**MONGODB QUERIES**

MOVIE DATABASE

db.movies.insertMany([

{"title" : "Fight Club",

"writer" : "Chuck Palahniuk",

"year" : 1999,

"actors" : [

"Brad Pitt",

"Edward Norton"

]},

{"title" : "Pulp Fiction",

"writer" : "Quentin Tarantino",

"year" : 1994,

"actors" : [

"John Travolta",

"Uma Thurman"

]},

{"title" : "Inglourious 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,

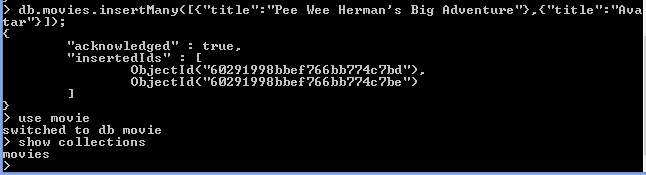
"franchise" : "The Hobbit",

"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"}]);





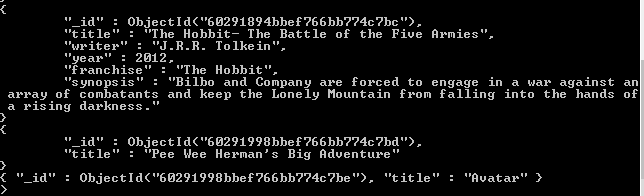
## Query / Find Documents

query the movies collection to

1. get all documents

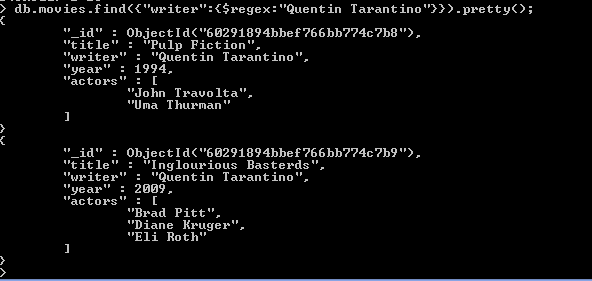
db.movies.find().pretty();





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

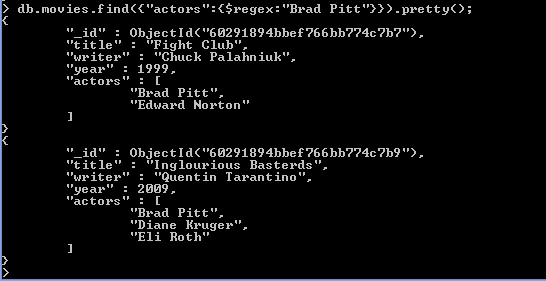
db.movies.find({"writer":{$regex:"Quentin Tarantino"}}).pretty();



Note:$regex has to be a string("").

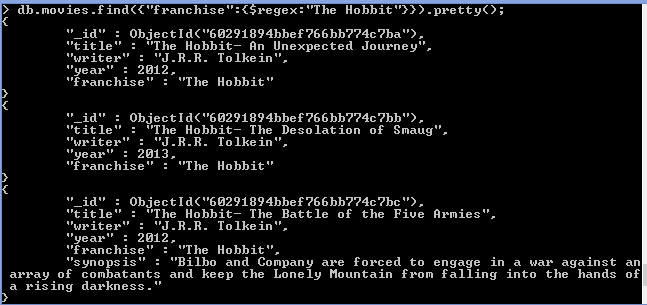
1. get all documents where actors include "Brad Pitt"

db.movies.find({"actors":{$regex:"Brad Pitt"}}).pretty();



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

db.movies.find({"franchise":{$regex:"The Hobbit"}}).pretty();



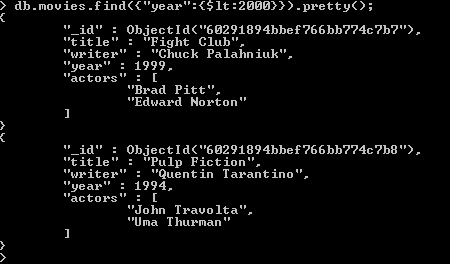
1. get all movies released in the 90s

db.movies.find({"year":{$gt:1989, $lt:2000}}).pretty();



or

db.movies.find({"year":{$lt:2000}}).pretty();



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

db.movies.find({$or:[ {year:{$gt:2010}}, {year:{$lt:2000}} ]}).pretty();

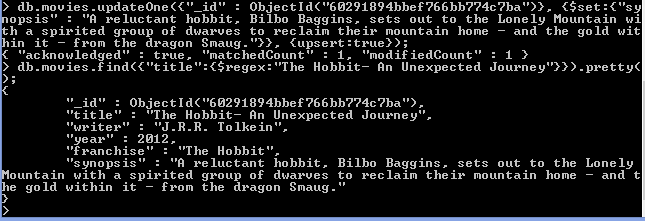


Note:$or must be an array.

