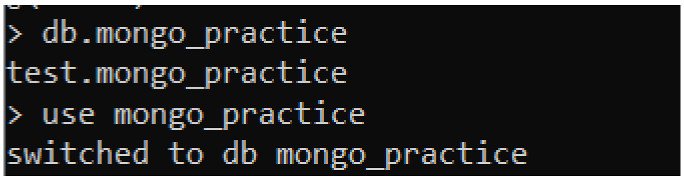
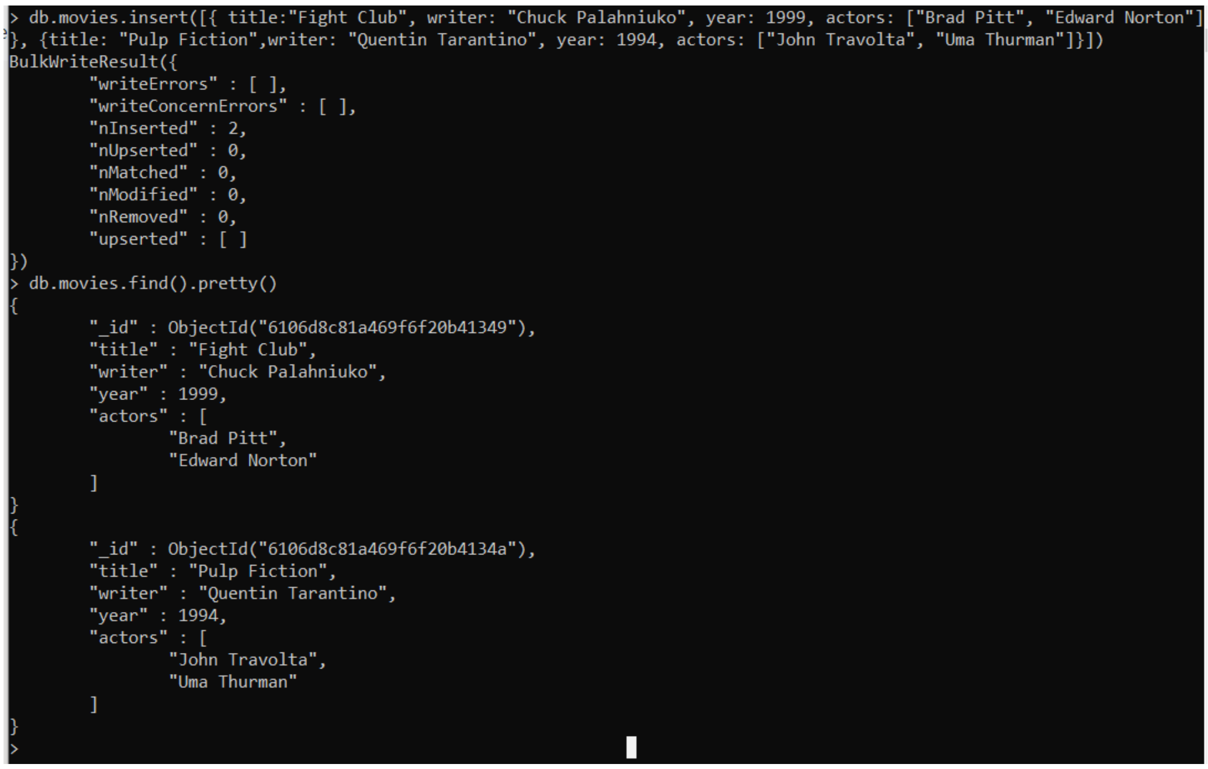
Assignment 1:

Connect to a running mongo instance, use a database named mongo_practice

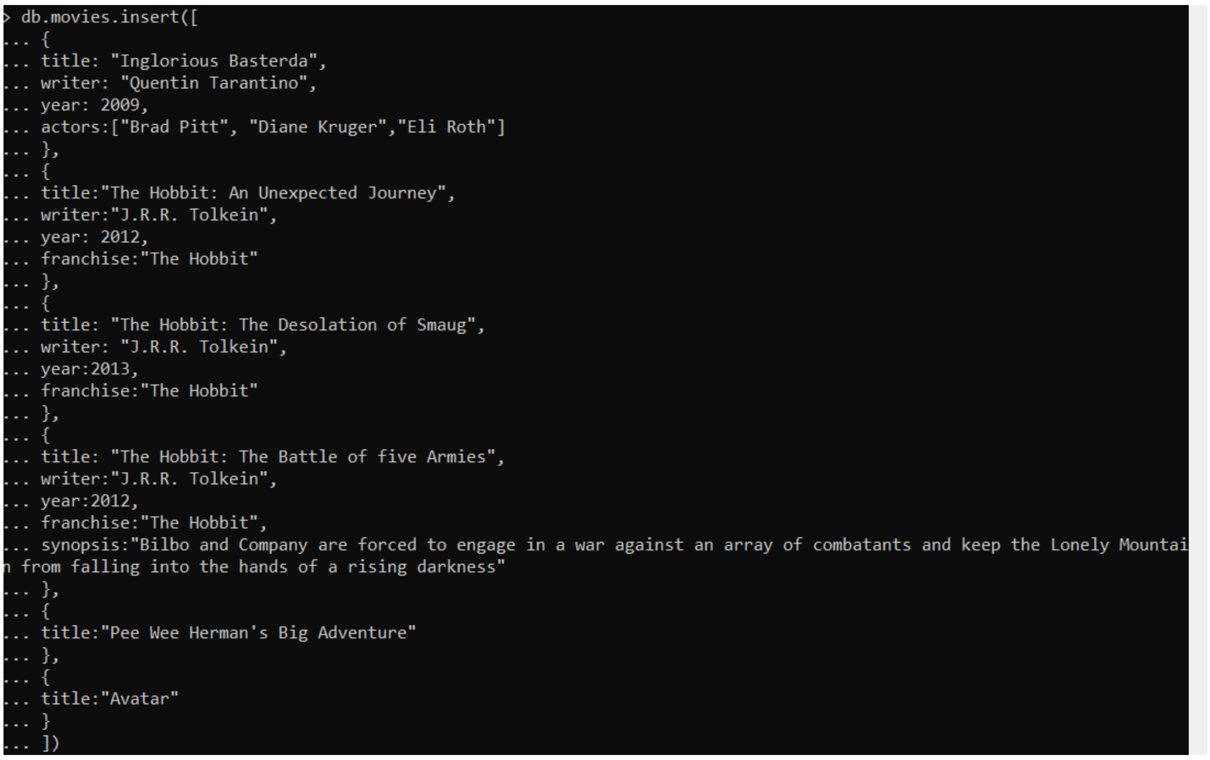


```
> db.mongo_practice
test.mongo_practice
> use mongo_practice
switched to db mongo_practice
```

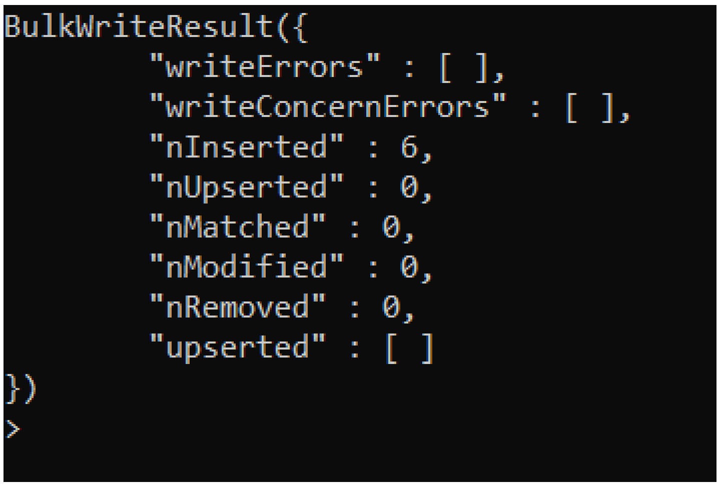
Insert following documents into a movies collection



```
> db.movies.insert([{ 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"]}])
BulkWriteResult({
        "writeErrors" : [ ],
        "writeConcernErrors" : [ ],
        "nInserted" : 2,
        "nUpserted" : 0,
        "nMatched" : 0,
        "nModified" : 0,
        "nRemoved" : 0,
        "upserted" : [ ]
})
> db.movies.find().pretty()
{
        "_id" : ObjectId("6106d8c81a469f6f20b41349"),
        "title" : "Fight Club",
        "writer" : "Chuck Palahniuko",
        "year" : 1999,
        "actors" : [
                "Brad Pitt",
                "Edward Norton"
        ]
}
{
        "_id" : ObjectId("6106d8c81a469f6f20b4134a"),
        "title" : "Pulp Fiction",
        "writer" : "Quentin Tarantino",
        "year" : 1994,
        "actors" : [
                "John Travolta",
                "Uma Thurman"
        ]
}
>
```



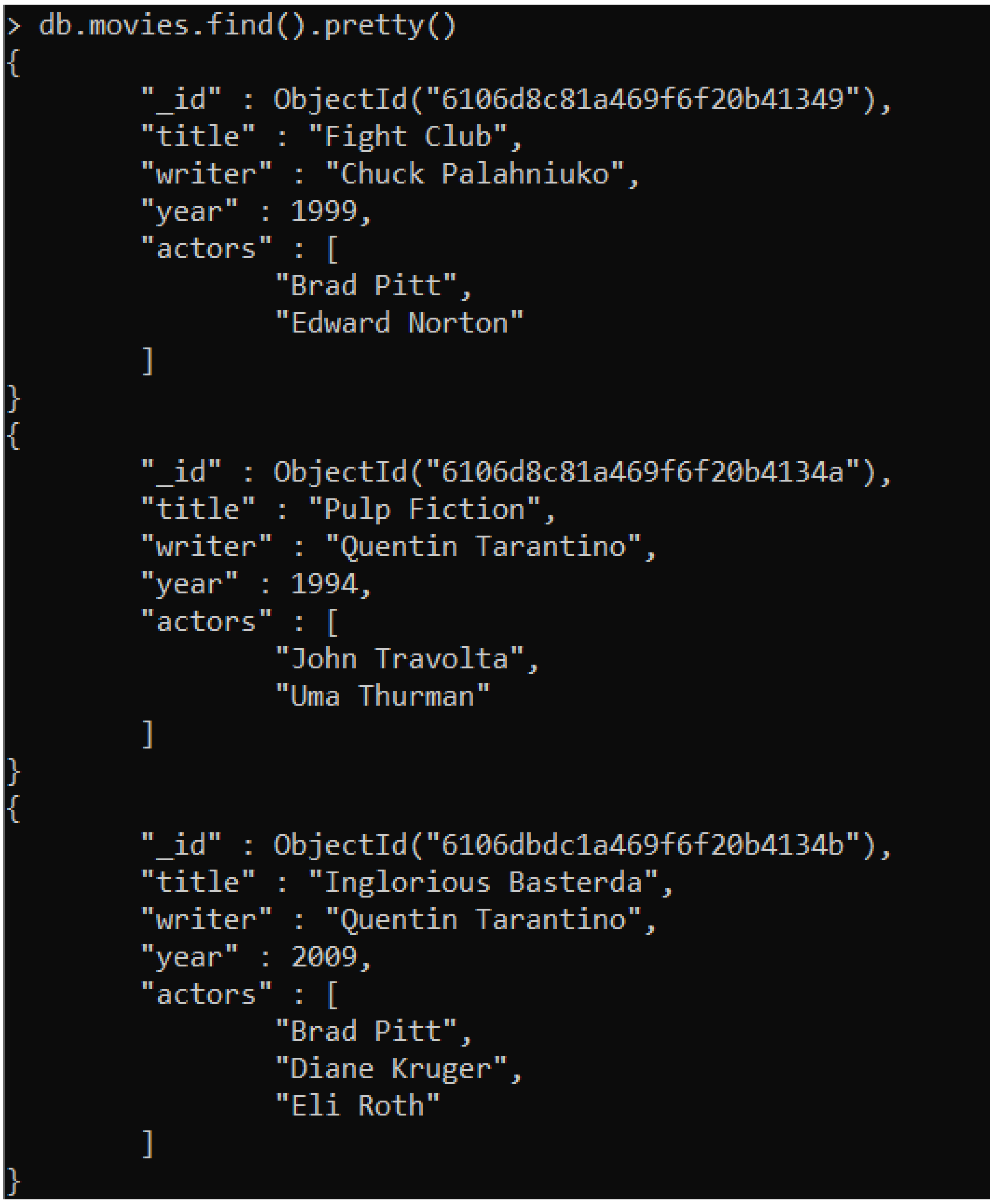
```
> db.movies.insert([
... {
... title: "Inglorious Basterda",
... writer: "Quentin Tarantino",
... year: 2009,
... actors:["Brad Pitt", "Diane Kruger","Eli Roth"]
... },
... {
... title:"The Hobbit: An Unexpected Journey",
... writer:"J.R.R. Tolkein",
... year: 2012,
... franchise:"The Hobbit"
... },
... {
... title: "The Hobbit: The Desolation of Smaug",
... writer: "J.R.R. Tolkein",
... year:2013,
... franchise:"The Hobbit"
... },
... {
... title: "The Hobbit: The Battle of 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 Mountai
n from falling into the hands of a rising darkness"
... },
... {
... title:"Pee Wee Herman's Big Adventure"
... },
... {
... title:"Avatar"
... }
... ])
```



