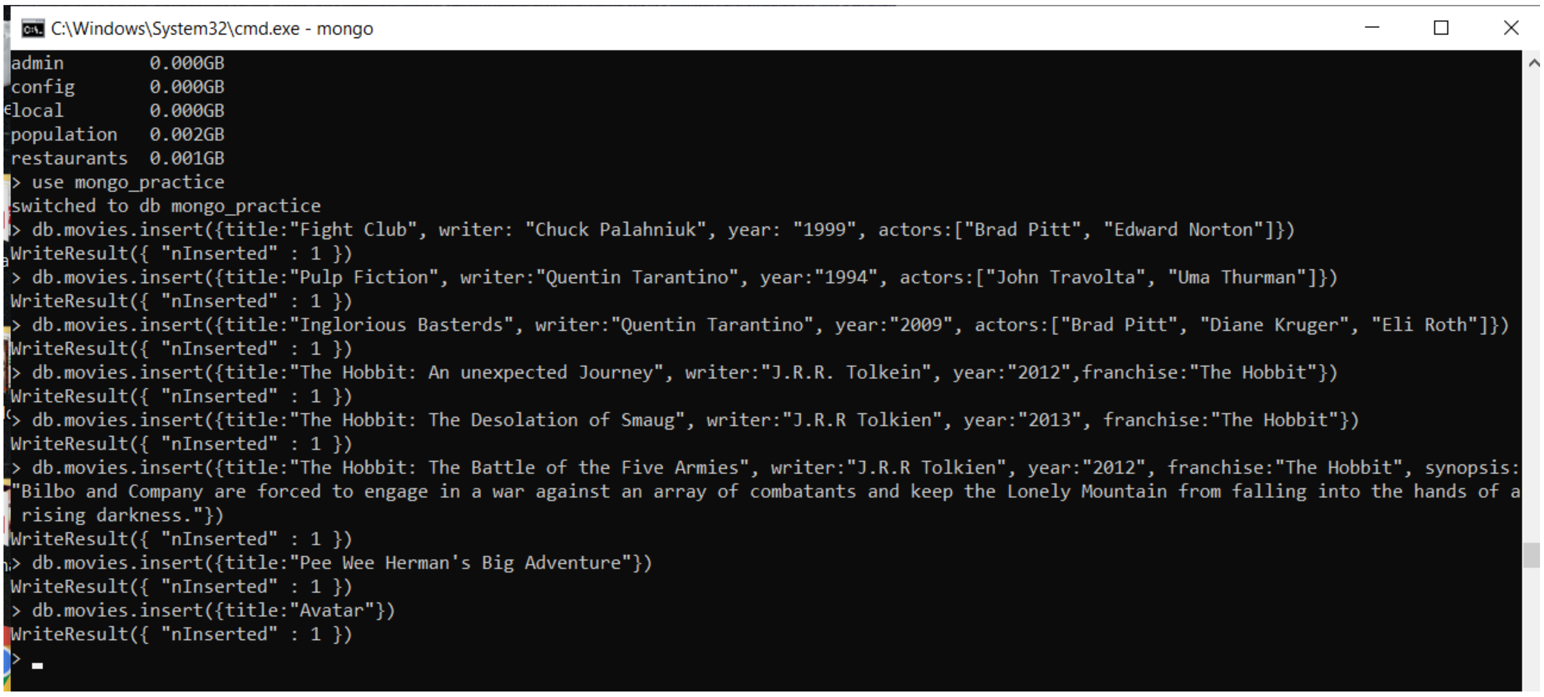
Assignment – 1

Insert Document

Queries used

- db.movies.insert({title:"Fight Club", writer: "Chuck Palahniuk", year: "1999", actors:["Brad Pitt", "Edward Norton"]})
- db.movies.insert({title:"Pulp Fiction", writer:"Quentin Tarantino", year:"1994", actors:["John Travolta", "Uma Thurman"]})
- db.movies.insert({title:"Inglorious Basterds", writer:"Quentin Tarantino", year:"2009", actors:["Brad Pitt", "Diane Kruger", "Eli Roth"]})
- db.movies.insert({title:"The Hobbit: An unexpected Journey", writer:"J.R.R. Tolkein", year:"2012",franchise:"The Hobbit"})
- db.movies.insert({title:"The Hobbit: The Desolation of Smaug", writer:"J.R.R Tolkien", year:"2013", franchise:"The Hobbit"})
- 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."})
- db.movies.insert({title:"Pee Wee Herman's Big Adventure"})
- db.movies.insert({title:"Avatar"})

Output



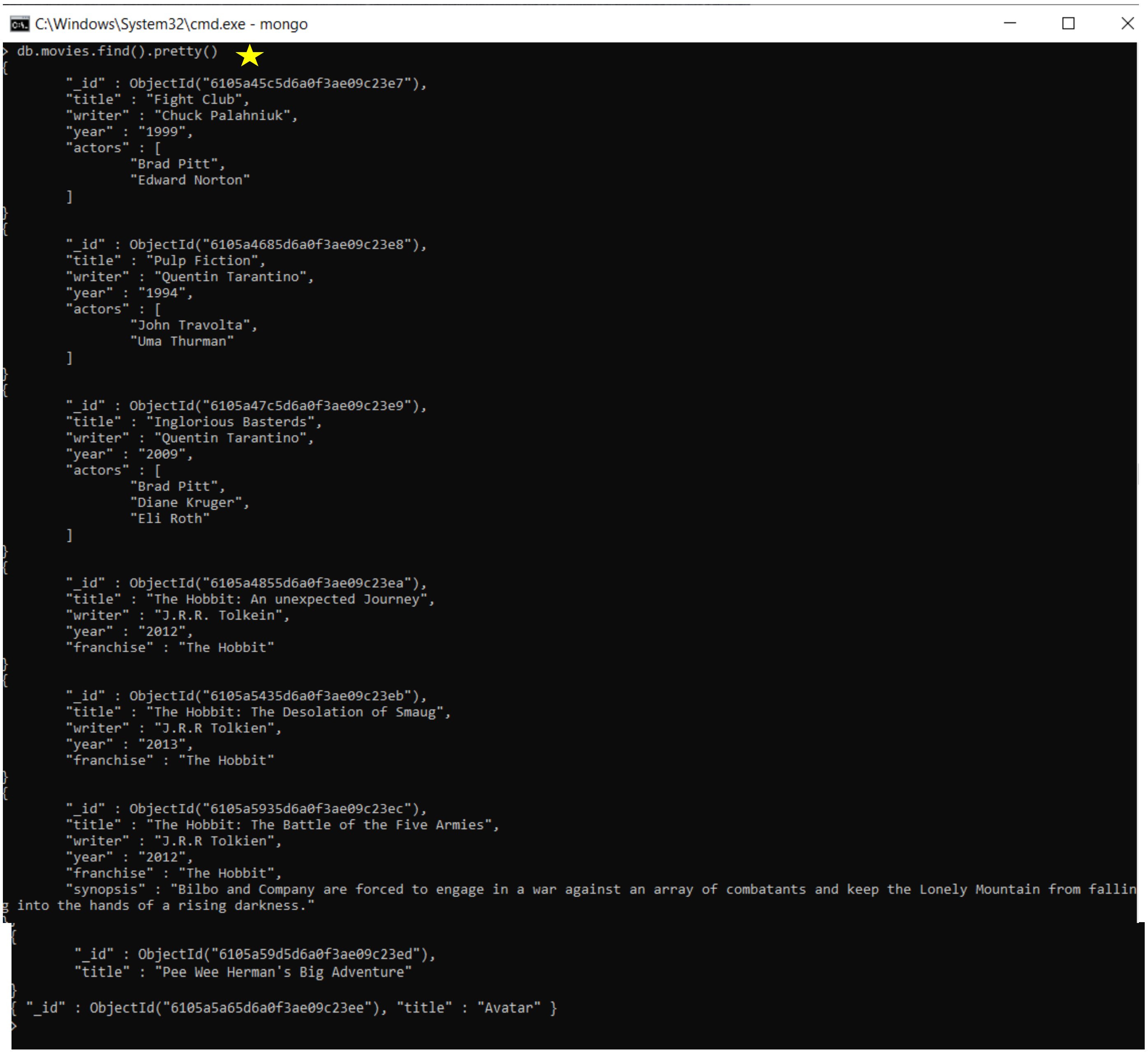
```
admin        0.000GB
config       0.000GB
local        0.000GB
population   0.002GB
restaurants  0.001GB
> use mongo_practice
switched to db mongo_practice
> db.movies.insert({title:"Fight Club", writer: "Chuck Palahniuk", year: "1999", actors:["Brad Pitt", "Edward Norton"]})
WriteResult({ "nInserted" : 1 })
> db.movies.insert({title:"Pulp Fiction", writer:"Quentin Tarantino", year:"1994", actors:["John Travolta", "Uma Thurman"]})
WriteResult({ "nInserted" : 1 })
> db.movies.insert({title:"Inglorious Basterds", writer:"Quentin Tarantino", year:"2009", actors:["Brad Pitt", "Diane Kruger", "Eli Roth"]})
WriteResult({ "nInserted" : 1 })
> db.movies.insert({title:"The Hobbit: An unexpected Journey", writer:"J.R.R. Tolkein", year:"2012",franchise:"The Hobbit"})
WriteResult({ "nInserted" : 1 })
> db.movies.insert({title:"The Hobbit: The Desolation of Smaug", writer:"J.R.R Tolkien", year:"2013", franchise:"The Hobbit"})
WriteResult({ "nInserted" : 1 })
> 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 })
> db.movies.insert({title:"Pee Wee Herman's Big Adventure"})
WriteResult({ "nInserted" : 1 })
> db.movies.insert({title:"Avatar"})
WriteResult({ "nInserted" : 1 })
>
```