## Update Documents

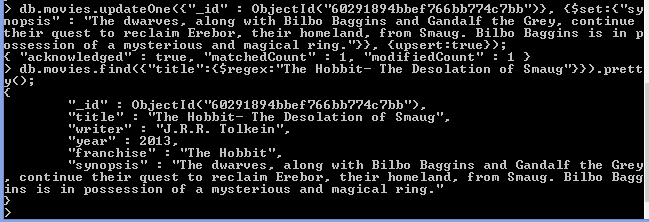
1. add a synopsis to "The Hobbit: An Unexpected Journey" : "A reluctant hobbit, Bilbo Baggins, sets out to the Lonely Mountain with a spirited group of dwarves to reclaim their mountain home - and the gold within it - from the dragon Smaug."

db.movies.updateOne({"\_id" : ObjectId("60291894bbef766bb774c7ba")}, {$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."}}, {upsert:true});



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."

db.movies.updateOne({"\_id" : ObjectId("60291894bbef766bb774c7bb")}, {$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."}}, {upsert:true});



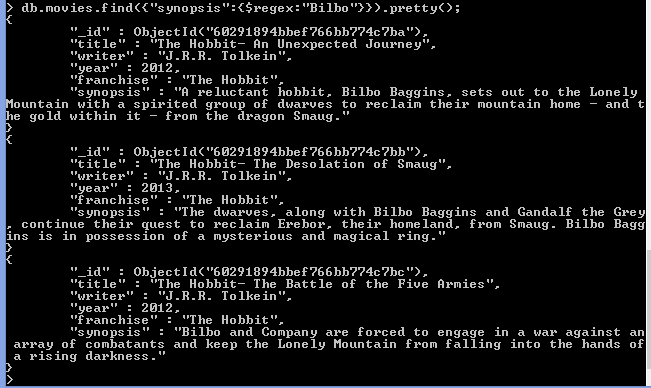
1. add an actor named "Samuel L. Jackson" to the movie "Pulp Fiction"

db.movies.updateOne({"\_id" : ObjectId("602936e9bbef766bb774c7b8")}, {$set:{["actors" : "Samuel L. Jackson"]}}, {upsert:true});

## Text Search

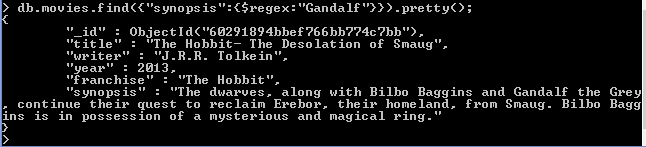
1. find all movies that have a synopsis that contains the word "Bilbo"

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



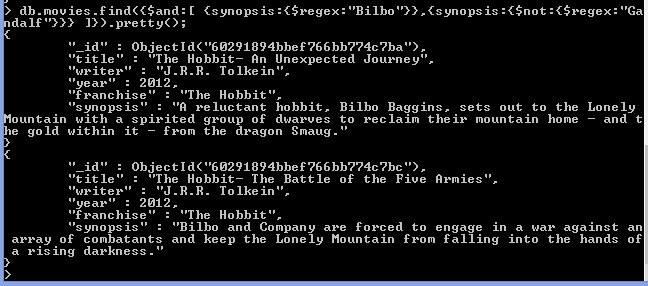
1. find all movies that have a synopsis that contains the word "Gandalf"

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



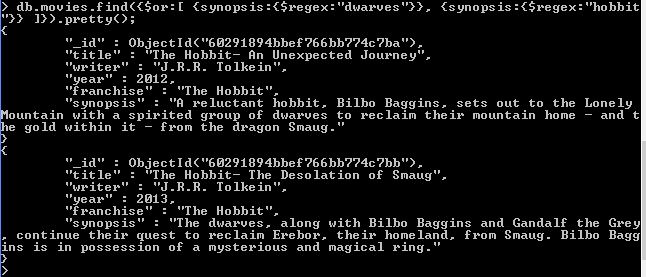
1. find all movies that have a synopsis that contains the word "Bilbo" and not the word "Gandalf"

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



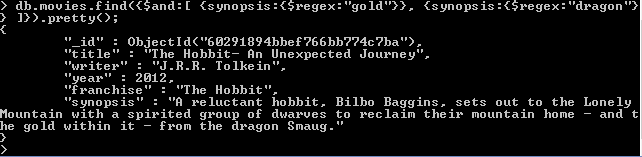
1. find all movies that have a synopsis that contains the word "dwarves" or "hobbit"

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



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"}} ]}).pretty();



## Delete Documents

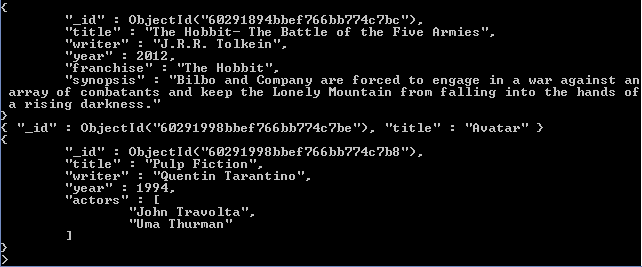
1. delete the movie "Pee Wee Herman's Big Adventure"

db.movies.deleteOne({"\_id":ObjectId("60291998bbef766bb774c7bd"),"title":"Pee Wee Herman's Big Adventure"});