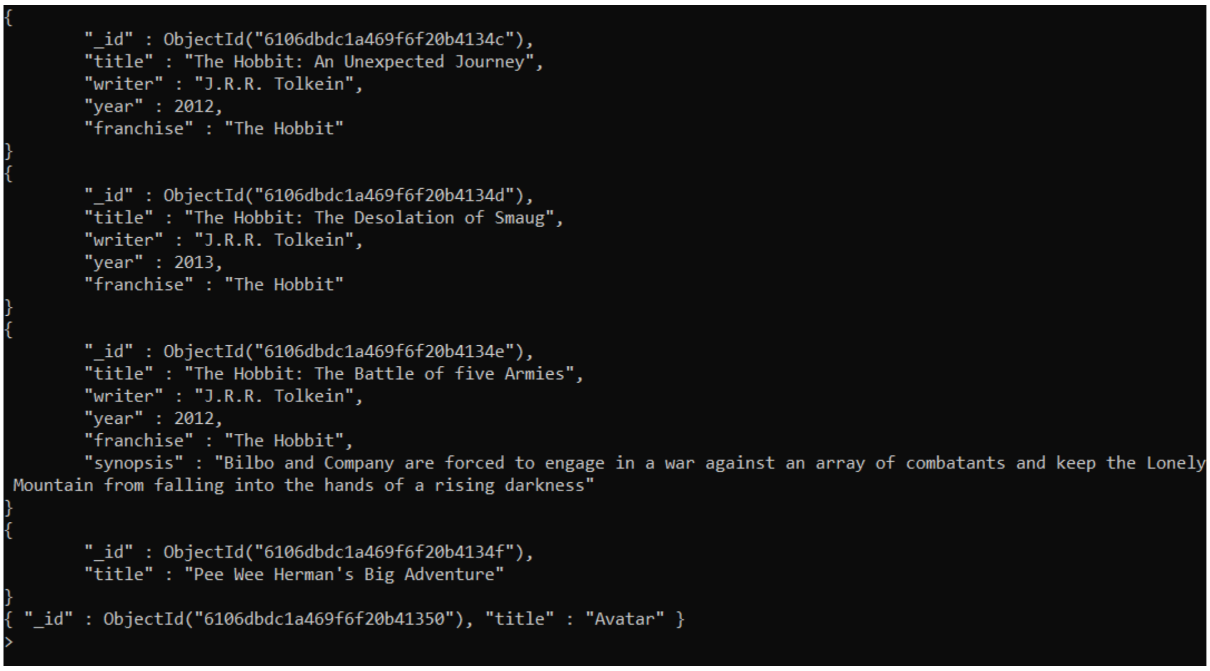
```
BulkWriteResult({
        "writeErrors" : [ ],
        "writeConcernErrors" : [ ],
        "nInserted" : 6,
        "nUpserted" : 0,
        "nMatched" : 0,
        "nModified" : 0,
        "nRemoved" : 0,
        "upserted" : [ ]
})
>
```

Query/Find Documents

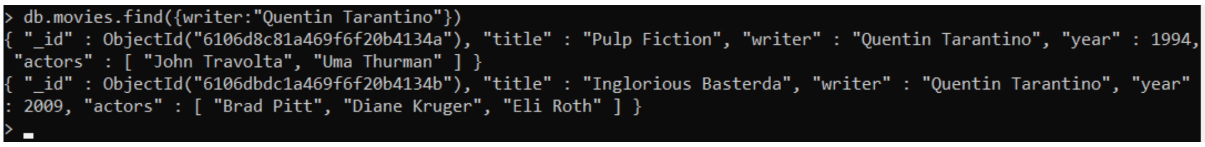
1) Get all documents:



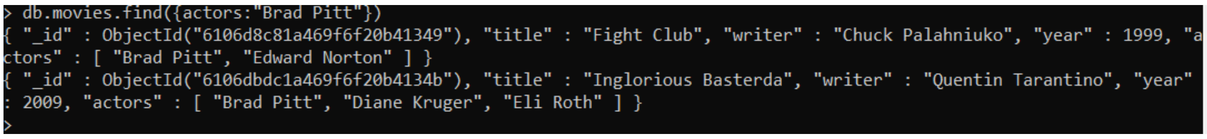
```
> db.movies.find().pretty()
{
        "_id" : ObjectId("6106d8c81a469f6f20b41349"),
        "title" : "Fight Club",
        "writer" : "Chuck Palahniuko",
        "year" : 1999,
        "actors" : [
                "Brad Pitt",
                "Edward Norton"
        ]
}
{
        "_id" : ObjectId("6106d8c81a469f6f20b4134a"),
        "title" : "Pulp Fiction",
        "writer" : "Quentin Tarantino",
        "year" : 1994,
        "actors" : [
                "John Travolta",
                "Uma Thurman"
        ]
}
{
        "_id" : ObjectId("6106dbdc1a469f6f20b4134b"),
        "title" : "Inglorious Basterda",
        "writer" : "Quentin Tarantino",
        "year" : 2009,
        "actors" : [
                "Brad Pitt",
                "Diane Kruger",
                "Eli Roth"
        ]
}
```


```
{
        "_id" : ObjectId("6106dbdc1a469f6f20b4134c"),
        "title" : "The Hobbit: An Unexpected Journey",
        "writer" : "J.R.R. Tolkein",
        "year" : 2012,
        "franchise" : "The Hobbit"
}
{
        "_id" : ObjectId("6106dbdc1a469f6f20b4134d"),
        "title" : "The Hobbit: The Desolation of Smaug",
        "writer" : "J.R.R. Tolkein",
        "year" : 2013,
        "franchise" : "The Hobbit"
}
{
        "_id" : ObjectId("6106dbdc1a469f6f20b4134e"),
        "title" : "The Hobbit: The Battle of 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("6106dbdc1a469f6f20b4134f"),
        "title" : "Pee Wee Herman's Big Adventure"
}
{ "_id" : ObjectId("6106dbdc1a469f6f20b41350"), "title" : "Avatar" }
>
```

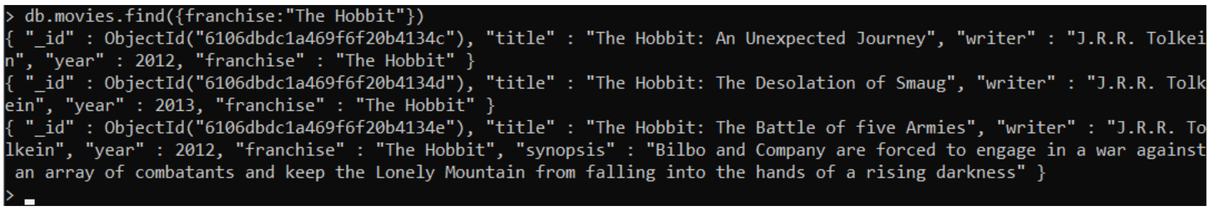
2)writer Quentin Tarantino


```
> db.movies.find({writer:"Quentin Tarantino"})
{ "_id" : ObjectId("6106d8c81a469f6f20b4134a"), "title" : "Pulp Fiction", "writer" : "Quentin Tarantino", "year" : 1994,
 "actors" : [ "John Travolta", "Uma Thurman" ] }
{ "_id" : ObjectId("6106dbdc1a469f6f20b4134b"), "title" : "Inglorious Basterda", "writer" : "Quentin Tarantino", "year"
: 2009, "actors" : [ "Brad Pitt", "Diane Kruger", "Eli Roth" ] }
>
```

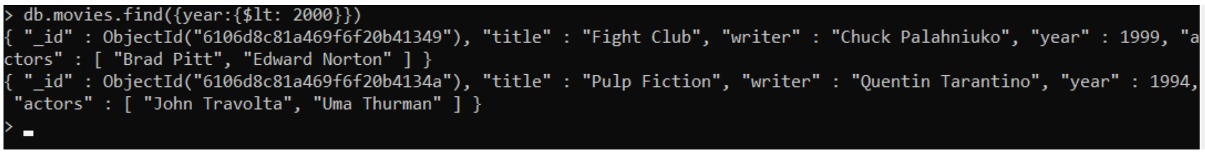
3) Actor : Brad Pitt


```
> db.movies.find({actors:"Brad Pitt"})
{ "_id" : ObjectId("6106d8c81a469f6f20b41349"), "title" : "Fight Club", "writer" : "Chuck Palahniuko", "year" : 1999, "a
ctors" : [ "Brad Pitt", "Edward Norton" ] }
{ "_id" : ObjectId("6106dbdc1a469f6f20b4134b"), "title" : "Inglorious Basterda", "writer" : "Quentin Tarantino", "year"
: 2009, "actors" : [ "Brad Pitt", "Diane Kruger", "Eli Roth" ] }
>
```

