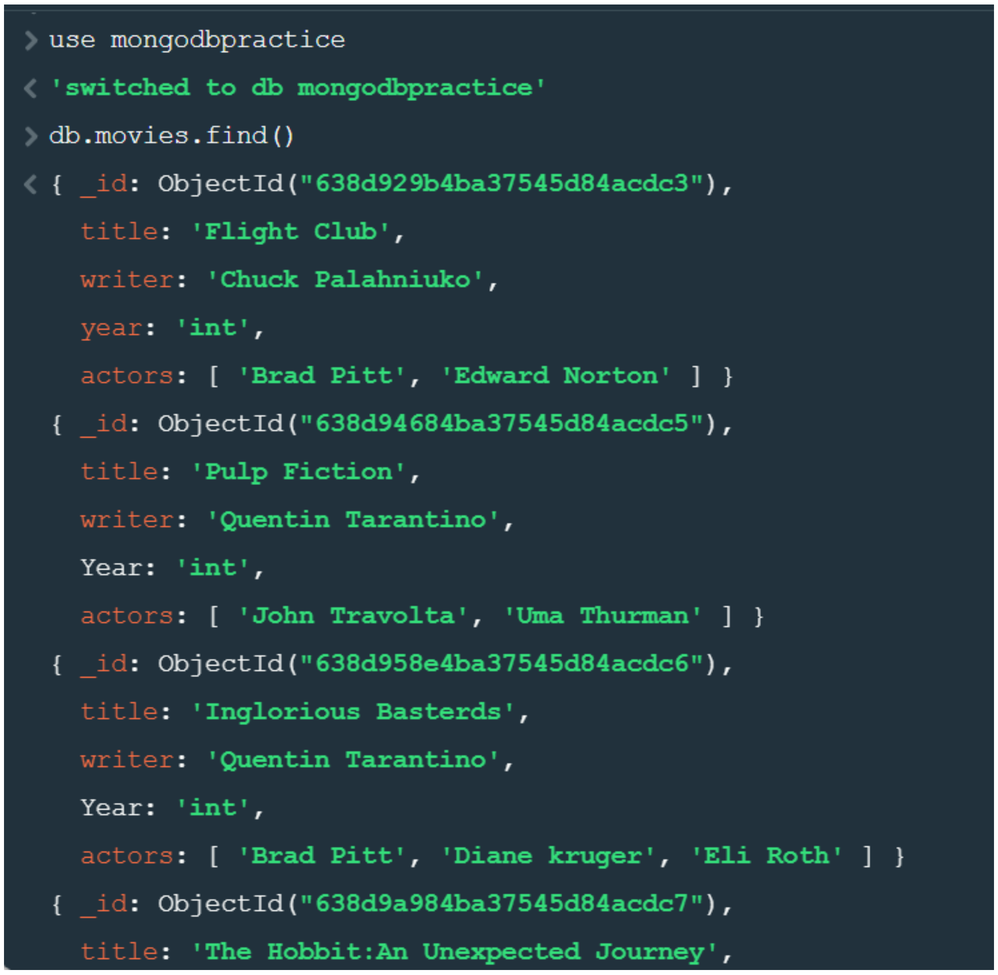
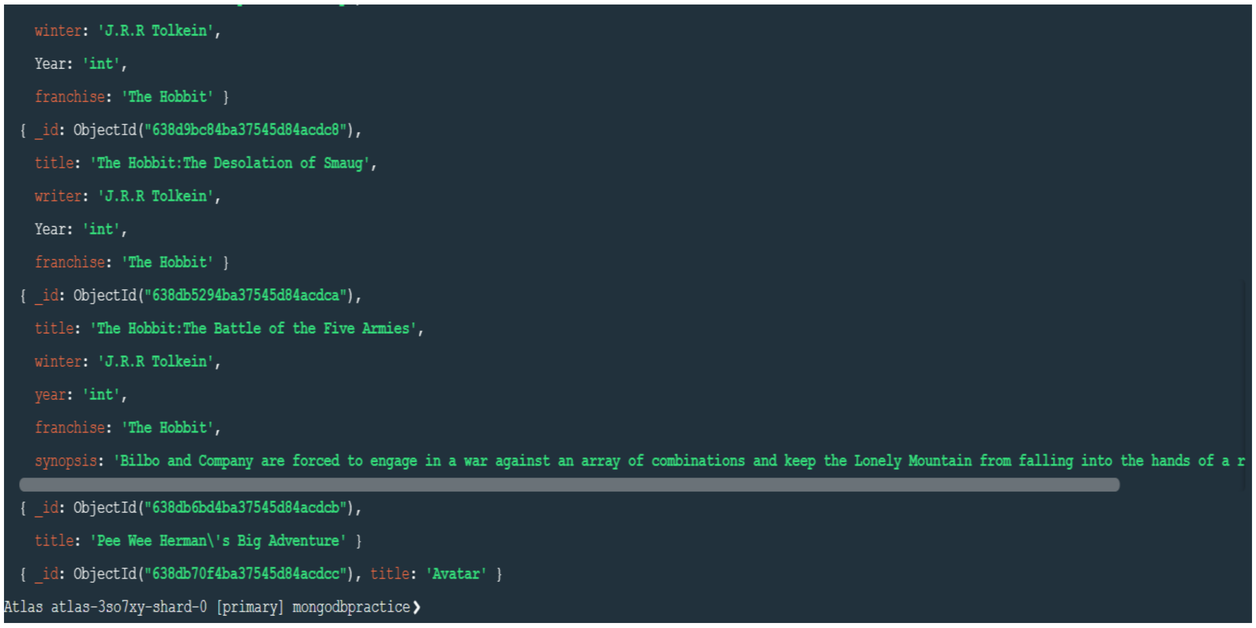
Query / Find Documents