Find Documents

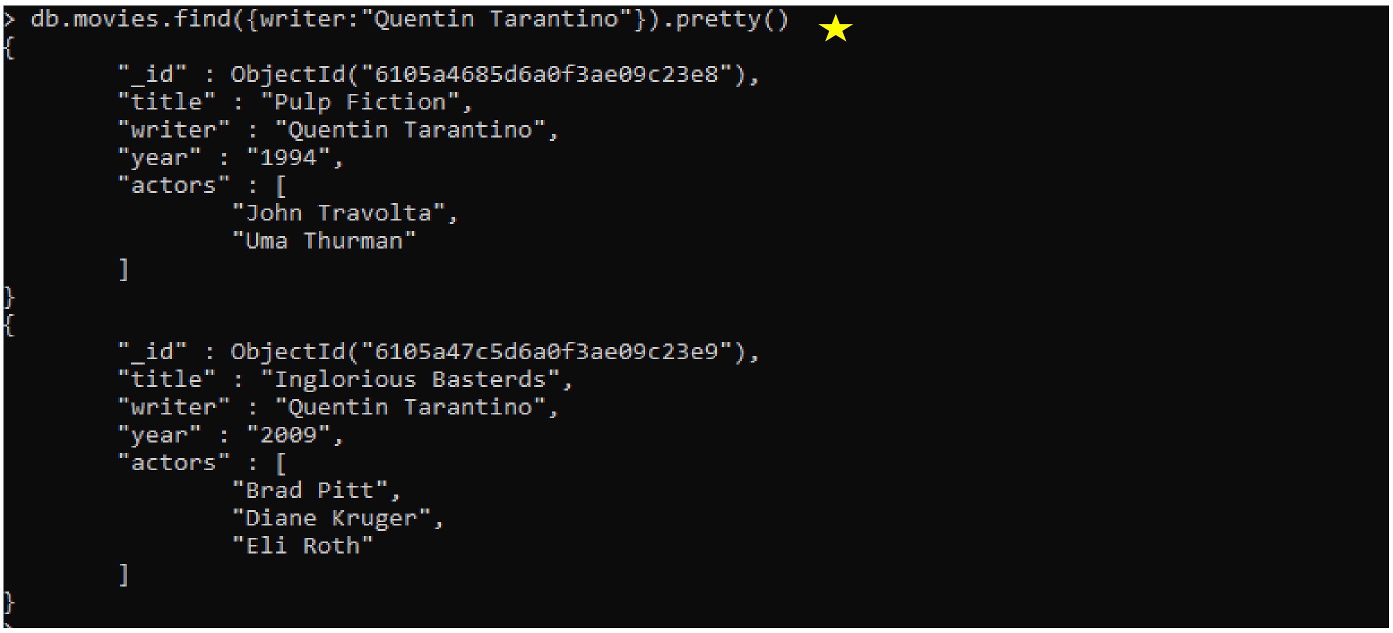
Queries used

1. db.movies.find().pretty()
2. db.movies.find({writer:"Quentin Tarantino"}).pretty()
3. db.movies.find({actors:"Brad Pitt"}).pretty()
4. db.movies.find({franchise:"The Hobbit"}).pretty()
5. db.movies.find({year:{$gt:"1990", $lt:"2000"}}).pretty()
6. db.movies.find({$or:[{year:{$gt:"2010"}},{year: {$lt:"2000"}}]}).pretty()