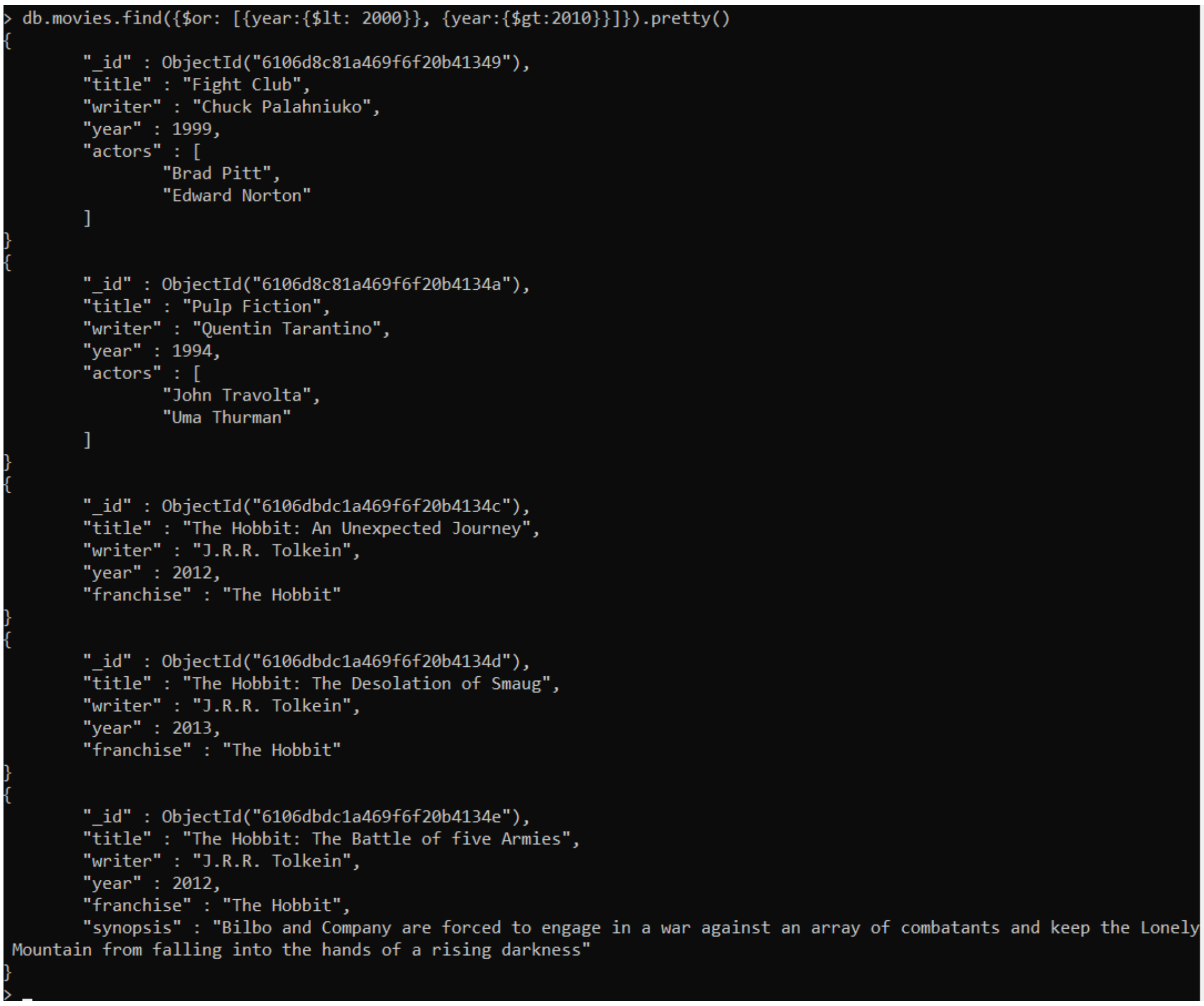
4) Frenchise Hobbit


```
> db.movies.find({franchise:"The Hobbit"})
{ "_id" : ObjectId("6106dbdc1a469f6f20b4134c"), "title" : "The Hobbit: An Unexpected Journey", "writer" : "J.R.R. Tolkei
n", "year" : 2012, "franchise" : "The Hobbit" }
{ "_id" : ObjectId("6106dbdc1a469f6f20b4134d"), "title" : "The Hobbit: The Desolation of Smaug", "writer" : "J.R.R. Tolk
ein", "year" : 2013, "franchise" : "The Hobbit" }
{ "_id" : ObjectId("6106dbdc1a469f6f20b4134e"), "title" : "The Hobbit: The Battle of five Armies", "writer" : "J.R.R. To
lkein", "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) Movies released in 90’s


```
> db.movies.find({year:{$lt: 2000}})
{ "_id" : ObjectId("6106d8c81a469f6f20b41349"), "title" : "Fight Club", "writer" : "Chuck Palahniuko", "year" : 1999, "a
ctors" : [ "Brad Pitt", "Edward Norton" ] }
{ "_id" : ObjectId("6106d8c81a469f6f20b4134a"), "title" : "Pulp Fiction", "writer" : "Quentin Tarantino", "year" : 1994,
 "actors" : [ "John Travolta", "Uma Thurman" ] }
>
```

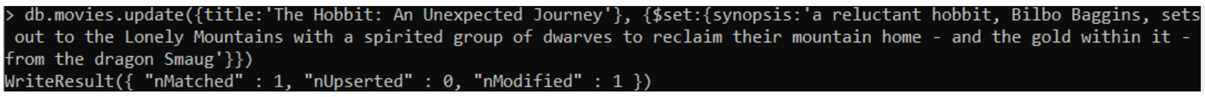
6) Movies released before 2000 or after 2010



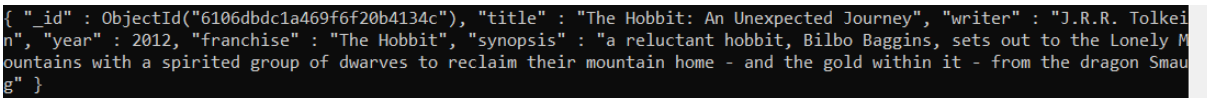
```
> db.movies.find({$or: [{year:{$lt: 2000}}, {year:{$gt:2010}}]}).pretty()
{
        "_id" : ObjectId("6106d8c81a469f6f20b41349"),
        "title" : "Fight Club",
        "writer" : "Chuck Palahniuko",
        "year" : 1999,
        "actors" : [
                "Brad Pitt",
                "Edward Norton"
        ]
}
{
        "_id" : ObjectId("6106d8c81a469f6f20b4134a"),
        "title" : "Pulp Fiction",
        "writer" : "Quentin Tarantino",
        "year" : 1994,
        "actors" : [
                "John Travolta",
                "Uma Thurman"
        ]
}
{
        "_id" : ObjectId("6106dbdc1a469f6f20b4134c"),
        "title" : "The Hobbit: An Unexpected Journey",
        "writer" : "J.R.R. Tolkein",
        "year" : 2012,
        "franchise" : "The Hobbit"
}
{
        "_id" : ObjectId("6106dbdc1a469f6f20b4134d"),
        "title" : "The Hobbit: The Desolation of Smaug",
        "writer" : "J.R.R. Tolkein",
        "year" : 2013,
        "franchise" : "The Hobbit"
}
{
        "_id" : ObjectId("6106dbdc1a469f6f20b4134e"),
        "title" : "The Hobbit: The Battle of 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

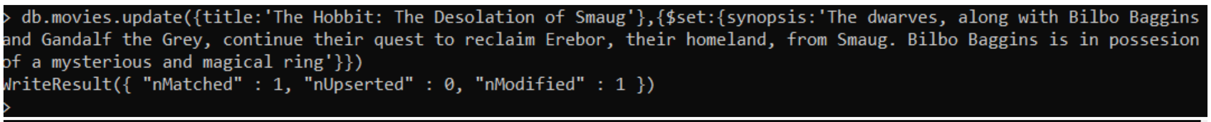


```
> db.movies.update({title:'The Hobbit: An Unexpected Journey'}, {$set:{synopsis:'a reluctant hobbit, Bilbo Baggins, sets
 out to the Lonely Mountains with a spirited group of dwarves to reclaim their mountain home - and the gold within it -
from the dragon Smaug'}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
```

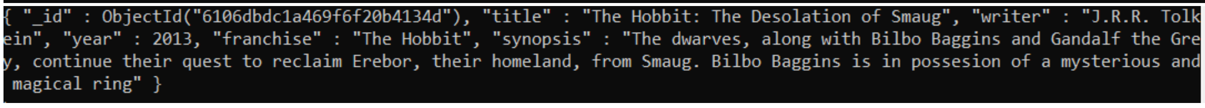


```
{ "_id" : ObjectId("6106dbdc1a469f6f20b4134c"), "title" : "The Hobbit: An Unexpected Journey", "writer" : "J.R.R. Tolkei
n", "year" : 2012, "franchise" : "The Hobbit", "synopsis" : "a reluctant hobbit, Bilbo Baggins, sets out to the Lonely M
ountains with a spirited group of dwarves to reclaim their mountain home - and the gold within it - from the dragon Smau
g" }
```

2) Add synopsis to: The Hobbit: The Desolation of Smaug

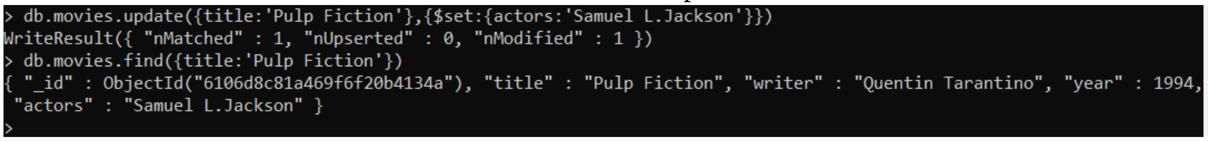


```
> db.movies.update({title:'The Hobbit: The Desolation of Smaug'},{$set:{synopsis:'The dwarves, along with Bilbo Baggins
and Gandalf the Grey, continue their quest to reclaim Erebor, their homeland, from Smaug. Bilbo Baggins is in possesion
of a mysterious and magical ring'}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
>
```



```
{ "_id" : ObjectId("6106dbdc1a469f6f20b4134d"), "title" : "The Hobbit: The Desolation of Smaug", "writer" : "J.R.R. Tolk
ein", "year" : 2013, "franchise" : "The Hobbit", "synopsis" : "The dwarves, along with Bilbo Baggins and Gandalf the Gre
y, continue their quest to reclaim Erebor, their homeland, from Smaug. Bilbo Baggins is in possesion of a mysterious and
 magical ring" }
```