1.get all documents

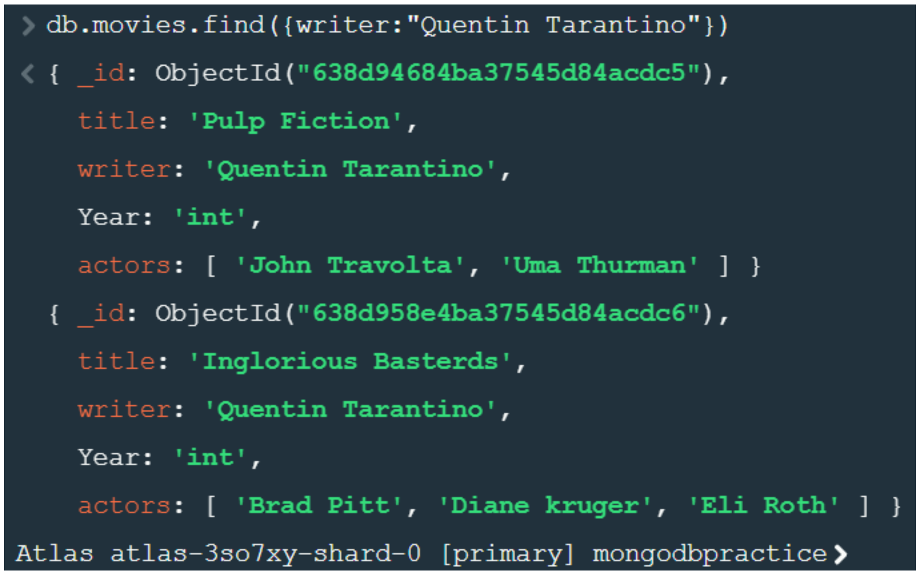


```
> use mongodbpractice
< 'switched to db mongodbpractice'
> db.movies.find()
< { _id: ObjectId("638d929b4ba37545d84acdc3"),
    title: 'Flight Club',
    writer: 'Chuck Palahniuko',
    year: 'int',
    actors: [ 'Brad Pitt', 'Edward Norton' ] }
  { _id: ObjectId("638d94684ba37545d84acdc5"),
    title: 'Pulp Fiction',
    writer: 'Quentin Tarantino',
    Year: 'int',
    actors: [ 'John Travolta', 'Uma Thurman' ] }
  { _id: ObjectId("638d958e4ba37545d84acdc6"),
    title: 'Inglorious Basterds',
    writer: 'Quentin Tarantino',
    Year: 'int',
    actors: [ 'Brad Pitt', 'Diane kruger', 'Eli Roth' ] }
  { _id: ObjectId("638d9a984ba37545d84acdc7"),
    title: 'The Hobbit:An Unexpected Journey',
```