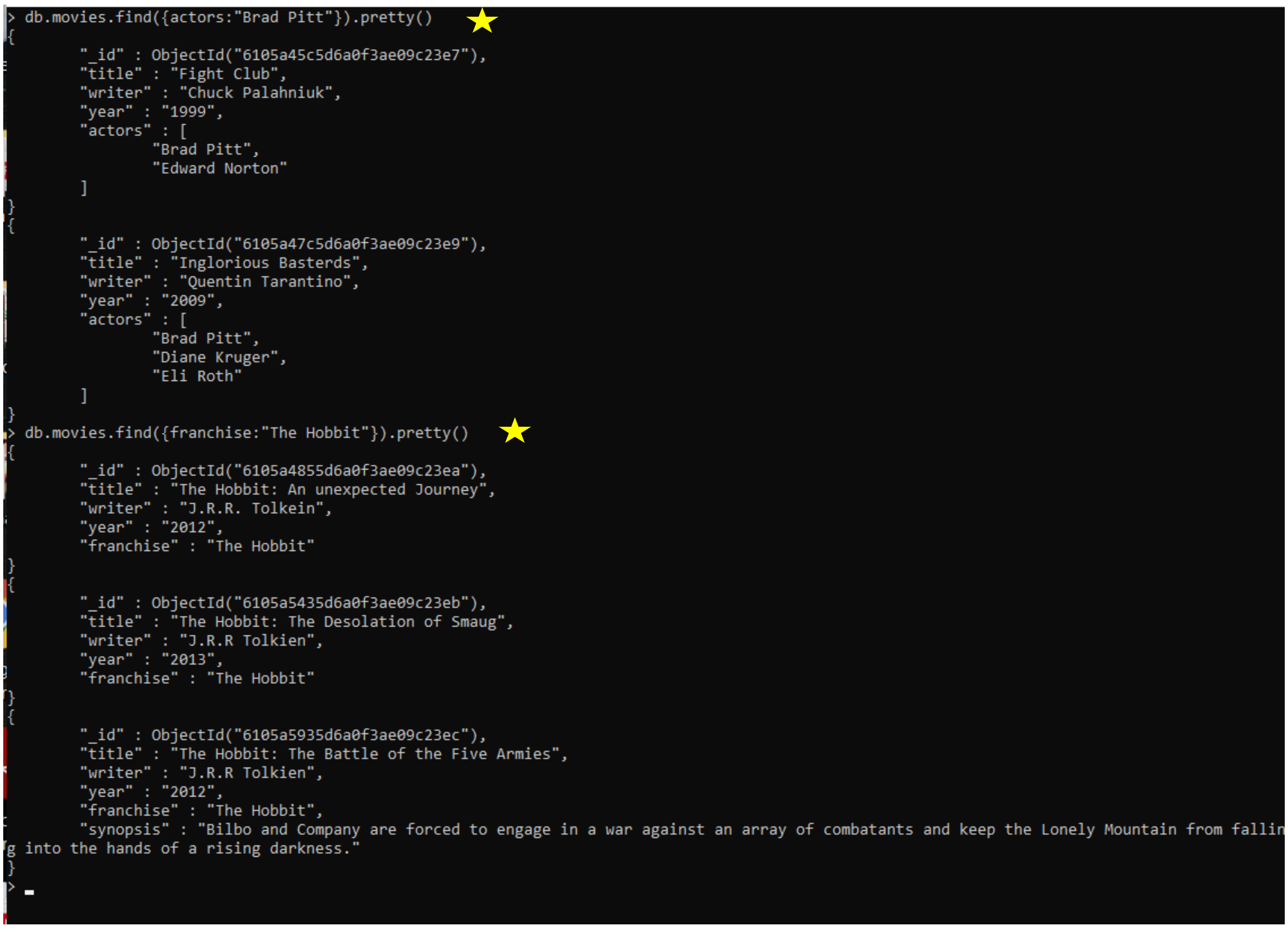
Output



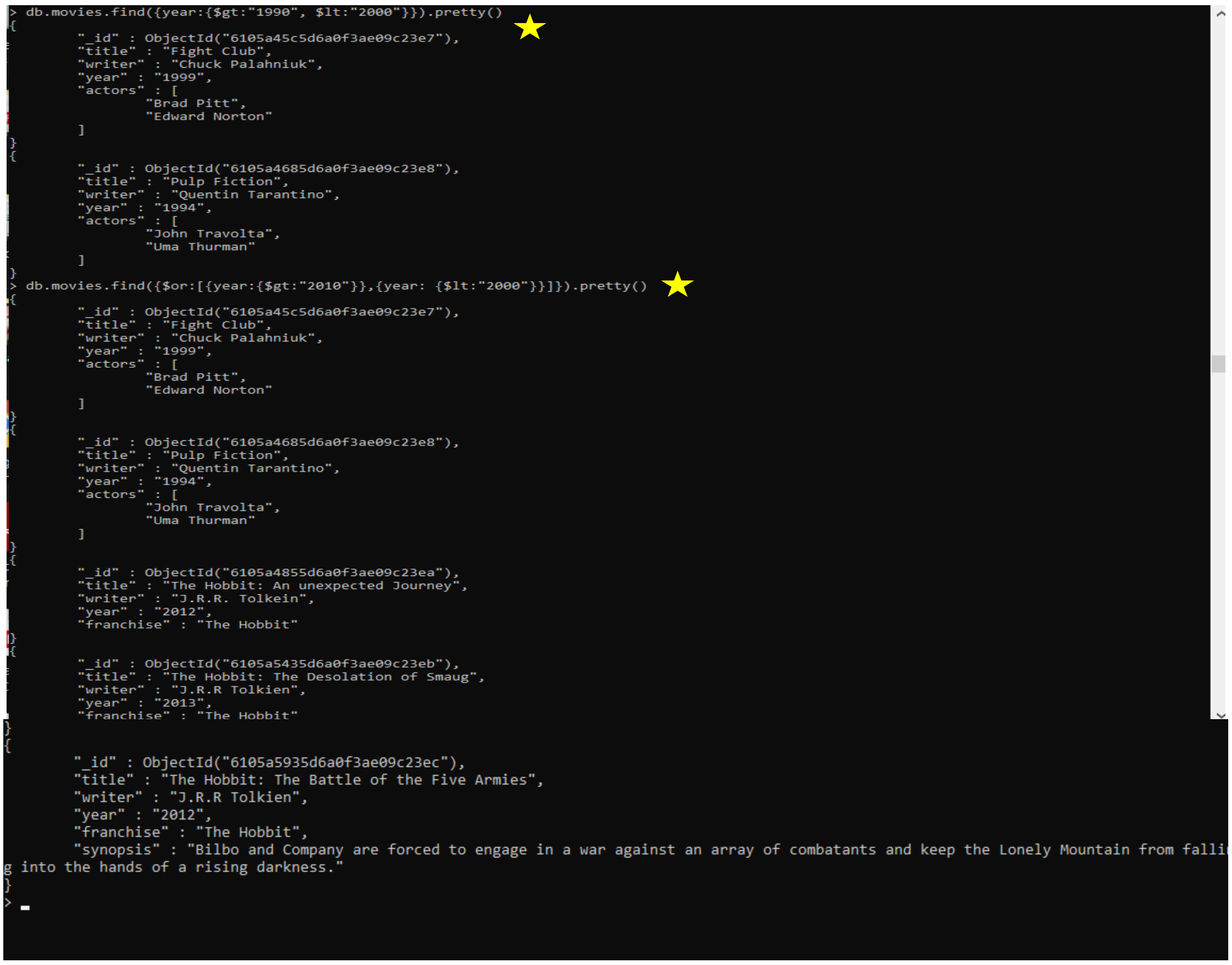
```
> db.movies.find().pretty()
{
        "_id" : ObjectId("6105a45c5d6a0f3ae09c23e7"),
        "title" : "Fight Club",
        "writer" : "Chuck Palahniuk",
        "year" : "1999",
        "actors" : [
                "Brad Pitt",
                "Edward Norton"
        ]
}
{
        "_id" : ObjectId("6105a4685d6a0f3ae09c23e8"),
        "title" : "Pulp Fiction",
        "writer" : "Quentin Tarantino",
        "year" : "1994",
        "actors" : [
                "John Travolta",
                "Uma Thurman"
        ]
}
{
        "_id" : ObjectId("6105a47c5d6a0f3ae09c23e9"),
        "title" : "Inglorious Basterds",
        "writer" : "Quentin Tarantino",
        "year" : "2009",
        "actors" : [
                "Brad Pitt",
                "Diane Kruger",
                "Eli Roth"
        ]
}
{
        "_id" : ObjectId("6105a4855d6a0f3ae09c23ea"),
        "title" : "The Hobbit: An unexpected Journey",
        "writer" : "J.R.R. Tolkein",
        "year" : "2012",
        "franchise" : "The Hobbit"
}
{
        "_id" : ObjectId("6105a5435d6a0f3ae09c23eb"),
        "title" : "The Hobbit: The Desolation of Smaug",
        "writer" : "J.R.R Tolkien",
        "year" : "2013",
        "franchise" : "The Hobbit"
}
{
        "_id" : ObjectId("6105a5935d6a0f3ae09c23ec"),
        "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 fallin
g into the hands of a rising darkness."
}
{
        "_id" : ObjectId("6105a59d5d6a0f3ae09c23ed"),
        "title" : "Pee Wee Herman's Big Adventure"
}
{ "_id" : ObjectId("6105a5a65d6a0f3ae09c23ee"), "title" : "Avatar" }
>
```



```
> db.movies.find({writer:"Quentin Tarantino"}).pretty()
{
        "_id" : ObjectId("6105a4685d6a0f3ae09c23e8"),
        "title" : "Pulp Fiction",
        "writer" : "Quentin Tarantino",
        "year" : "1994",
        "actors" : [
                "John Travolta",
                "Uma Thurman"
        ]
}
{
        "_id" : ObjectId("6105a47c5d6a0f3ae09c23e9"),
        "title" : "Inglorious Basterds",
        "writer" : "Quentin Tarantino",
        "year" : "2009",
        "actors" : [
                "Brad Pitt",
                "Diane Kruger",
                "Eli Roth"
        ]
}
```



```
> db.movies.find({actors:"Brad Pitt"}).pretty()
{
        "_id" : ObjectId("6105a45c5d6a0f3ae09c23e7"),
        "title" : "Fight Club",
        "writer" : "Chuck Palahniuk",
        "year" : "1999",
        "actors" : [
                "Brad Pitt",
                "Edward Norton"
        ]
}
{
        "_id" : ObjectId("6105a47c5d6a0f3ae09c23e9"),
        "title" : "Inglorious Basterds",
        "writer" : "Quentin Tarantino",
        "year" : "2009",
        "actors" : [
                "Brad Pitt",
                "Diane Kruger",
                "Eli Roth"
        ]
}
> db.movies.find({franchise:"The Hobbit"}).pretty()
{
        "_id" : ObjectId("6105a4855d6a0f3ae09c23ea"),
        "title" : "The Hobbit: An unexpected Journey",
        "writer" : "J.R.R. Tolkein",
        "year" : "2012",
        "franchise" : "The Hobbit"
}
{
        "_id" : ObjectId("6105a5435d6a0f3ae09c23eb"),
        "title" : "The Hobbit: The Desolation of Smaug",
        "writer" : "J.R.R Tolkien",
        "year" : "2013",
        "franchise" : "The Hobbit"
}
{
        "_id" : ObjectId("6105a5935d6a0f3ae09c23ec"),
        "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 fallin
g into the hands of a rising darkness."
}
> 
```



```
> db.movies.find({year:{$gt:"1990", $lt:"2000"}}).pretty()
{
        "_id" : ObjectId("6105a45c5d6a0f3ae09c23e7"),
        "title" : "Fight Club",
        "writer" : "Chuck Palahniuk",
        "year" : "1999",
        "actors" : [
                "Brad Pitt",
                "Edward Norton"
        ]
}
{
        "_id" : ObjectId("6105a4685d6a0f3ae09c23e8"),
        "title" : "Pulp Fiction",
        "writer" : "Quentin Tarantino",
        "year" : "1994",
        "actors" : [
                "John Travolta",
                "Uma Thurman"
        ]
}
> db.movies.find({$or:[{year:{$gt:"2010"}},{year: {$lt:"2000"}}]}).pretty()
{
        "_id" : ObjectId("6105a45c5d6a0f3ae09c23e7"),
        "title" : "Fight Club",
        "writer" : "Chuck Palahniuk",
        "year" : "1999",
        "actors" : [
                "Brad Pitt",
                "Edward Norton"
        ]
}
{
        "_id" : ObjectId("6105a4685d6a0f3ae09c23e8"),
        "title" : "Pulp Fiction",
        "writer" : "Quentin Tarantino",
        "year" : "1994",
        "actors" : [
                "John Travolta",
                "Uma Thurman"
        ]
}
{
        "_id" : ObjectId("6105a4855d6a0f3ae09c23ea"),
        "title" : "The Hobbit: An unexpected Journey",
        "writer" : "J.R.R. Tolkein",
        "year" : "2012",
        "franchise" : "The Hobbit"
}
{
        "_id" : ObjectId("6105a5435d6a0f3ae09c23eb"),
        "title" : "The Hobbit: The Desolation of Smaug",
        "writer" : "J.R.R Tolkien",
        "year" : "2013",
        "franchise" : "The Hobbit"
}
{
        "_id" : ObjectId("6105a5935d6a0f3ae09c23ec"),
        "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 falli
g into the hands of a rising darkness."
}
> _
```