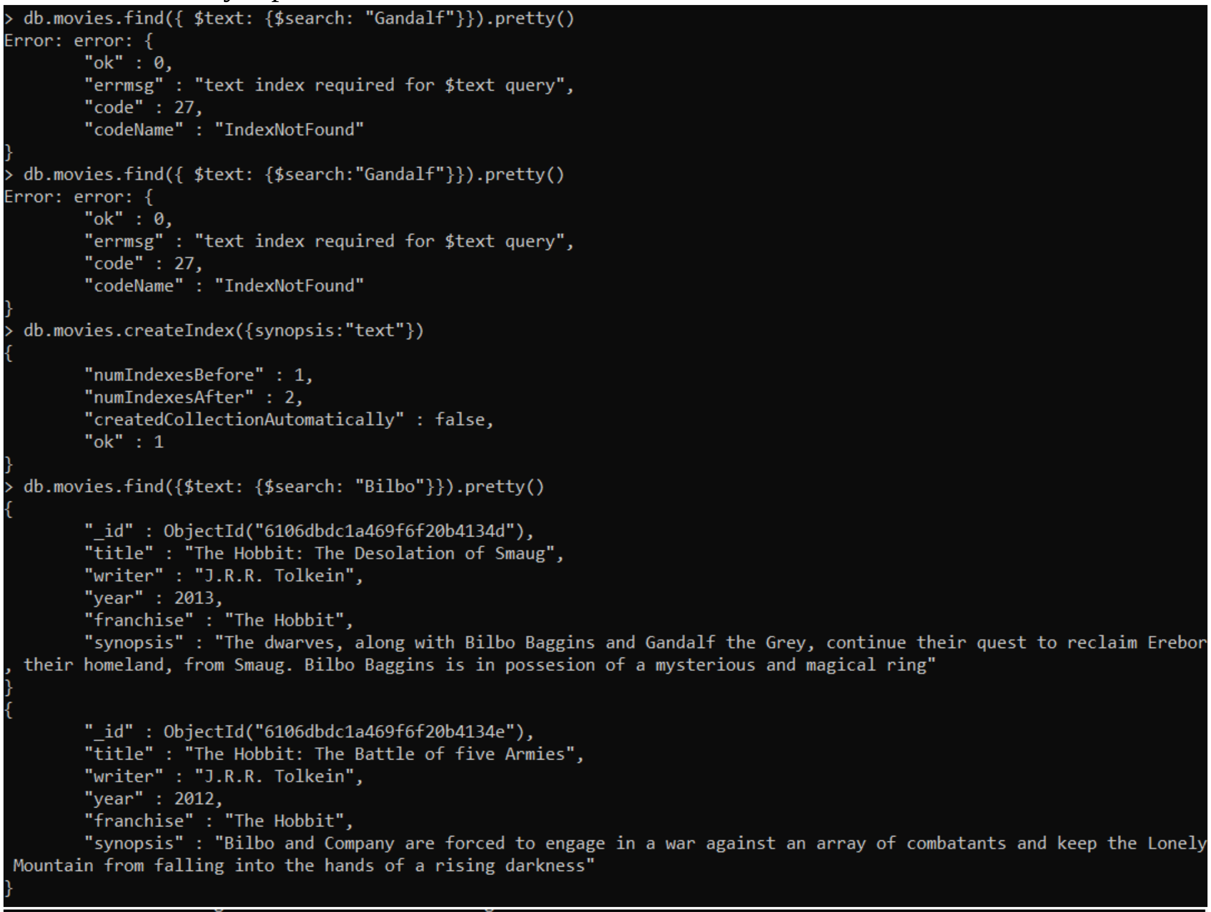
3) Add an actor named Samuel L. Jackson to movie Pulp Fiction



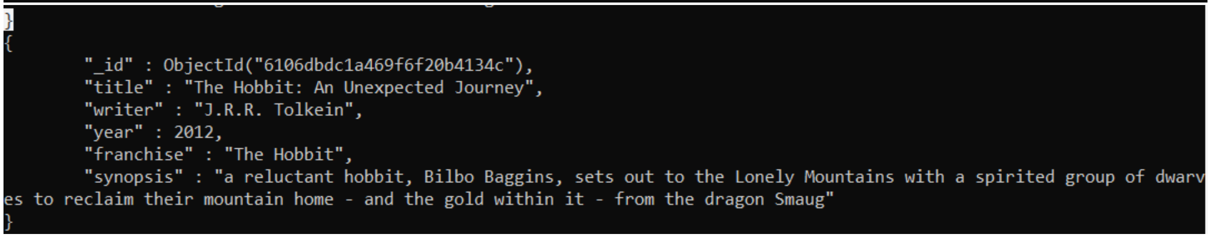
```
> db.movies.update({title:'Pulp Fiction'},{$set:{actors:'Samuel L.Jackson'}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.movies.find({title:'Pulp Fiction'})
{ "_id" : ObjectId("6106d8c81a469f6f20b4134a"), "title" : "Pulp Fiction", "writer" : "Quentin Tarantino", "year" : 1994,
 "actors" : "Samuel L.Jackson" }
>
```

Text Search

1) Find all movies synopsis with word Bilbo

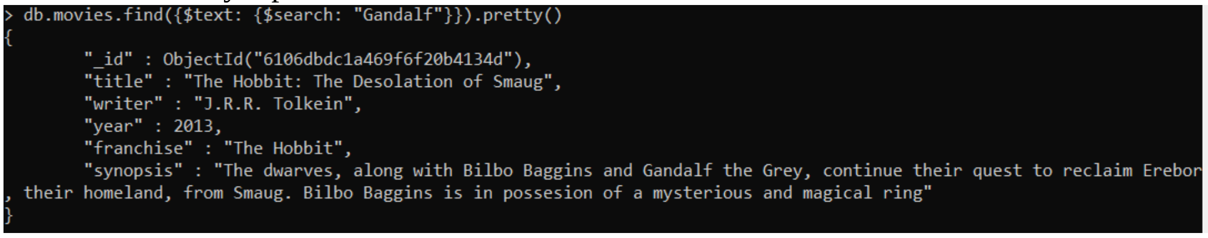


```
> db.movies.find({ $text: {$search: "Gandalf"}}).pretty()
Error: error: {
        "ok" : 0,
        "errmsg" : "text index required for $text query",
        "code" : 27,
        "codeName" : "IndexNotFound"
}
> db.movies.find({ $text: {$search:"Gandalf"}}).pretty()
Error: error: {
        "ok" : 0,
        "errmsg" : "text index required for $text query",
        "code" : 27,
        "codeName" : "IndexNotFound"
}
> db.movies.createIndex({synopsis:"text"})
{
        "numIndexesBefore" : 1,
        "numIndexesAfter" : 2,
        "createdCollectionAutomatically" : false,
        "ok" : 1
}
> db.movies.find({$text: {$search: "Bilbo"}}).pretty()
{
        "_id" : ObjectId("6106dbdc1a469f6f20b4134d"),
        "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 possesion of a mysterious and magical ring"
}
{
        "_id" : ObjectId("6106dbdc1a469f6f20b4134e"),
        "title" : "The Hobbit: The Battle of 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"
}
```



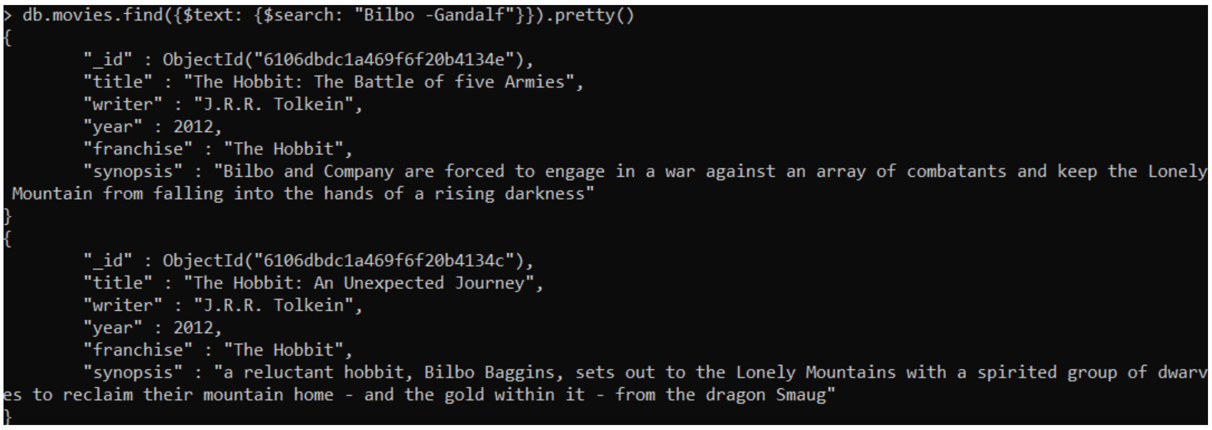
```
}
{
        "_id" : ObjectId("6106dbdc1a469f6f20b4134c"),
        "title" : "The Hobbit: An Unexpected Journey",
        "writer" : "J.R.R. Tolkein",
        "year" : 2012,
        "franchise" : "The Hobbit",
        "synopsis" : "a reluctant hobbit, Bilbo Baggins, sets out to the Lonely Mountains with a spirited group of dwarv
es to reclaim their mountain home - and the gold within it - from the dragon Smaug"
}
```

2) Find all movies synopsis with word Gandalf



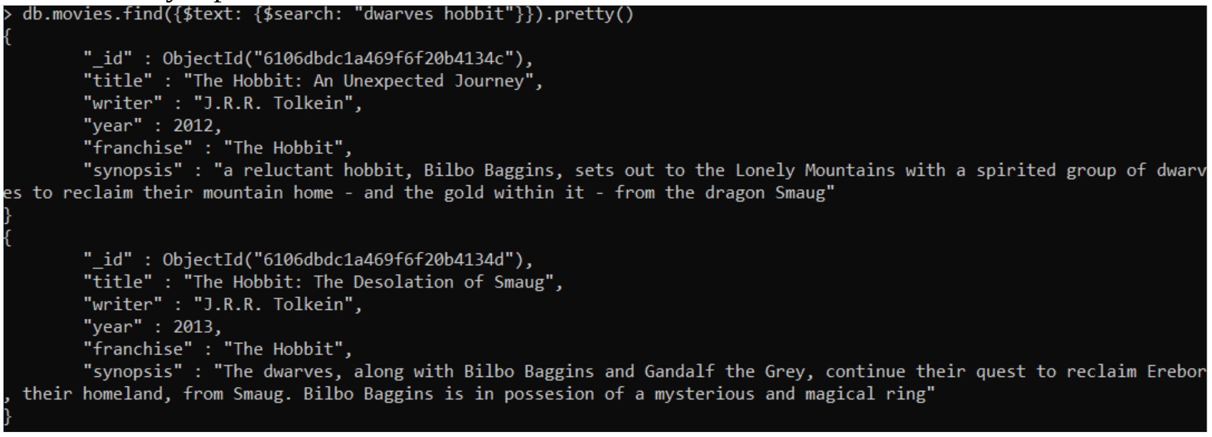
```
> db.movies.find({$text: {$search: "Gandalf"}}).pretty()
{
        "_id" : ObjectId("6106dbdc1a469f6f20b4134d"),
        "title" : "The Hobbit: The Desolation of Smaug",
        "writer" : "J.R.R. Tolkein",
        "year" : 2013,
        "franchise" : "The Hobbit",
        "synopsis" : "The dwarves, along with Bilbo Baggins and Gandalf the Grey, continue their quest to reclaim Erebor
, their homeland, from Smaug. Bilbo Baggins is in possesion of a mysterious and magical ring"
}
```

3) Synopsis with word Bilbo and not the word Gandalf



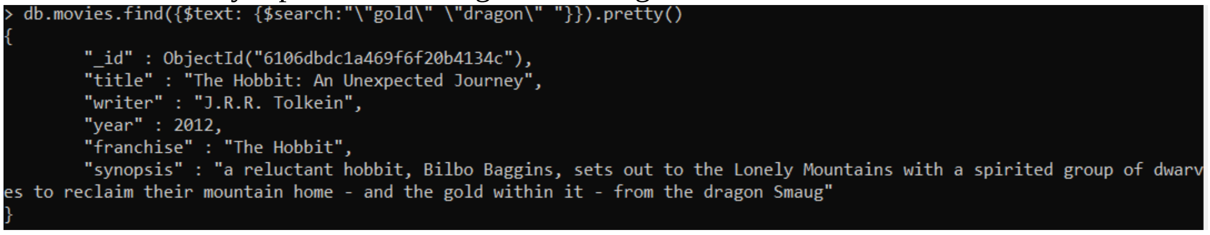
```
> db.movies.find({$text: {$search: "Bilbo -Gandalf"}}).pretty()
{
        "_id" : ObjectId("6106dbdc1a469f6f20b4134e"),
        "title" : "The Hobbit: The Battle of five Armies",
        "writer" : "J.R.R. Tolkein",
        "year" : 2012,
        "franchise" : "The Hobbit",
        "synopsis" : "Bilbo 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("6106dbdc1a469f6f20b4134c"),
        "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 Mountains with a spirited group of dwarv
es to reclaim their mountain home - and the gold within it - from the dragon Smaug"
}
```