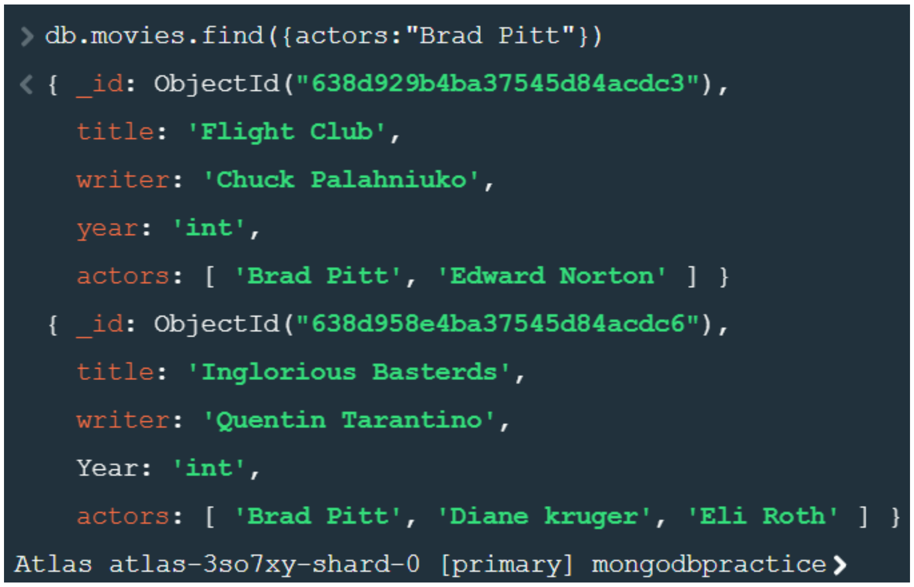
```
  winter: 'J.R.R Tolkein',
  Year: 'int',
  franchise: 'The Hobbit' }
{ _id: ObjectId("638d9bc84ba37545d84acdc8"),
  title: 'The Hobbit:The Desolation of Smaug',
  writer: 'J.R.R Tolkein',
  Year: 'int',
  franchise: 'The Hobbit' }
{ _id: ObjectId("638db5294ba37545d84acdca"),
  title: 'The Hobbit:The Battle of the Five Armies',
  winter: 'J.R.R Tolkein',
  year: 'int',
  franchise: 'The Hobbit',
  synopsis: 'Bilbo and Company are forced to engage in a war against an array of combinations and keep the Lonely Mountain from falling into the hands of a r
{ _id: ObjectId("638db6bd4ba37545d84acdcb"),
  title: 'Pee Wee Herman\'s Big Adventure' }
{ _id: ObjectId("638db70f4ba37545d84acdcc"), title: 'Avatar' }
Atlas atlas-3so7xy-shard-0 [primary] mongodbpractice❯
```

2.get all documents with `writer` set to "Quentin Tarantino"



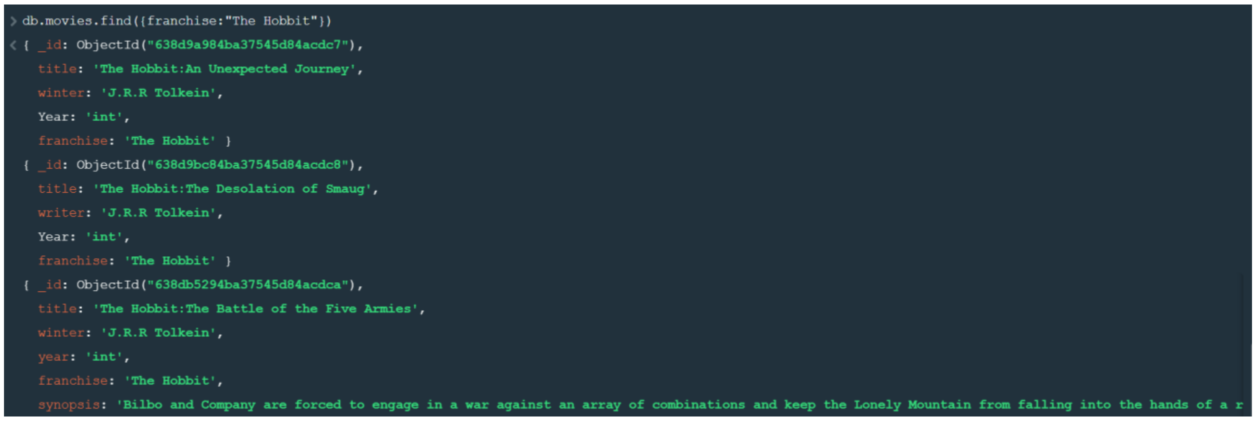
```
> db.movies.find({writer:"Quentin Tarantino"})
< { _id: ObjectId("638d94684ba37545d84acdc5"),
    title: 'Pulp Fiction',
    writer: 'Quentin Tarantino',
    Year: 'int',
    actors: [ 'John Travolta', 'Uma Thurman' ] }
  { _id: ObjectId("638d958e4ba37545d84acdc6"),
    title: 'Inglorious Basterds',
    writer: 'Quentin Tarantino',
    Year: 'int',
    actors: [ 'Brad Pitt', 'Diane kruger', 'Eli Roth' ] }
Atlas atlas-3so7xy-shard-0 [primary] mongodbpractice❯
```