Update Documents

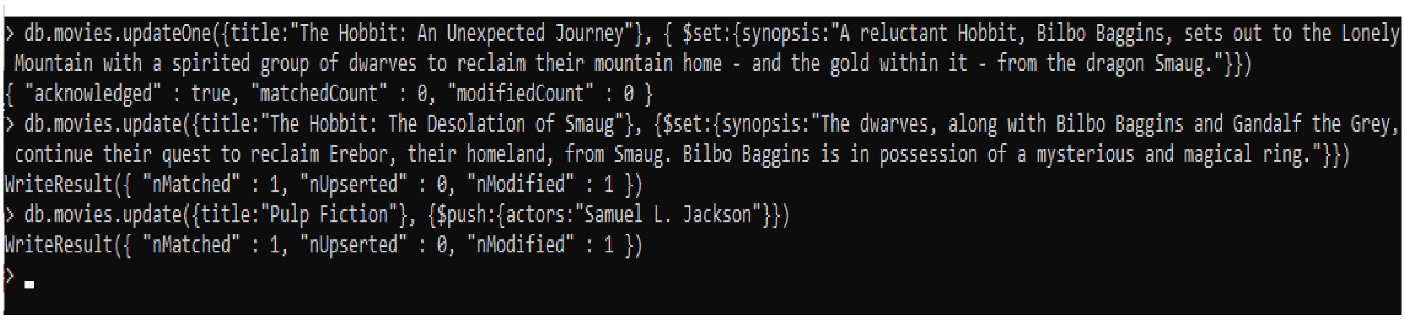
Queries Used

1. db.movies.updateOne({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."}})

2. 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."}})

3. db.movies.update({title:"Pulp Fiction"}, {$push:{actors:"Samuel L. Jackson"}})

Output



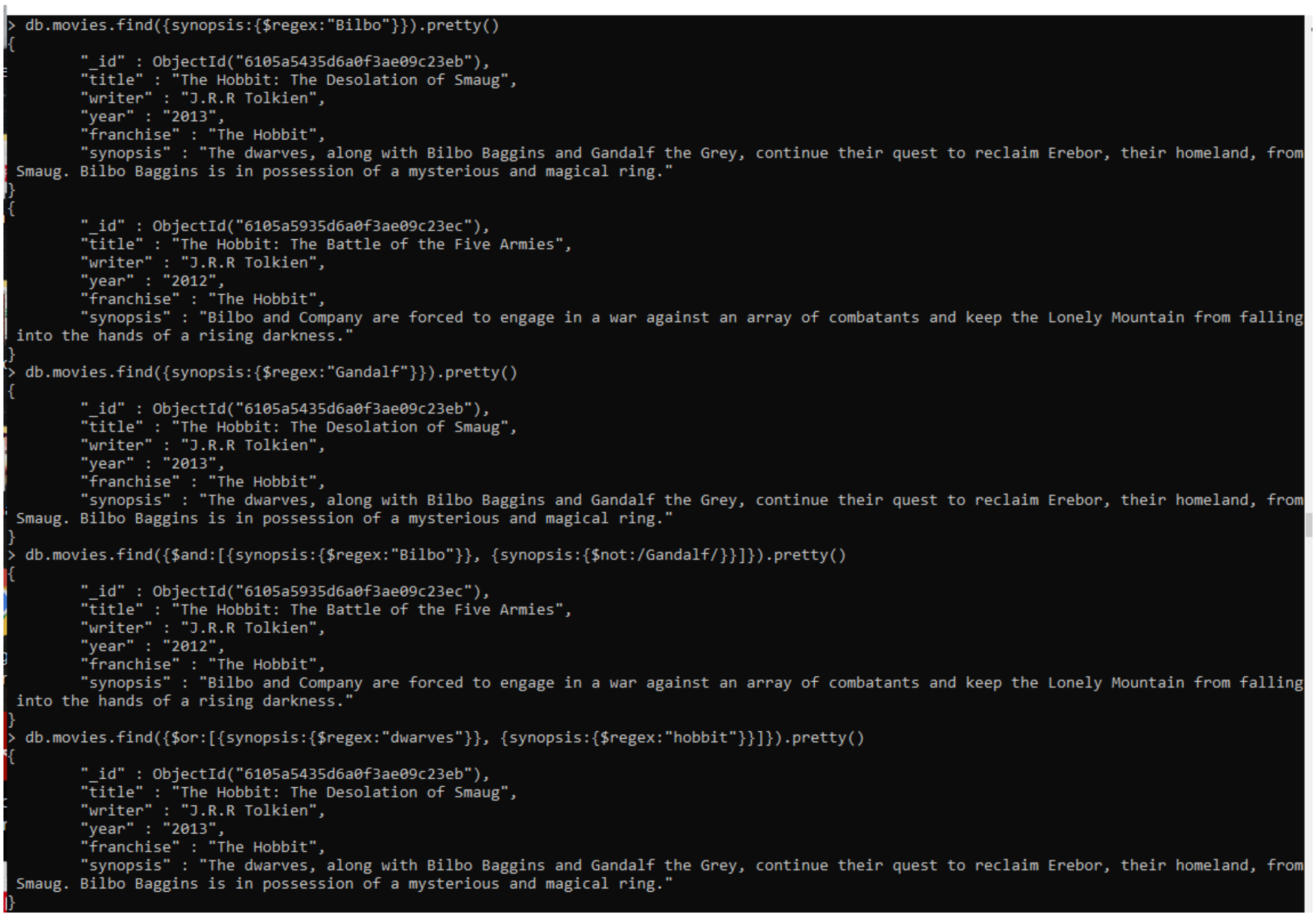
```
> db.movies.updateOne({title:"The Hobbit: An Unexpected Journey"}, { $set:{synopsis:"A reluctant Hobbit, Bilbo Baggins, sets out to the Lonely
 Mountain with a spirited group of dwarves to reclaim their mountain home - and the gold within it - from the dragon Smaug."}})
{ "acknowledged" : true, "matchedCount" : 0, "modifiedCount" : 0 }
> db.movies.update({title:"The Hobbit: The Desolation of Smaug"}, {$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."}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.movies.update({title:"Pulp Fiction"}, {$push:{actors:"Samuel L. Jackson"}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> _
```

Text Search

Queries Used

db.movies.find({synopsis:{$regex:"Bilbo"}}).pretty()
db.movies.find({synopsis:{$regex:"Gandalf"}}).pretty()
db.movies.find({$and:[{synopsis:{$regex:"Bilbo"}}, {synopsis:{$not:/Gandalf/}}]}).pretty()
db.movies.find({$or:[{synopsis:{$regex:"dwarves"}}, {synopsis:{$regex:"hobbit"}}]}).pretty()
db.movies.find({$and:[{synopsis:{$regex:"gold"}}, {synopsis:{$regex:"dragon"}}]}).pretty()