4) All movies synopsis contain dwarves or hobbit



```
> db.movies.find({$text: {$search: "dwarves hobbit"}}).pretty()
{
        "_id" : ObjectId("6106dbdc1a469f6f20b4134c"),
        "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 Mountains with a spirited group of dwarv
es to reclaim their mountain home - and the gold within it - from the dragon Smaug"
}
{
        "_id" : ObjectId("6106dbdc1a469f6f20b4134d"),
        "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 possesion of a mysterious and magical ring"
}
```

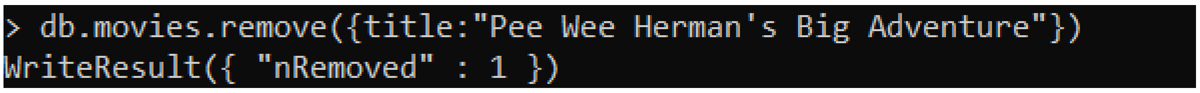
5) All movies with synopsis with word gold and dragon



```
> db.movies.find({$text: {$search:"\"gold\" \"dragon\" "}}).pretty()
{
        "_id" : ObjectId("6106dbdc1a469f6f20b4134c"),
        "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 Mountains with a spirited group of dwarv
es to reclaim their mountain home - and the gold within it - from the dragon Smaug"
}
```

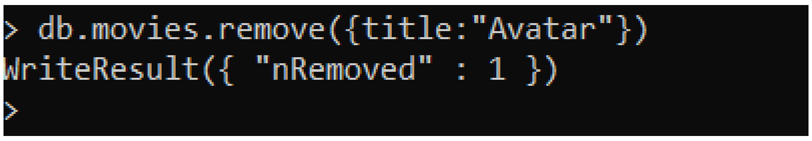
DELETE DOCUMENTS

1) Delete movie “Pee Wee …..”



```
> db.movies.remove({title:"Pee Wee Herman's Big Adventure"})
WriteResult({ "nRemoved" : 1 })
```

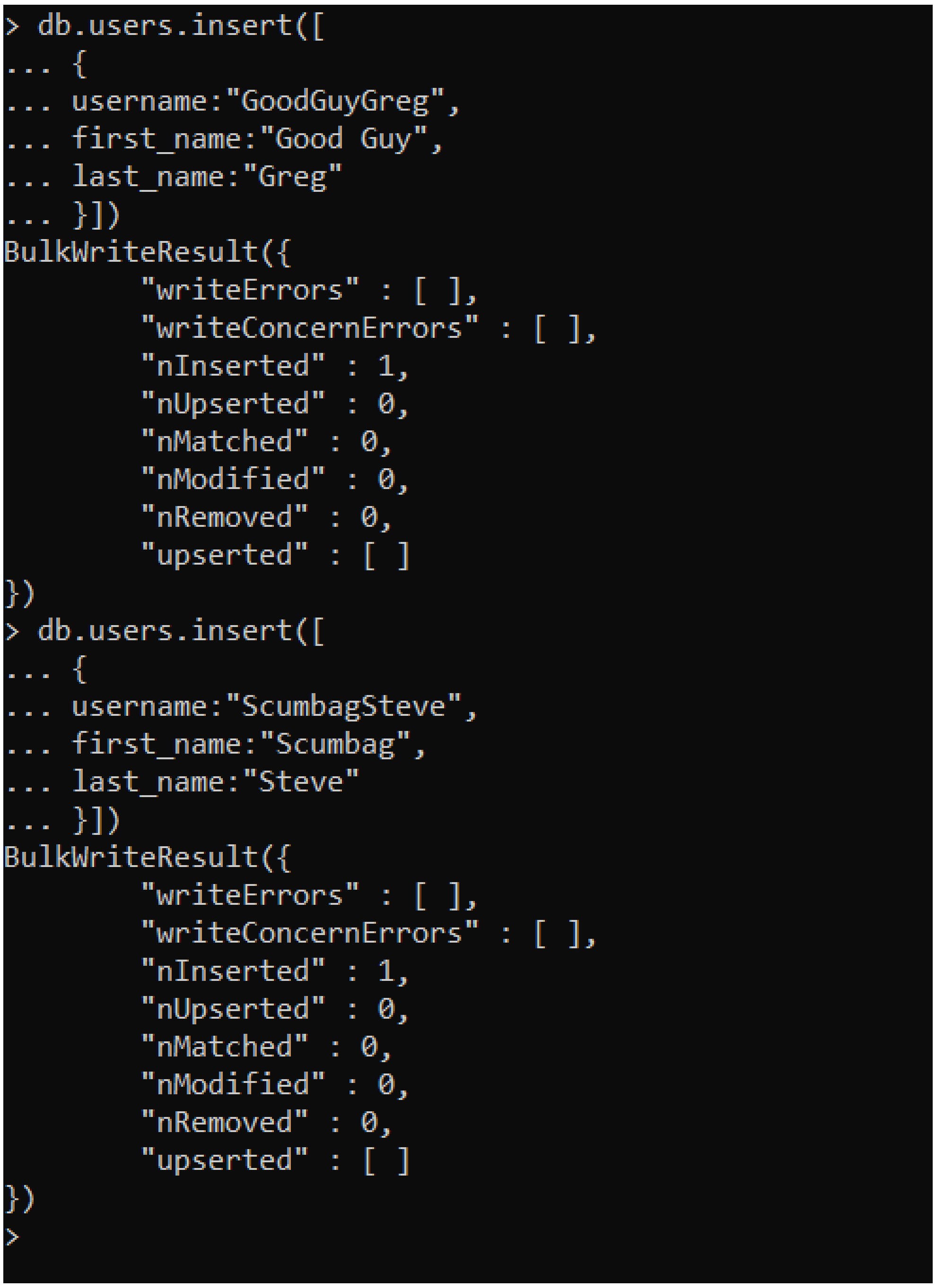
2) Delete movie “Avatar”



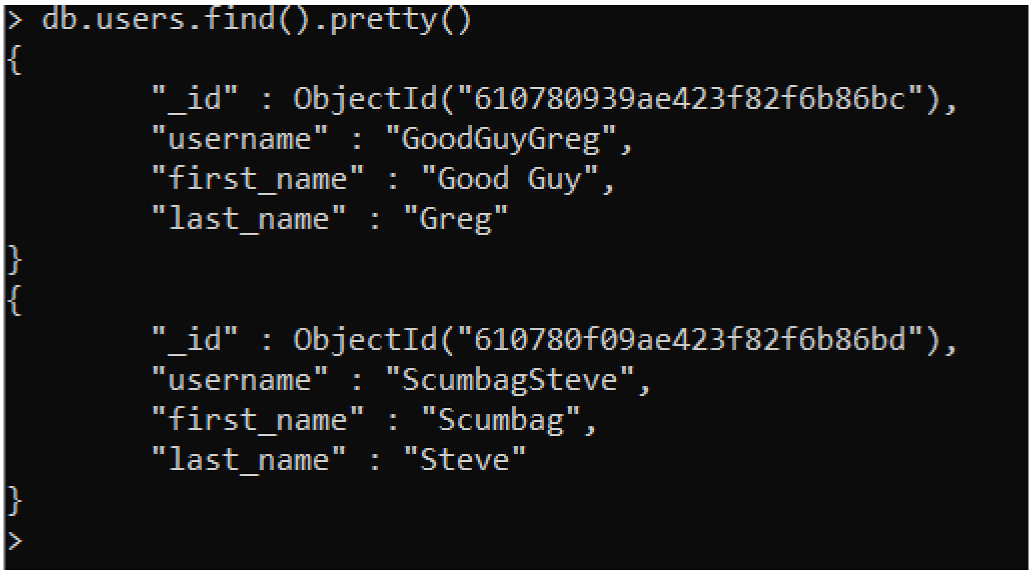
```
> db.movies.remove({title:"Avatar"})
WriteResult({ "nRemoved" : 1 })
>
```

Relationships

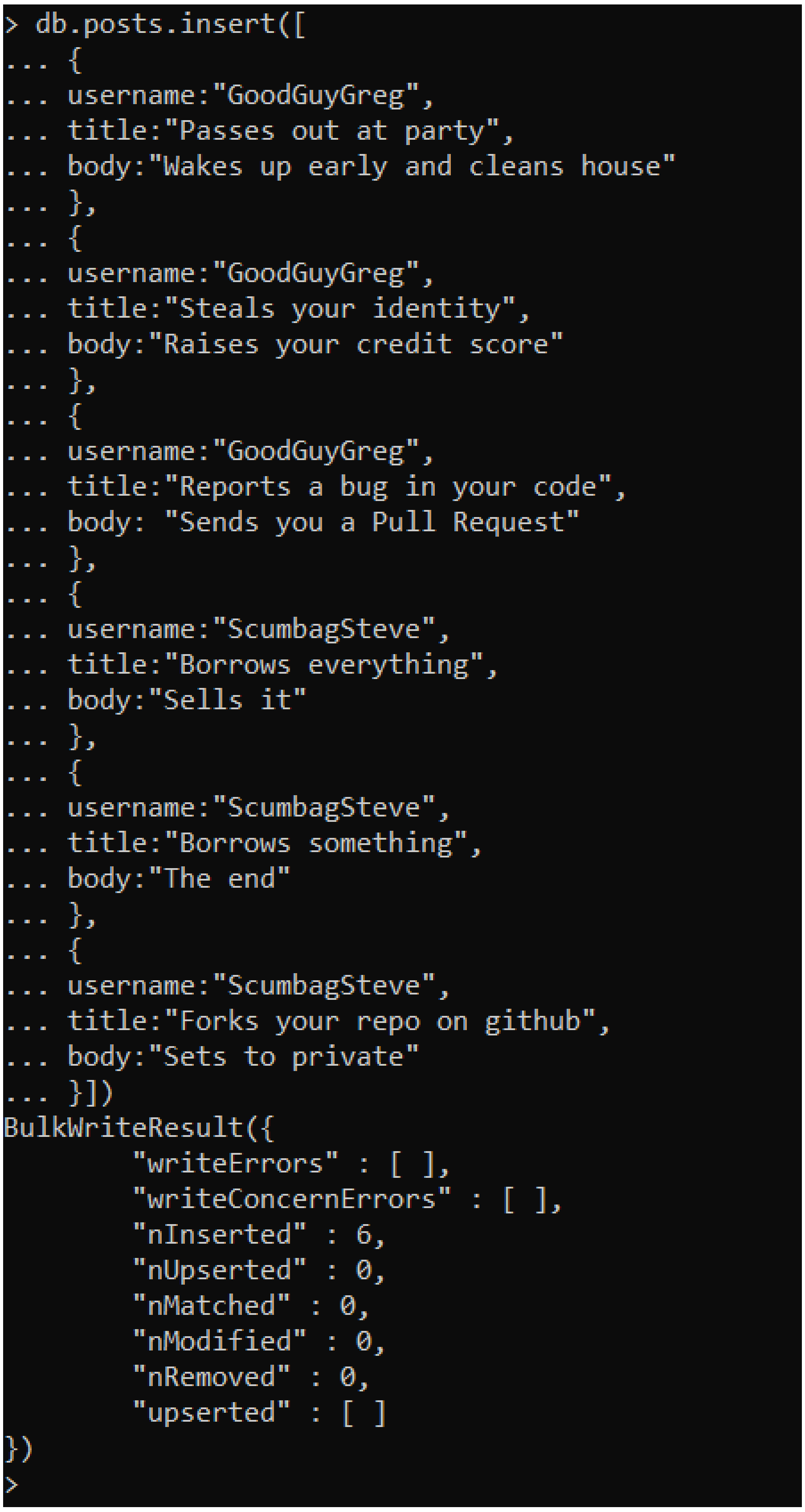
Insert following into a users collection ….