3.get all documents where `actors` include "Brad Pitt"



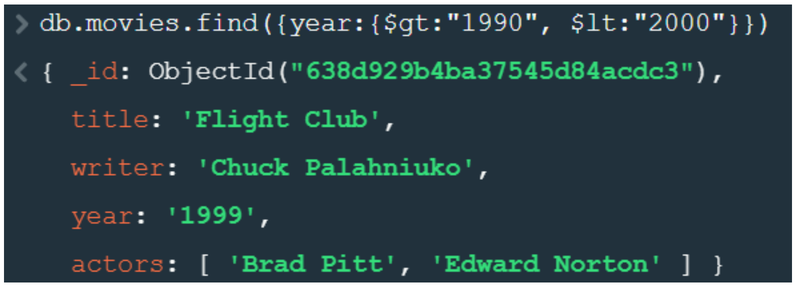
```
> db.movies.find({actors:"Brad Pitt"})
< { _id: ObjectId("638d929b4ba37545d84acdc3"),
    title: 'Flight Club',
    writer: 'Chuck Palahniuko',
    year: 'int',
    actors: [ 'Brad Pitt', 'Edward Norton' ] }
  { _id: ObjectId("638d958e4ba37545d84acdc6"),
    title: 'Inglorious Basterds',
    writer: 'Quentin Tarantino',
    Year: 'int',
    actors: [ 'Brad Pitt', 'Diane kruger', 'Eli Roth' ] }
Atlas atlas-3so7xy-shard-0 [primary] mongodbpractice>
```

4.get all documents with `franchise` set to "The Hobbit"



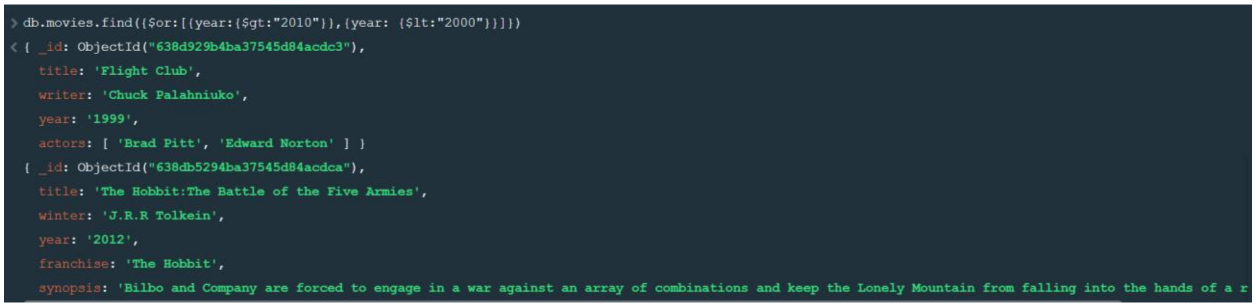
```
> db.movies.find({franchise:"The Hobbit"})
< { _id: ObjectId("638d9a984ba37545d84acdc7"),
    title: 'The Hobbit:An Unexpected Journey',
    winter: 'J.R.R Tolkein',
    Year: 'int',
    franchise: 'The Hobbit' }
  { _id: ObjectId("638d9bc84ba37545d84acdc8"),
    title: 'The Hobbit:The Desolation of Smaug',
    writer: 'J.R.R Tolkein',
    Year: 'int',
    franchise: 'The Hobbit' }
  { _id: ObjectId("638db5294ba37545d84acdca"),
    title: 'The Hobbit:The Battle of the Five Armies',
    winter: 'J.R.R Tolkein',
    year: 'int',
    franchise: 'The Hobbit',
    synopsis: 'Bilbo and Company are forced to engage in a war against an array of combinations and keep the Lonely Mountain from falling into the hands of a r
```