OR

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

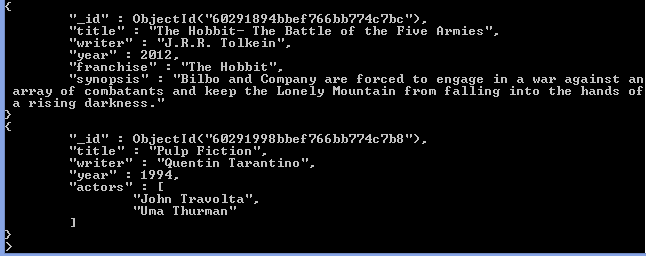




1. delete the movie "Avatar"

db.movies.deleteOne({"\_id":ObjectId("60291998bbef766bb774c7be"),"title":"Avatar});





## Relationships

### Insert the following documents into a users collection

([

{"username" : "GoodGuyGreg",

"first\_name" : "Good Guy",

"last\_name" : "Greg"

},

{"username" : "ScumbagSteve",

"full\_name" :[

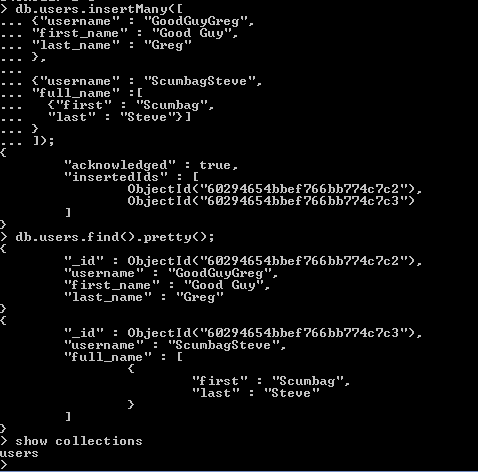
{"first" : "Scumbag",

"last" : "Steve"}]

}

]);







### 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"}]);







### Insert the following documents into a comments collection

db.comments.insertOne(

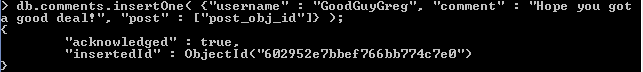
{"username" : "GoodGuyGreg",

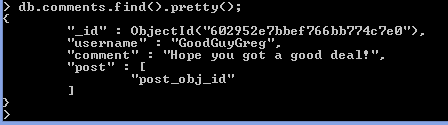
"comment" : "Hope you got a good deal!",

"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]

db.posts.updateOne({"\_id":ObjectId("60294cfdbbef766bb774c7ca")},{post:[post\_obj\_id:{ObjectId("60294cfdbbef766bb774c7cf")}]},{upsert:true});

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]

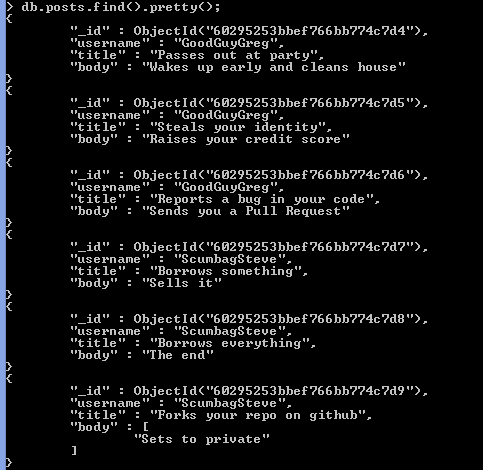
where [post\_obj\_id] is the ObjectId of the posts document: "Reports a bug in your code"

## Querying related collections

1. find all users

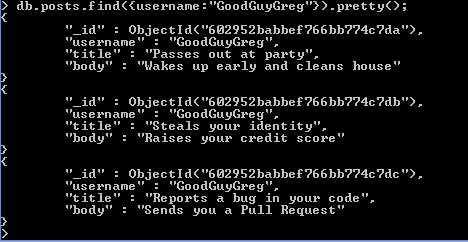


1. find all posts



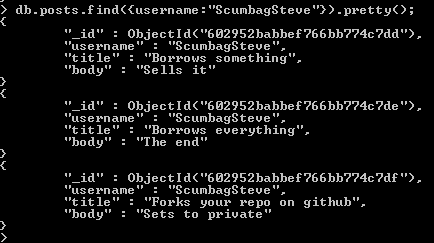
1. find all posts that was authored by "GoodGuyGreg"

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



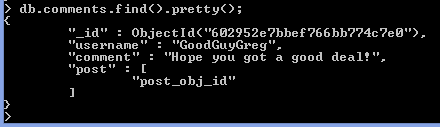
1. find all posts that was authored by "ScumbagSteve"

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



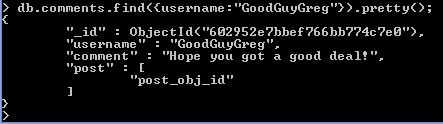
1. find all comments

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({username:"ScumbagSteve"}).pretty();