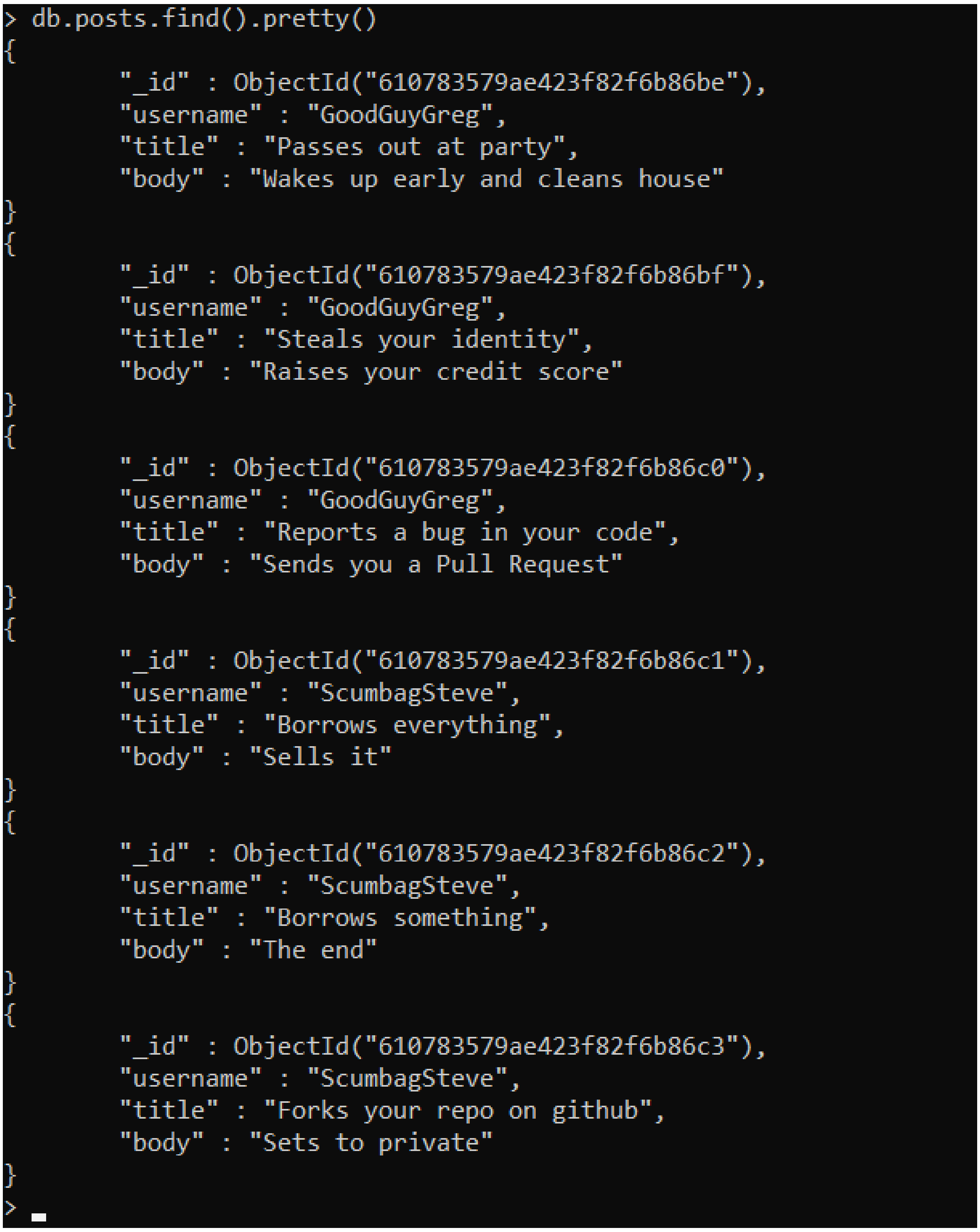
```
> db.users.insert([
... {
... username:"GoodGuyGreg",
... first_name:"Good Guy",
... last_name:"Greg"
... }])
BulkWriteResult({
        "writeErrors" : [ ],
        "writeConcernErrors" : [ ],
        "nInserted" : 1,
        "nUpserted" : 0,
        "nMatched" : 0,
        "nModified" : 0,
        "nRemoved" : 0,
        "upserted" : [ ]
})
> db.users.insert([
... {
... username:"ScumbagSteve",
... first_name:"Scumbag",
... last_name:"Steve"
... }])
BulkWriteResult({
        "writeErrors" : [ ],
        "writeConcernErrors" : [ ],
        "nInserted" : 1,
        "nUpserted" : 0,
        "nMatched" : 0,
        "nModified" : 0,
        "nRemoved" : 0,
        "upserted" : [ ]
})
>
```


```
> db.users.find().pretty()
{
        "_id" : ObjectId("610780939ae423f82f6b86bc"),
        "username" : "GoodGuyGreg",
        "first_name" : "Good Guy",
        "last_name" : "Greg"
}
{
        "_id" : ObjectId("610780f09ae423f82f6b86bd"),
        "username" : "ScumbagSteve",
        "first_name" : "Scumbag",
        "last_name" : "Steve"
}
>
```

Insert the following documents into a posts collection

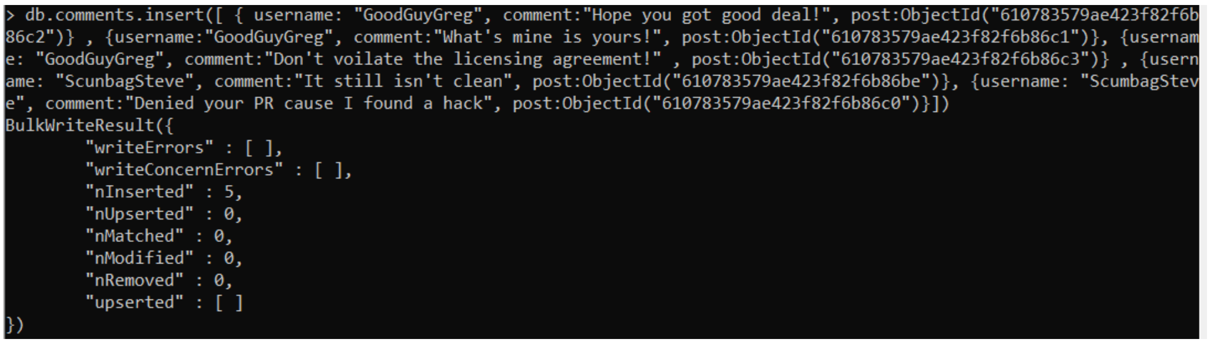


```
> db.posts.insert([
... {
... username:"GoodGuyGreg",
... title:"Passes out at party",
... body:"Wakes up early and cleans house"
... },
... {
... username:"GoodGuyGreg",
... title:"Steals your identity",
... body:"Raises your credit score"
... },
... {
... username:"GoodGuyGreg",
... title:"Reports a bug in your code",
... body: "Sends you a Pull Request"
... },
... {
... username:"ScumbagSteve",
... title:"Borrows everything",
... body:"Sells it"
... },
... {
... username:"ScumbagSteve",
... title:"Borrows something",
... body:"The end"
... },
... {
... username:"ScumbagSteve",
... title:"Forks your repo on github",
... body:"Sets to private"
... }])
BulkWriteResult({
        "writeErrors" : [ ],
        "writeConcernErrors" : [ ],
        "nInserted" : 6,
        "nUpserted" : 0,
        "nMatched" : 0,
        "nModified" : 0,
        "nRemoved" : 0,
        "upserted" : [ ]
})
>
```

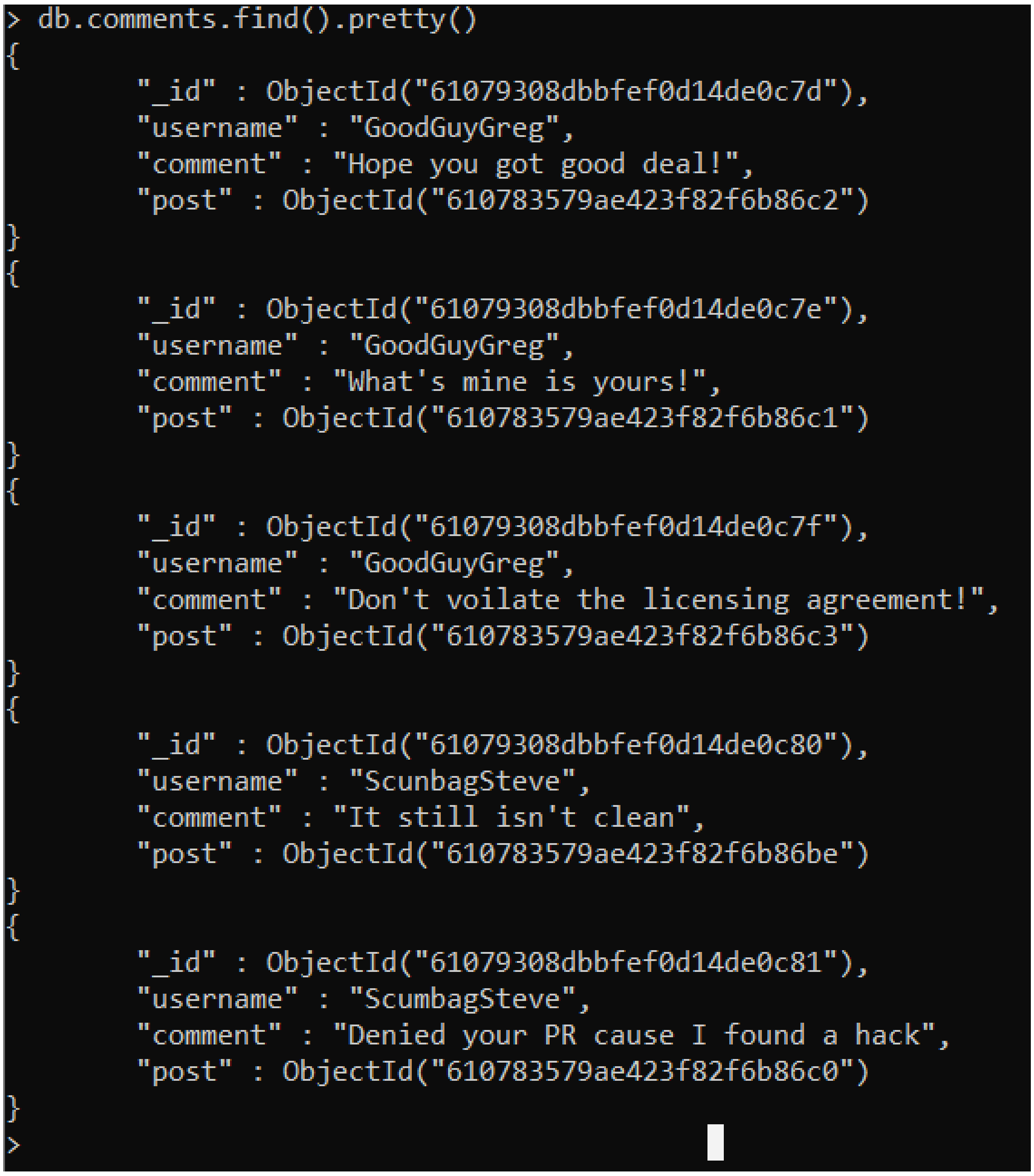


```
> db.posts.find().pretty()
{
        "_id" : ObjectId("610783579ae423f82f6b86be"),
        "username" : "GoodGuyGreg",
        "title" : "Passes out at party",
        "body" : "Wakes up early and cleans house"
}
{
        "_id" : ObjectId("610783579ae423f82f6b86bf"),
        "username" : "GoodGuyGreg",
        "title" : "Steals your identity",
        "body" : "Raises your credit score"
}
{
        "_id" : ObjectId("610783579ae423f82f6b86c0"),
        "username" : "GoodGuyGreg",
        "title" : "Reports a bug in your code",
        "body" : "Sends you a Pull Request"
}
{
        "_id" : ObjectId("610783579ae423f82f6b86c1"),
        "username" : "ScumbagSteve",
        "title" : "Borrows everything",
        "body" : "Sells it"
}
{
        "_id" : ObjectId("610783579ae423f82f6b86c2"),
        "username" : "ScumbagSteve",
        "title" : "Borrows something",
        "body" : "The end"
}
{
        "_id" : ObjectId("610783579ae423f82f6b86c3"),
        "username" : "ScumbagSteve",
        "title" : "Forks your repo on github",
        "body" : "Sets to private"
}
> 
```