5.get all movies released in the 90s



```
> db.movies.find({year:{$gt:"1990", $lt:"2000"}})
< { _id: ObjectId("638d929b4ba37545d84acdc3"),
    title: 'Flight Club',
    writer: 'Chuck Palahniuko',
    year: '1999',
    actors: [ 'Brad Pitt', 'Edward Norton' ] }
```

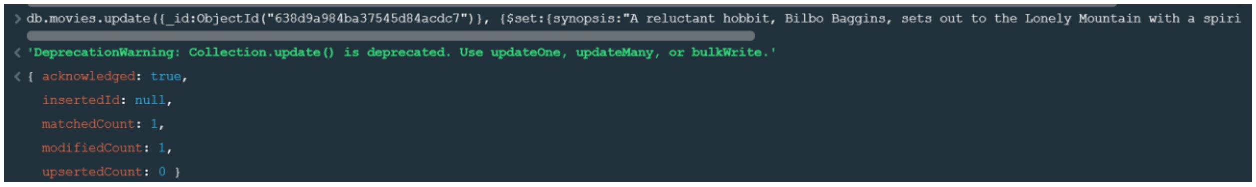
6.get all movies released before the year 2000 or after 2010



```
> db.movies.find({$or:[{year:{$gt:"2010"}},{year: {$lt:"2000"}}]})
< { _id: ObjectId("638d929b4ba37545d84acdc3"),
    title: 'Flight Club',
    writer: 'Chuck Palahniuko',
    year: '1999',
    actors: [ 'Brad Pitt', 'Edward Norton' ] }
  { _id: ObjectId("638db5294ba37545d84acdca"),
    title: 'The Hobbit:The Battle of the Five Armies',
    winter: 'J.R.R Tolkein',
    year: '2012',
    franchise: 'The Hobbit',
    synopsis: 'Bilbo and Company are forced to engage in a war against an array of combinations and keep the Lonely Mountain from falling into the hands of a r
```

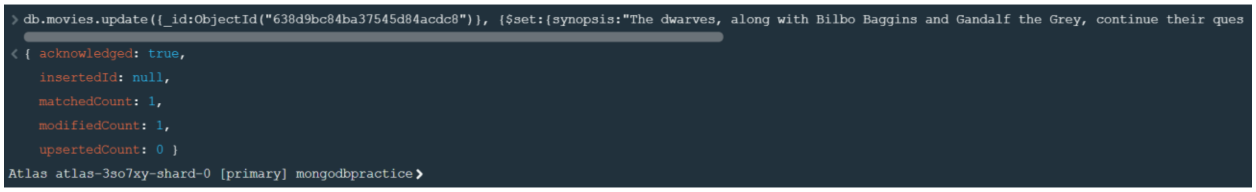
Update Documents

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."