Results

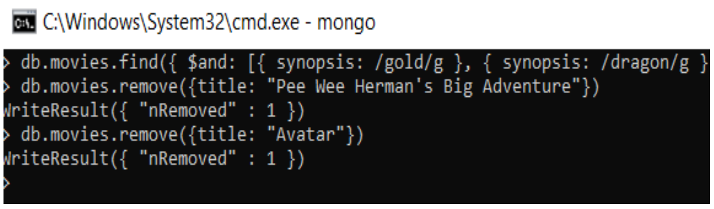


```
> db.movies.find({synopsis:{$regex:"Bilbo"}}).pretty()
{
        "_id" : ObjectId("6105a5435d6a0f3ae09c23eb"),
        "title" : "The Hobbit: The Desolation of Smaug",
        "writer" : "J.R.R Tolkien",
        "year" : "2013",
        "franchise" : "The Hobbit",
        "synopsis" : "The dwarves, along with Bilbo Baggins and Gandalf the Grey, continue their quest to reclaim Erebor, their homeland, from
 Smaug. Bilbo Baggins is in possession of a mysterious and magical ring."
}
{
        "_id" : ObjectId("6105a5935d6a0f3ae09c23ec"),
        "title" : "The Hobbit: The Battle of the Five Armies",
        "writer" : "J.R.R Tolkien",
        "year" : "2012",
        "franchise" : "The Hobbit",
        "synopsis" : "Bilbo and Company are forced to engage in a war against an array of combatants and keep the Lonely Mountain from falling
 into the hands of a rising darkness."
}
> db.movies.find({synopsis:{$regex:"Gandalf"}}).pretty()
{
        "_id" : ObjectId("6105a5435d6a0f3ae09c23eb"),
        "title" : "The Hobbit: The Desolation of Smaug",
        "writer" : "J.R.R Tolkien",
        "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."
}
> db.movies.find({$and:[{synopsis:{$regex:"Bilbo"}}, {synopsis:{$not:/Gandalf/}}]}).pretty()
{
        "_id" : ObjectId("6105a5935d6a0f3ae09c23ec"),
        "title" : "The Hobbit: The Battle of the Five Armies",
        "writer" : "J.R.R Tolkien",
        "year" : "2012",
        "franchise" : "The Hobbit",
        "synopsis" : "Bilbo and Company are forced to engage in a war against an array of combatants and keep the Lonely Mountain from falling
 into the hands of a rising darkness."
}
> db.movies.find({$or:[{synopsis:{$regex:"dwarves"}}, {synopsis:{$regex:"hobbit"}}]}).pretty()
{
        "_id" : ObjectId("6105a5435d6a0f3ae09c23eb"),
        "title" : "The Hobbit: The Desolation of Smaug",
        "writer" : "J.R.R Tolkien",
        "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."
}
```

Delete Documents

Queries Used

1. db.movies.remove({title: "Pee Wee Herman's Big Adventure"})
2. db.movies.remove({title: "Avatar"})



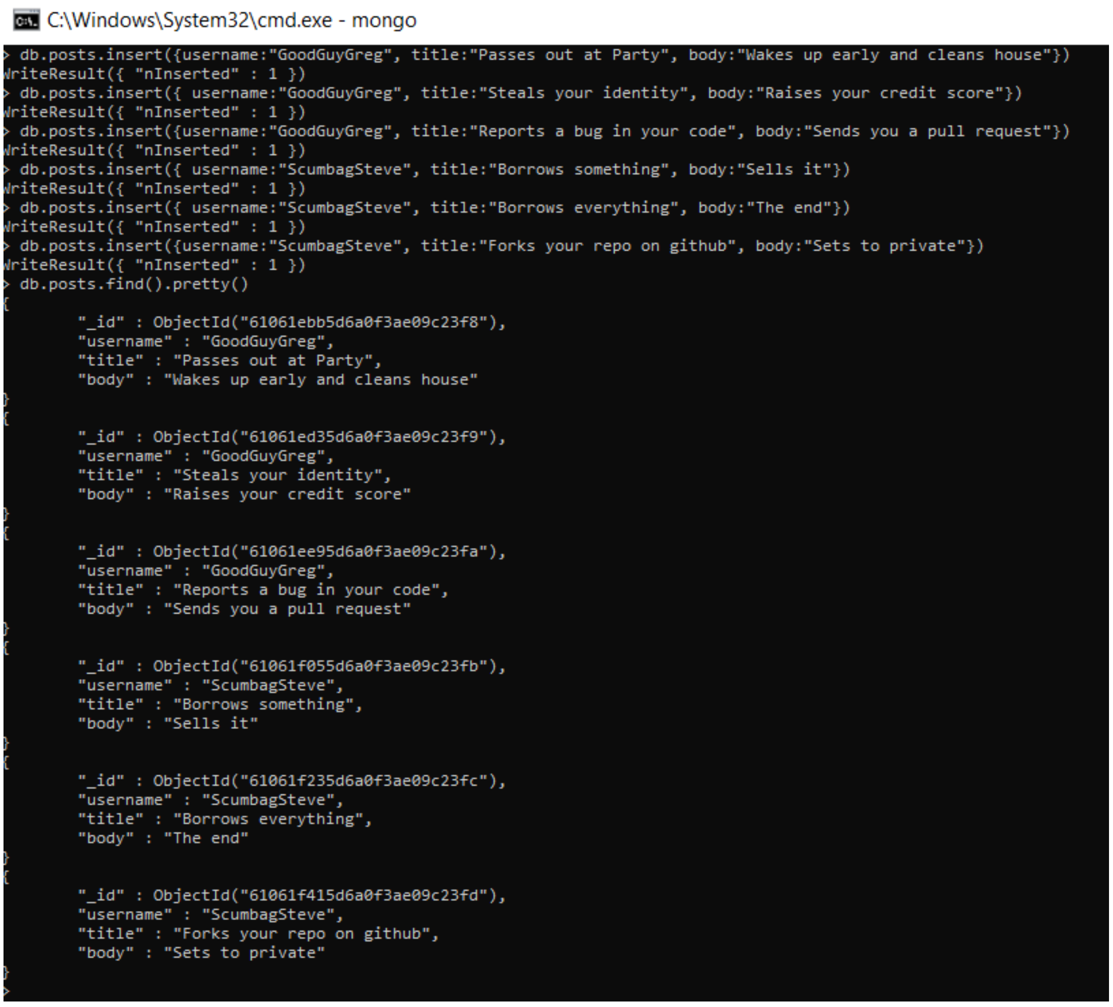
```
> db.movies.find({ $and: [{ synopsis: /gold/g }, { synopsis: /dragon/g }
> db.movies.remove({title: "Pee Wee Herman's Big Adventure"})
WriteResult({ "nRemoved" : 1 })
> db.movies.remove({title: "Avatar"})
WriteResult({ "nRemoved" : 1 })
>
```

Relationships

Queries Used

1. db.users.insert({_id:1,username:"GoodGuyGreg", first_name:"Good Guy", last_name:"Greg"})
2. db.users.insert({_id:2, username:"ScumbagSteve", fullname:{first: "Scumbag", last:"Steve"}})
3. db.posts.insert({username:"GoodGuyGreg", title:"Passes out at Party", body:"Wakes up early and cleans house"})
4. db.posts.insert({ username:"GoodGuyGreg", title:"Steals your identity", body:"Raises your credit score"})
5. db.posts.insert({username:"GoodGuyGreg", title:"Reports a bug in your code", body:"Sends you a pull request"})
6. db.posts.insert({ username:"ScumbagSteve", title:"Borrows something", body:"Sells it"})
7. db.posts.insert({ username:"ScumbagSteve", title:"Borrows everything", body:"The end"})
8. db.posts.insert({username:"ScumbagSteve", title:"Forks your repo on github", body:"Sets to private"})