Insert following documents into a comments collection



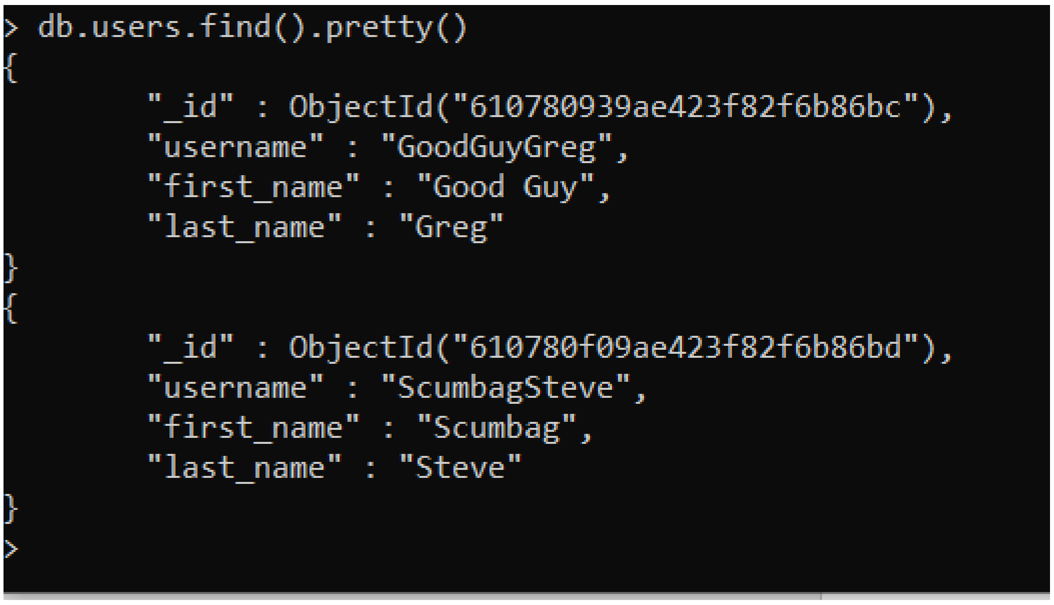
```
> db.comments.insert([ { username: "GoodGuyGreg", comment:"Hope you got good deal!", post:ObjectId("610783579ae423f82f6b
86c2")} , {username:"GoodGuyGreg", comment:"What's mine is yours!", post:ObjectId("610783579ae423f82f6b86c1")}, {usernam
e: "GoodGuyGreg", comment:"Don't voilate the licensing agreement!" , post:ObjectId("610783579ae423f82f6b86c3")} , {usern
ame: "ScunbagSteve", comment:"It still isn't clean", post:ObjectId("610783579ae423f82f6b86be")}, {username: "ScumbagStev
e", comment:"Denied your PR cause I found a hack", post:ObjectId("610783579ae423f82f6b86c0")}])
BulkWriteResult({
        "writeErrors" : [ ],
        "writeConcernErrors" : [ ],
        "nInserted" : 5,
        "nUpserted" : 0,
        "nMatched" : 0,
        "nModified" : 0,
        "nRemoved" : 0,
        "upserted" : [ ]
})
```



```
> db.comments.find().pretty()
{
        "_id" : ObjectId("61079308dbbfef0d14de0c7d"),
        "username" : "GoodGuyGreg",
        "comment" : "Hope you got good deal!",
        "post" : ObjectId("610783579ae423f82f6b86c2")
}
{
        "_id" : ObjectId("61079308dbbfef0d14de0c7e"),
        "username" : "GoodGuyGreg",
        "comment" : "What's mine is yours!",
        "post" : ObjectId("610783579ae423f82f6b86c1")
}
{
        "_id" : ObjectId("61079308dbbfef0d14de0c7f"),
        "username" : "GoodGuyGreg",
        "comment" : "Don't voilate the licensing agreement!",
        "post" : ObjectId("610783579ae423f82f6b86c3")
}
{
        "_id" : ObjectId("61079308dbbfef0d14de0c80"),
        "username" : "ScunbagSteve",
        "comment" : "It still isn't clean",
        "post" : ObjectId("610783579ae423f82f6b86be")
}
{
        "_id" : ObjectId("61079308dbbfef0d14de0c81"),
        "username" : "ScumbagSteve",
        "comment" : "Denied your PR cause I found a hack",
        "post" : ObjectId("610783579ae423f82f6b86c0")
}
>
```

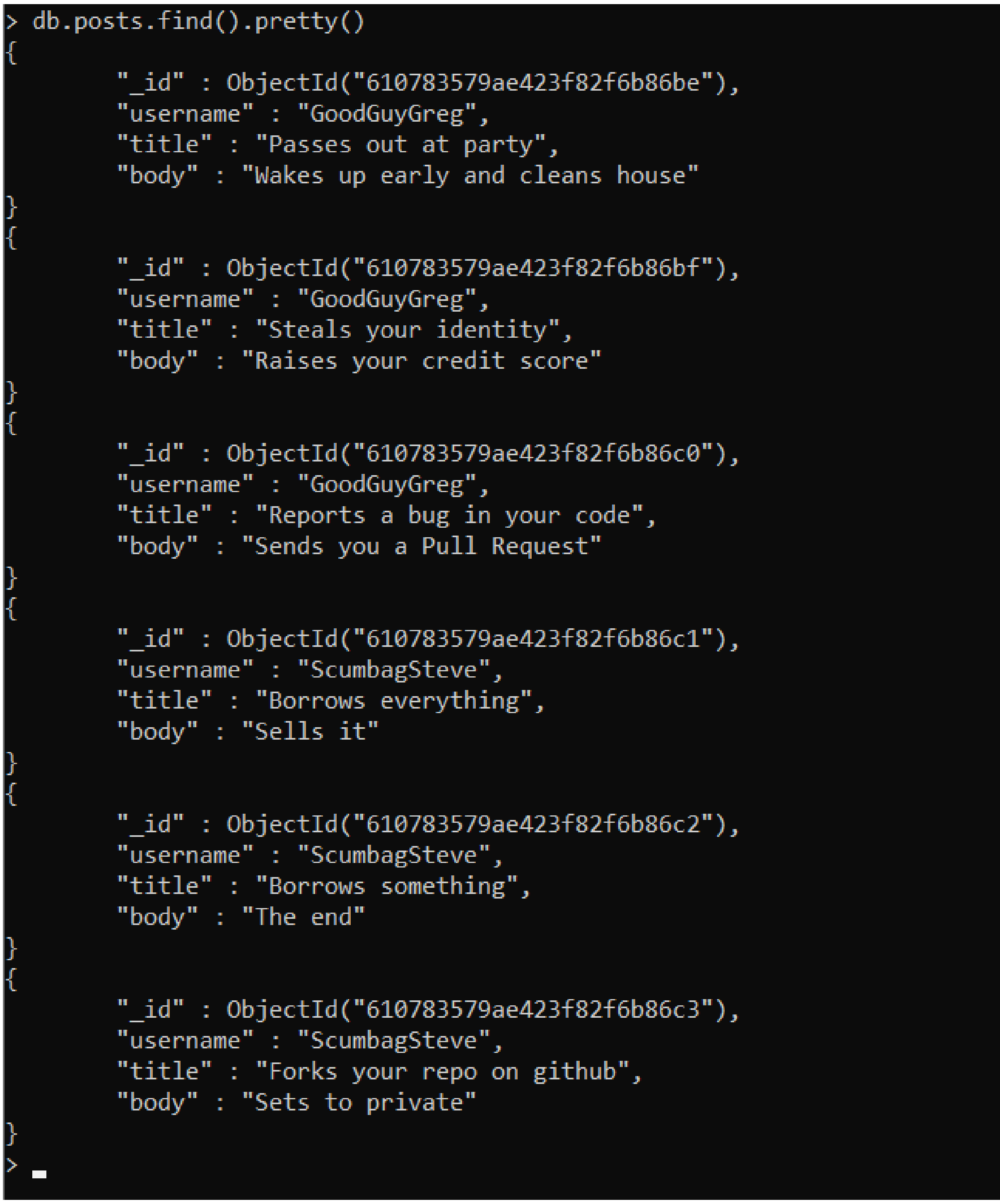
Query Related collections

1) Find all users



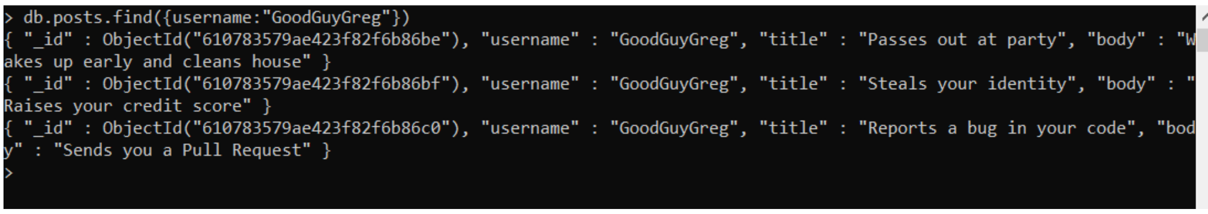
```
> db.users.find().pretty()
{
        "_id" : ObjectId("610780939ae423f82f6b86bc"),
        "username" : "GoodGuyGreg",
        "first_name" : "Good Guy",
        "last_name" : "Greg"
}
{
        "_id" : ObjectId("610780f09ae423f82f6b86bd"),
        "username" : "ScumbagSteve",
        "first_name" : "Scumbag",
        "last_name" : "Steve"
}
>
```