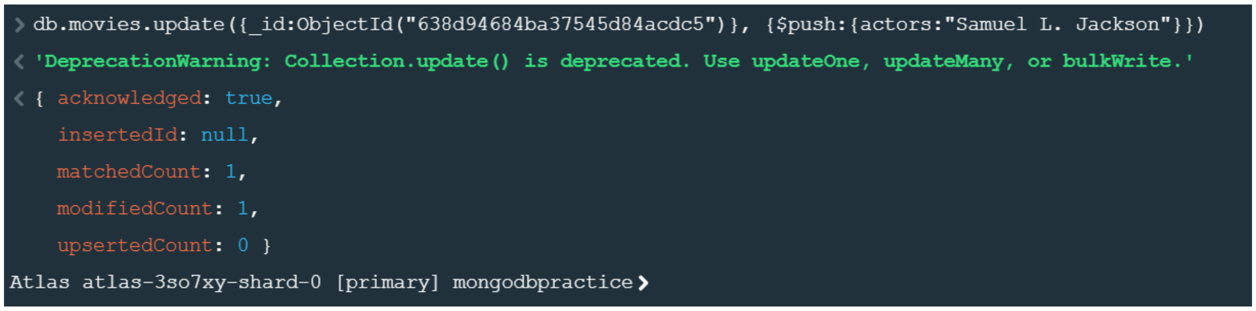
```
> db.movies.update({_id:ObjectId("638d9a984ba37545d84acdc7")}, {$set:{synopsis:"A reluctant hobbit, Bilbo Baggins, sets out to the Lonely Mountain with a spiri
< 'DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany, or bulkWrite.'
< { acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0 }
```

2.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.update({_id:ObjectId("638d9bc84ba37545d84acdc8")}, {$set:{synopsis:"The dwarves, along with Bilbo Baggins and Gandalf the Grey, continue their ques
< { acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0 }
Atlas atlas-3so7xy-shard-0 [primary] mongodbpractice>
```

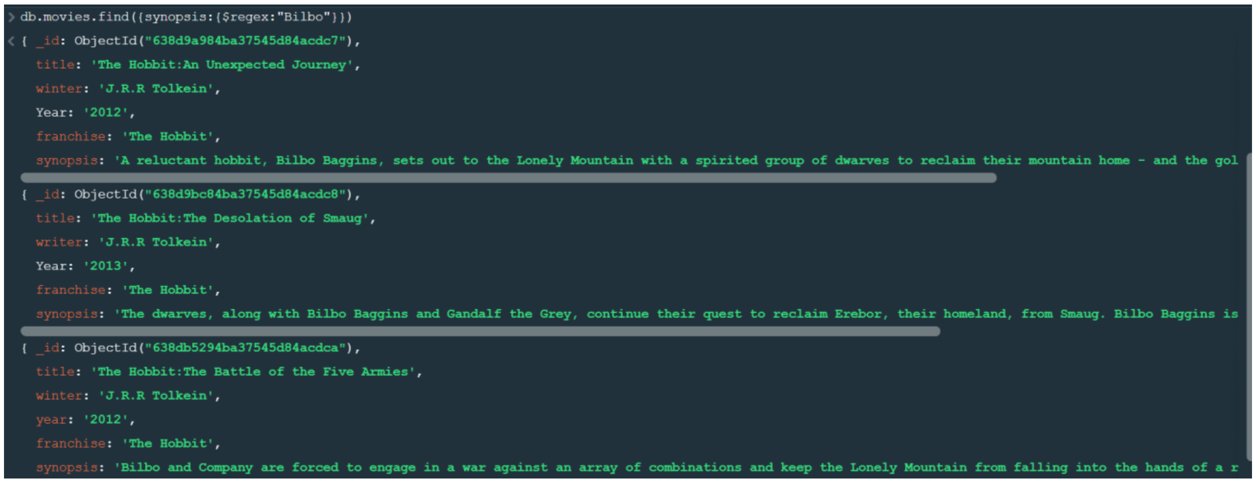
3.add an actor named "Samuel L. Jackson" to the movie "Pulp Fiction".