Results

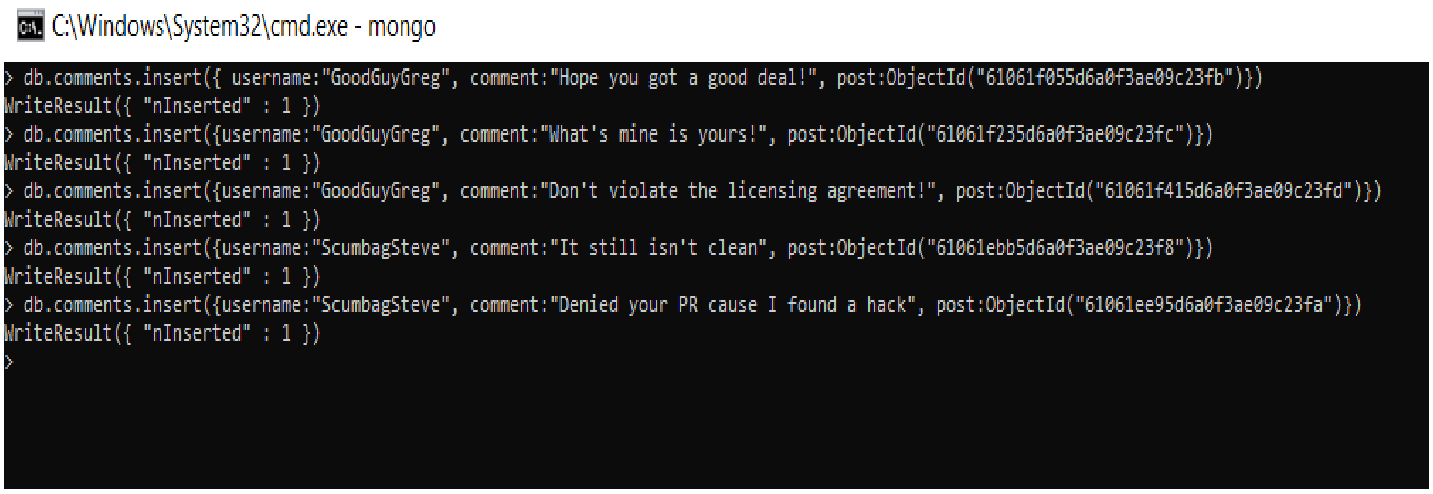


C:\Windows\System32\cmd.exe - mongo

```
> db.posts.insert({username:"GoodGuyGreg", title:"Passes out at Party", body:"Wakes up early and cleans house"})
WriteResult({ "nInserted" : 1 })
> db.posts.insert({ username:"GoodGuyGreg", title:"Steals your identity", body:"Raises your credit score"})
WriteResult({ "nInserted" : 1 })
> db.posts.insert({username:"GoodGuyGreg", title:"Reports a bug in your code", body:"Sends you a pull request"})
WriteResult({ "nInserted" : 1 })
> db.posts.insert({ username:"ScumbagSteve", title:"Borrows something", body:"Sells it"})
WriteResult({ "nInserted" : 1 })
> db.posts.insert({ username:"ScumbagSteve", title:"Borrows everything", body:"The end"})
WriteResult({ "nInserted" : 1 })
> db.posts.insert({username:"ScumbagSteve", title:"Forks your repo on github", body:"Sets to private"})
WriteResult({ "nInserted" : 1 })
> db.posts.find().pretty()
{
        "_id" : ObjectId("61061ebb5d6a0f3ae09c23f8"),
        "username" : "GoodGuyGreg",
        "title" : "Passes out at Party",
        "body" : "Wakes up early and cleans house"
}
{
        "_id" : ObjectId("61061ed35d6a0f3ae09c23f9"),
        "username" : "GoodGuyGreg",
        "title" : "Steals your identity",
        "body" : "Raises your credit score"
}
{
        "_id" : ObjectId("61061ee95d6a0f3ae09c23fa"),
        "username" : "GoodGuyGreg",
        "title" : "Reports a bug in your code",
        "body" : "Sends you a pull request"
}
{
        "_id" : ObjectId("61061f055d6a0f3ae09c23fb"),
        "username" : "ScumbagSteve",
        "title" : "Borrows something",
        "body" : "Sells it"
}
{
        "_id" : ObjectId("61061f235d6a0f3ae09c23fc"),
        "username" : "ScumbagSteve",
        "title" : "Borrows everything",
        "body" : "The end"
}
{
        "_id" : ObjectId("61061f415d6a0f3ae09c23fd"),
        "username" : "ScumbagSteve",
        "title" : "Forks your repo on github",
        "body" : "Sets to private"
}
>
```

1. db.comments.insert({ username:"GoodGuyGreg", comment:"Hope you got a good deal!", post:ObjectId("61061f055d6a0f3ae09c23fb")})
2. db.comments.insert({username:"GoodGuyGreg", comment:"What's mine is yours!", post:ObjectId("61061f235d6a0f3ae09c23fc")})
3. db.comments.insert({username:"GoodGuyGreg", comment:"Don't violate the licensing agreement!", post:ObjectId("61061f415d6a0f3ae09c23fd")})
4. db.comments.insert({username:"ScumbagSteve", comment:"It still isn't clean", post:ObjectId("61061ebb5d6a0f3ae09c23f8")})
5. db.comments.insert({username:"ScumbagSteve", comment:"Denied your PR cause I found a hack", post:ObjectId("61061ee95d6a0f3ae09c23fa")})

Results

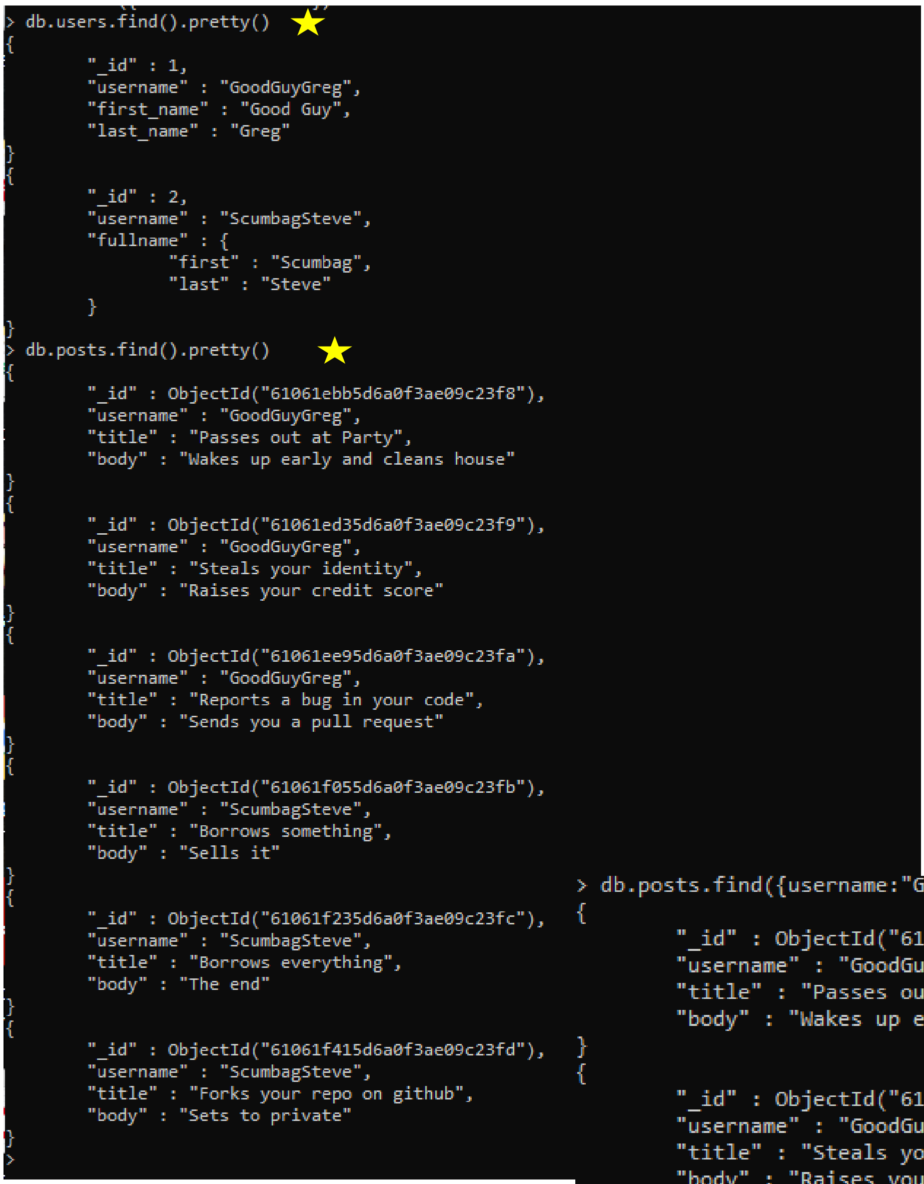


```
> db.comments.insert({ username:"GoodGuyGreg", comment:"Hope you got a good deal!", post:ObjectId("61061f055d6a0f3ae09c23fb")})
WriteResult({ "nInserted" : 1 })
> db.comments.insert({username:"GoodGuyGreg", comment:"What's mine is yours!", post:ObjectId("61061f235d6a0f3ae09c23fc")})
WriteResult({ "nInserted" : 1 })
> db.comments.insert({username:"GoodGuyGreg", comment:"Don't violate the licensing agreement!", post:ObjectId("61061f415d6a0f3ae09c23fd")})
WriteResult({ "nInserted" : 1 })
> db.comments.insert({username:"ScumbagSteve", comment:"It still isn't clean", post:ObjectId("61061ebb5d6a0f3ae09c23f8")})
WriteResult({ "nInserted" : 1 })
> db.comments.insert({username:"ScumbagSteve", comment:"Denied your PR cause I found a hack", post:ObjectId("61061ee95d6a0f3ae09c23fa")})
WriteResult({ "nInserted" : 1 })
>
```