2) Find all posts



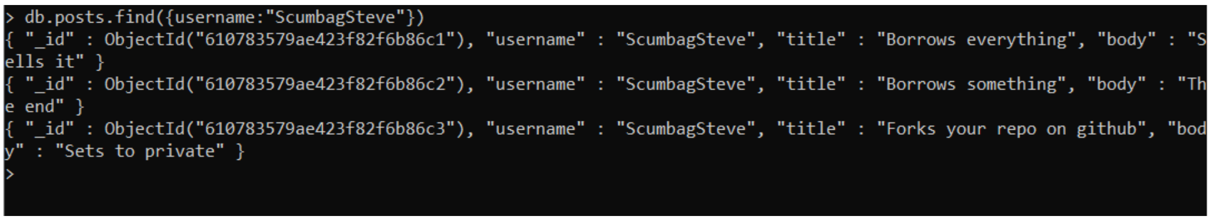
```
> db.posts.find().pretty()
{
        "_id" : ObjectId("610783579ae423f82f6b86be"),
        "username" : "GoodGuyGreg",
        "title" : "Passes out at party",
        "body" : "Wakes up early and cleans house"
}
{
        "_id" : ObjectId("610783579ae423f82f6b86bf"),
        "username" : "GoodGuyGreg",
        "title" : "Steals your identity",
        "body" : "Raises your credit score"
}
{
        "_id" : ObjectId("610783579ae423f82f6b86c0"),
        "username" : "GoodGuyGreg",
        "title" : "Reports a bug in your code",
        "body" : "Sends you a Pull Request"
}
{
        "_id" : ObjectId("610783579ae423f82f6b86c1"),
        "username" : "ScumbagSteve",
        "title" : "Borrows everything",
        "body" : "Sells it"
}
{
        "_id" : ObjectId("610783579ae423f82f6b86c2"),
        "username" : "ScumbagSteve",
        "title" : "Borrows something",
        "body" : "The end"
}
{
        "_id" : ObjectId("610783579ae423f82f6b86c3"),
        "username" : "ScumbagSteve",
        "title" : "Forks your repo on github",
        "body" : "Sets to private"
}
> 
```

3) All posts authorised by “GoodGuyGreg”



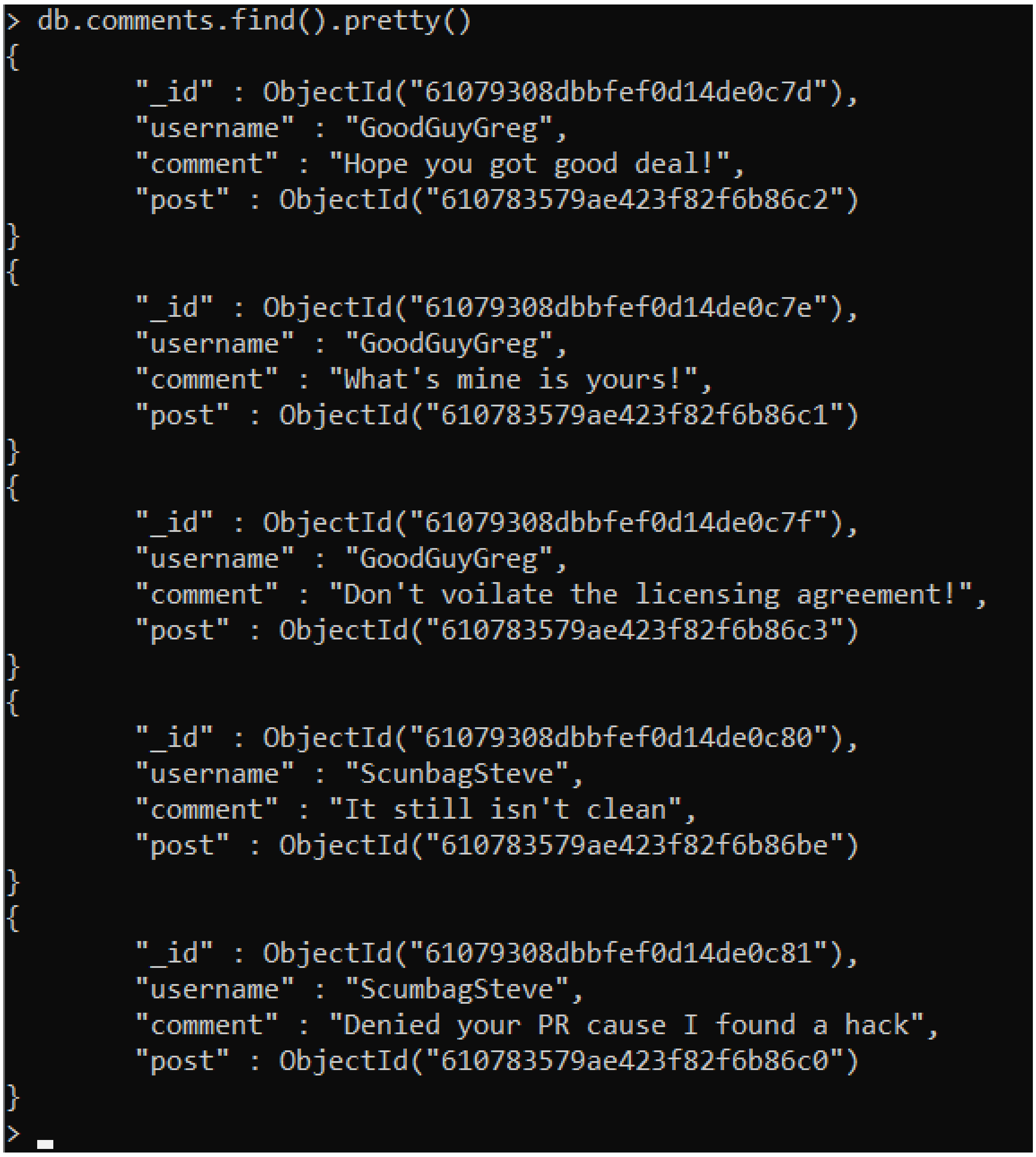
```
> db.posts.find({username:"GoodGuyGreg"})
{ "_id" : ObjectId("610783579ae423f82f6b86be"), "username" : "GoodGuyGreg", "title" : "Passes out at party", "body" : "W
akes up early and cleans house" }
{ "_id" : ObjectId("610783579ae423f82f6b86bf"), "username" : "GoodGuyGreg", "title" : "Steals your identity", "body" : "
Raises your credit score" }
{ "_id" : ObjectId("610783579ae423f82f6b86c0"), "username" : "GoodGuyGreg", "title" : "Reports a bug in your code", "bod
y" : "Sends you a Pull Request" }
>
```

4) All posts authorised by “ScumbagSteve”



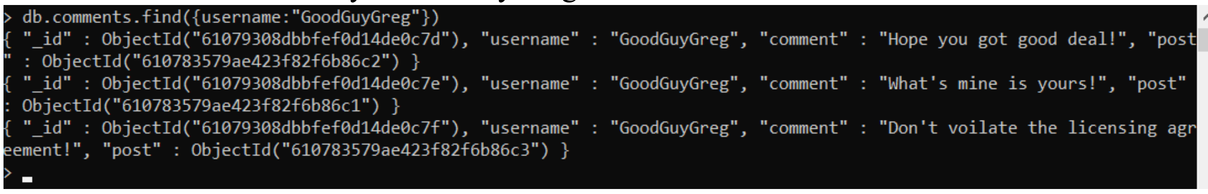
```
> db.posts.find({username:"ScumbagSteve"})
{ "_id" : ObjectId("610783579ae423f82f6b86c1"), "username" : "ScumbagSteve", "title" : "Borrows everything", "body" : "S
ells it" }
{ "_id" : ObjectId("610783579ae423f82f6b86c2"), "username" : "ScumbagSteve", "title" : "Borrows something", "body" : "Th
e end" }
{ "_id" : ObjectId("610783579ae423f82f6b86c3"), "username" : "ScumbagSteve", "title" : "Forks your repo on github", "bod
y" : "Sets to private" }
>
```

5) Find all comments



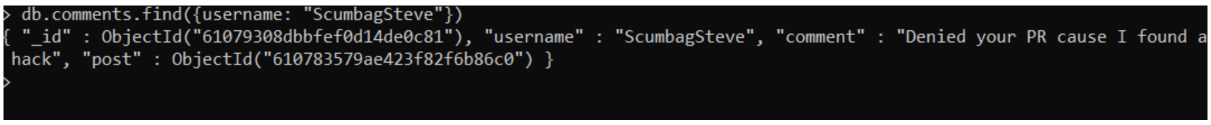
```
> db.comments.find().pretty()
{
        "_id" : ObjectId("61079308dbbfef0d14de0c7d"),
        "username" : "GoodGuyGreg",
        "comment" : "Hope you got good deal!",
        "post" : ObjectId("610783579ae423f82f6b86c2")
}
{
        "_id" : ObjectId("61079308dbbfef0d14de0c7e"),
        "username" : "GoodGuyGreg",
        "comment" : "What's mine is yours!",
        "post" : ObjectId("610783579ae423f82f6b86c1")
}
{
        "_id" : ObjectId("61079308dbbfef0d14de0c7f"),
        "username" : "GoodGuyGreg",
        "comment" : "Don't voilate the licensing agreement!",
        "post" : ObjectId("610783579ae423f82f6b86c3")
}
{
        "_id" : ObjectId("61079308dbbfef0d14de0c80"),
        "username" : "ScunbagSteve",
        "comment" : "It still isn't clean",
        "post" : ObjectId("610783579ae423f82f6b86be")
}
{
        "_id" : ObjectId("61079308dbbfef0d14de0c81"),
        "username" : "ScumbagSteve",
        "comment" : "Denied your PR cause I found a hack",
        "post" : ObjectId("610783579ae423f82f6b86c0")
}
>
```

6) All comments authorised by "GoodGuyGreg"



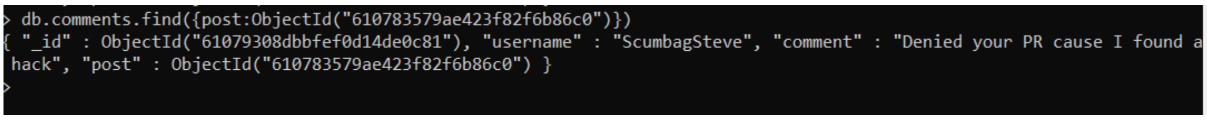
```
> db.comments.find({username:"GoodGuyGreg"})
{ "_id" : ObjectId("61079308dbbfef0d14de0c7d"), "username" : "GoodGuyGreg", "comment" : "Hope you got good deal!", "post
" : ObjectId("610783579ae423f82f6b86c2") }
{ "_id" : ObjectId("61079308dbbfef0d14de0c7e"), "username" : "GoodGuyGreg", "comment" : "What's mine is yours!", "post"
: ObjectId("610783579ae423f82f6b86c1") }
{ "_id" : ObjectId("61079308dbbfef0d14de0c7f"), "username" : "GoodGuyGreg", "comment" : "Don't voilate the licensing agr
eement!", "post" : ObjectId("610783579ae423f82f6b86c3") }
>
```

7) All comments authorised by “ScumbagSteve”



```
> db.comments.find({username: "ScumbagSteve"})
{ "_id" : ObjectId("61079308dbbfef0d14de0c81"), "username" : "ScumbagSteve", "comment" : "Denied your PR cause I found a
 hack", "post" : ObjectId("610783579ae423f82f6b86c0") }
>
```

8) All comments belonging to “Reports a bug in your code”



```
> db.comments.find({post:ObjectId("610783579ae423f82f6b86c0")})
{ "_id" : ObjectId("61079308dbbfef0d14de0c81"), "username" : "ScumbagSteve", "comment" : "Denied your PR cause I found a
 hack", "post" : ObjectId("610783579ae423f82f6b86c0") }
>
```