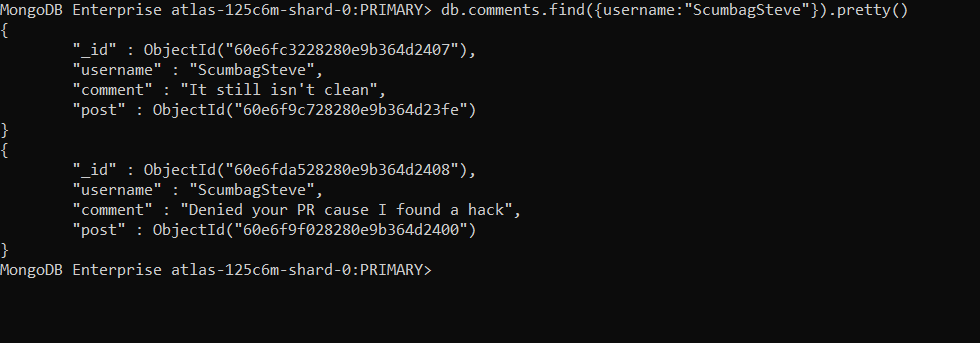
1. find all comments that was authored by "GoodGuyGreg"

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



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

db.comments.find({username:"ScumbagSteve"}).pretty()



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

db.comments.find({"post": ObjectId("60e6f9f028280e9b364d2400")}).pretty()