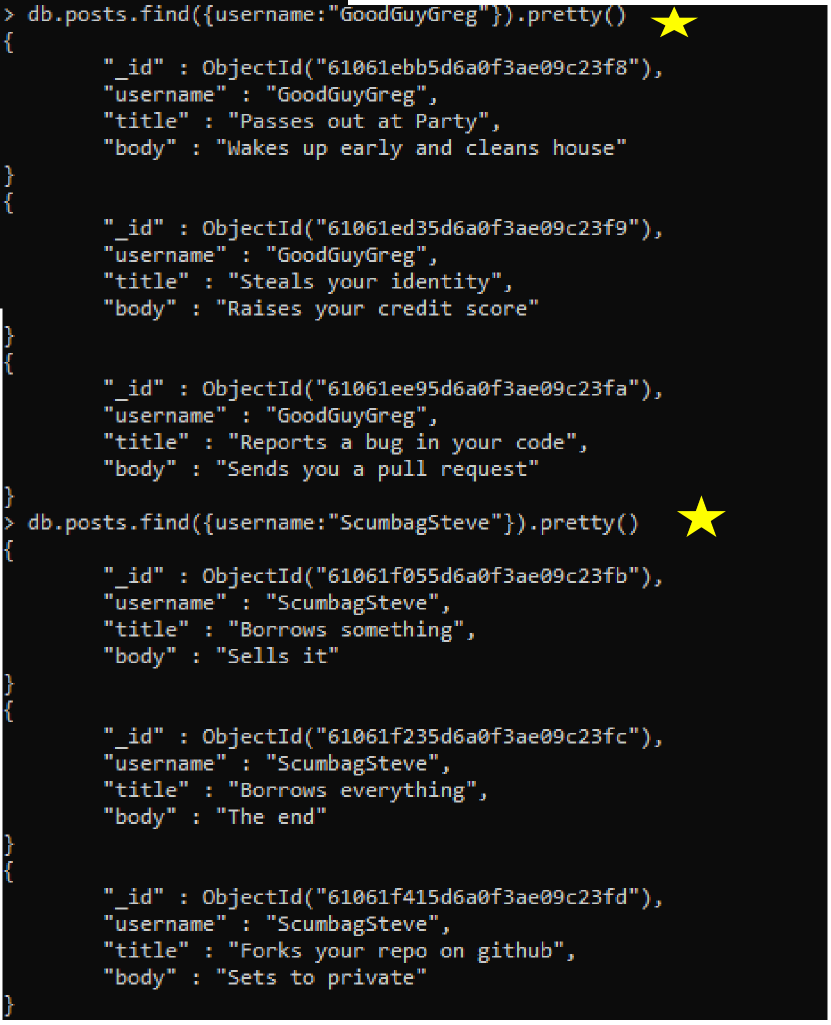
Querying Related Collection

Queries Used

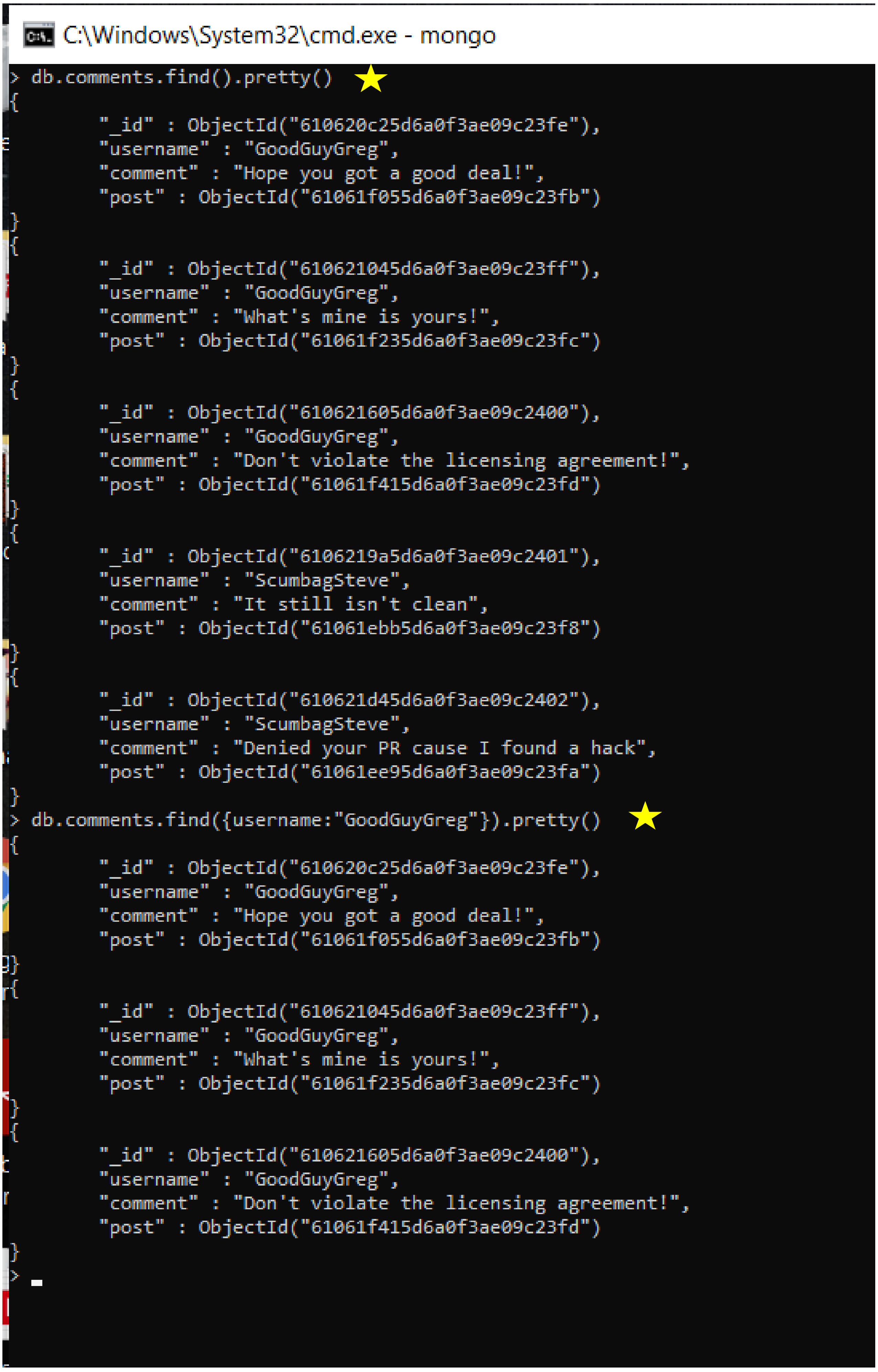
1. db.users.find().pretty()
2. db.posts.find().pretty()
3. db.posts.find({username:"GoodGuyGreg"}).pretty()
4. db.posts.find({username:"ScumbagSteve"}).pretty()
5. db.comments.find().pretty()
6. db.comments.find({username:"GoodGuyGreg"})
7. db.comments.find({username:"ScumbagSteve"})



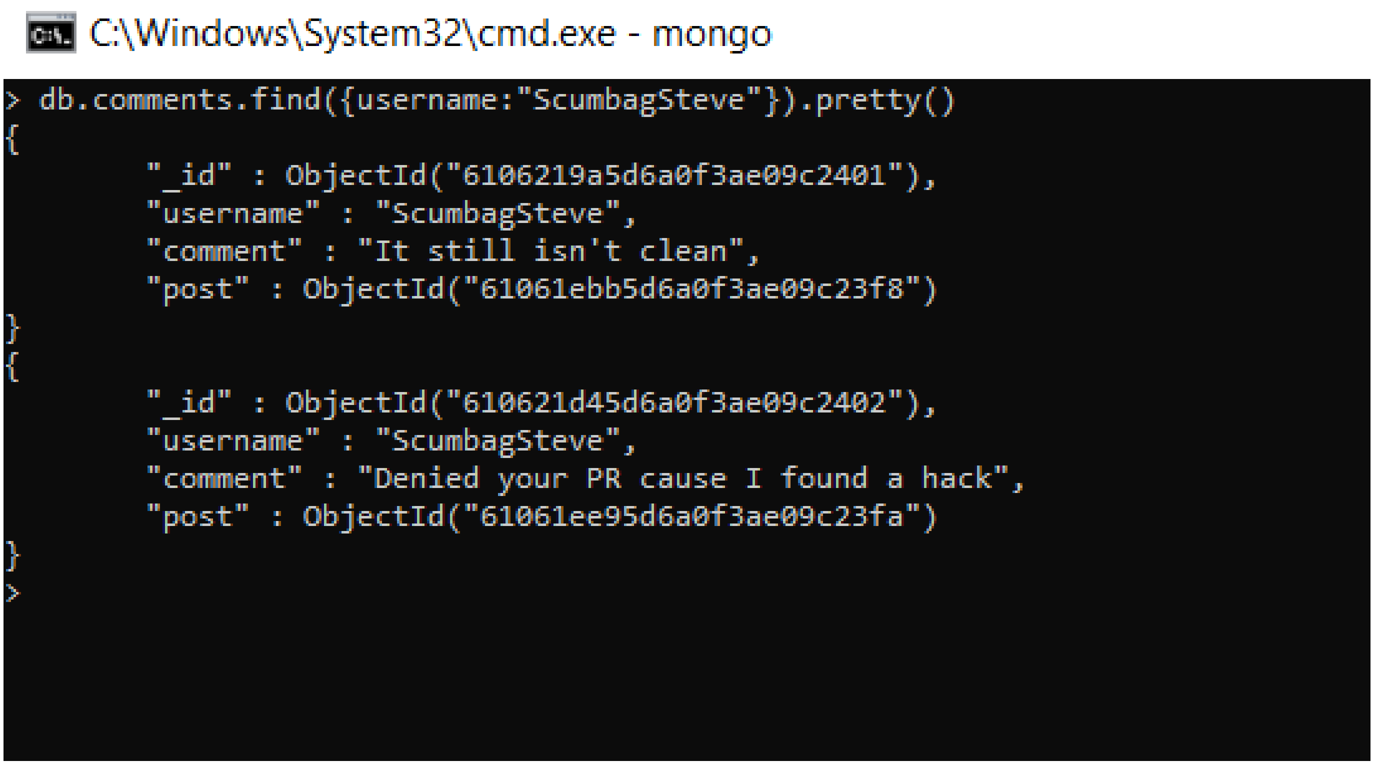
```
> db.users.find().pretty()
{
        "_id" : 1,
        "username" : "GoodGuyGreg",
        "first_name" : "Good Guy",
        "last_name" : "Greg"
}
{
        "_id" : 2,
        "username" : "ScumbagSteve",
        "fullname" : {
                "first" : "Scumbag",
                "last" : "Steve"
        }
}
> db.posts.find().pretty()
{
        "_id" : ObjectId("61061ebb5d6a0f3ae09c23f8"),
        "username" : "GoodGuyGreg",
        "title" : "Passes out at Party",
        "body" : "Wakes up early and cleans house"
}
{
        "_id" : ObjectId("61061ed35d6a0f3ae09c23f9"),
        "username" : "GoodGuyGreg",
        "title" : "Steals your identity",
        "body" : "Raises your credit score"
}
{
        "_id" : ObjectId("61061ee95d6a0f3ae09c23fa"),
        "username" : "GoodGuyGreg",
        "title" : "Reports a bug in your code",
        "body" : "Sends you a pull request"
}
{
        "_id" : ObjectId("61061f055d6a0f3ae09c23fb"),
        "username" : "ScumbagSteve",
        "title" : "Borrows something",
        "body" : "Sells it"
}
{
        "_id" : ObjectId("61061f235d6a0f3ae09c23fc"),
        "username" : "ScumbagSteve",
        "title" : "Borrows everything",
        "body" : "The end"
}
{
        "_id" : ObjectId("61061f415d6a0f3ae09c23fd"),
        "username" : "ScumbagSteve",
        "title" : "Forks your repo on github",
        "body" : "Sets to private"
}
>
```



```
> db.posts.find({username:"GoodGuyGreg"}).pretty()
{
        "_id" : ObjectId("61061ebb5d6a0f3ae09c23f8"),
        "username" : "GoodGuyGreg",
        "title" : "Passes out at Party",
        "body" : "Wakes up early and cleans house"
}
{
        "_id" : ObjectId("61061ed35d6a0f3ae09c23f9"),
        "username" : "GoodGuyGreg",
        "title" : "Steals your identity",
        "body" : "Raises your credit score"
}
{
        "_id" : ObjectId("61061ee95d6a0f3ae09c23fa"),
        "username" : "GoodGuyGreg",
        "title" : "Reports a bug in your code",
        "body" : "Sends you a pull request"
}
> db.posts.find({username:"ScumbagSteve"}).pretty()
{
        "_id" : ObjectId("61061f055d6a0f3ae09c23fb"),
        "username" : "ScumbagSteve",
        "title" : "Borrows something",
        "body" : "Sells it"
}
{
        "_id" : ObjectId("61061f235d6a0f3ae09c23fc"),
        "username" : "ScumbagSteve",
        "title" : "Borrows everything",
        "body" : "The end"
}
{
        "_id" : ObjectId("61061f415d6a0f3ae09c23fd"),
        "username" : "ScumbagSteve",
        "title" : "Forks your repo on github",
        "body" : "Sets to private"
}
```



```
> db.comments.find().pretty()
{
        "_id" : ObjectId("610620c25d6a0f3ae09c23fe"),
        "username" : "GoodGuyGreg",
        "comment" : "Hope you got a good deal!",
        "post" : ObjectId("61061f055d6a0f3ae09c23fb")
}
{
        "_id" : ObjectId("610621045d6a0f3ae09c23ff"),
        "username" : "GoodGuyGreg",
        "comment" : "What's mine is yours!",
        "post" : ObjectId("61061f235d6a0f3ae09c23fc")
}
{
        "_id" : ObjectId("610621605d6a0f3ae09c2400"),
        "username" : "GoodGuyGreg",
        "comment" : "Don't violate the licensing agreement!",
        "post" : ObjectId("61061f415d6a0f3ae09c23fd")
}
{
        "_id" : ObjectId("6106219a5d6a0f3ae09c2401"),
        "username" : "ScumbagSteve",
        "comment" : "It still isn't clean",
        "post" : ObjectId("61061ebb5d6a0f3ae09c23f8")
}
{
        "_id" : ObjectId("610621d45d6a0f3ae09c2402"),
        "username" : "ScumbagSteve",
        "comment" : "Denied your PR cause I found a hack",
        "post" : ObjectId("61061ee95d6a0f3ae09c23fa")
}
> db.comments.find({username:"GoodGuyGreg"}).pretty()
{
        "_id" : ObjectId("610620c25d6a0f3ae09c23fe"),
        "username" : "GoodGuyGreg",
        "comment" : "Hope you got a good deal!",
        "post" : ObjectId("61061f055d6a0f3ae09c23fb")
}
{
        "_id" : ObjectId("610621045d6a0f3ae09c23ff"),
        "username" : "GoodGuyGreg",
        "comment" : "What's mine is yours!",
        "post" : ObjectId("61061f235d6a0f3ae09c23fc")
}
{
        "_id" : ObjectId("610621605d6a0f3ae09c2400"),
        "username" : "GoodGuyGreg",
        "comment" : "Don't violate the licensing agreement!",
        "post" : ObjectId("61061f415d6a0f3ae09c23fd")
}
> _
```



C:\Windows\System32\cmd.exe - mongo

```
> db.comments.find({username:"ScumbagSteve"}).pretty()
{
        "_id" : ObjectId("6106219a5d6a0f3ae09c2401"),
        "username" : "ScumbagSteve",
        "comment" : "It still isn't clean",
        "post" : ObjectId("61061ebb5d6a0f3ae09c23f8")
}
{
        "_id" : ObjectId("610621d45d6a0f3ae09c2402"),
        "username" : "ScumbagSteve",
        "comment" : "Denied your PR cause I found a hack",
        "post" : ObjectId("61061ee95d6a0f3ae09c23fa")
}
>
```