```
> db.movies.update({_id:ObjectId("638d94684ba37545d84acdc5")}, {$push:{actors:"Samuel L. Jackson"}})
< 'DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany, or bulkWrite.'
< { acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0 }
Atlas atlas-3so7xy-shard-0 [primary] mongodbpractice>
```

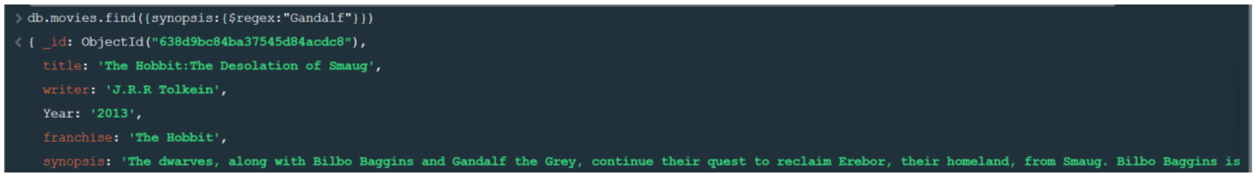
Text Search

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



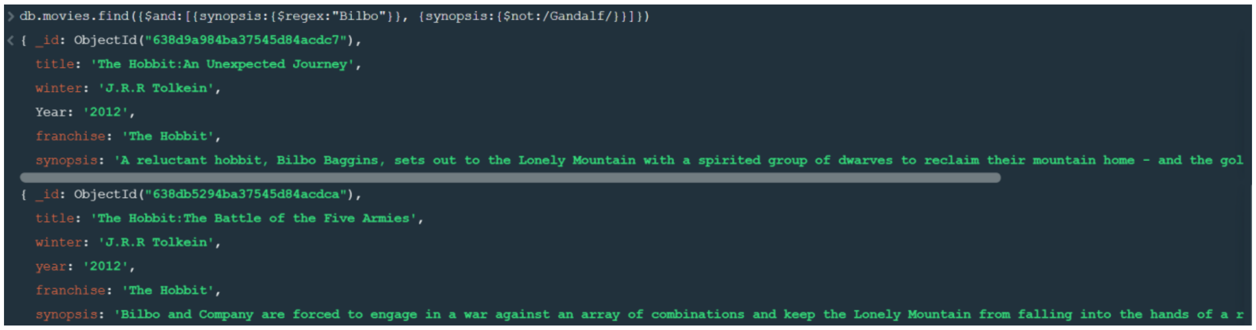
```
> db.movies.find({synopsis:{$regex:"Bilbo"}})
< { _id: ObjectId("638d9a984ba37545d84acdc7"),
    title: 'The Hobbit:An Unexpected Journey',
    winter: 'J.R.R Tolkein',
    Year: '2012',
    franchise: 'The Hobbit',
    synopsis: 'A reluctant hobbit, Bilbo Baggins, sets out to the Lonely Mountain with a spirited group of dwarves to reclaim their mountain home - and the gol
  { _id: ObjectId("638d9bc84ba37545d84acdc8"),
    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
  { _id: ObjectId("638db5294ba37545d84acdca"),
    title: 'The Hobbit:The Battle of the Five Armies',
    winter: 'J.R.R Tolkein',
    year: '2012',
    franchise: 'The Hobbit',
    synopsis: 'Bilbo and Company are forced to engage in a war against an array of combinations and keep the Lonely Mountain from falling into the hands of a r
```

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



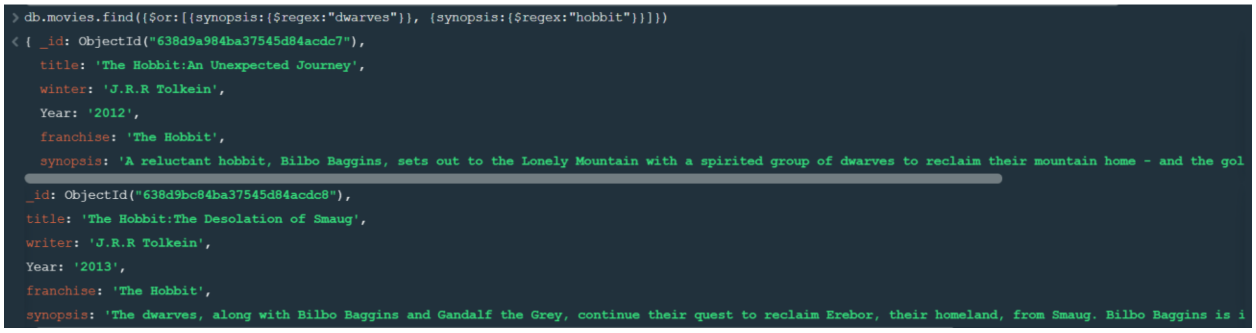
```
> db.movies.find({synopsis:{$regex:"Gandalf"}})
< { _id: ObjectId("638d9bc84ba37545d84acdc8"),
    title: 'The Hobbit:The Desolation of Smaug',
    writer: 'J.R.R Tolkein',
    Year: '2013',
    franchise: 'The Hobbit',
    synopsis: 'The dwarves, along with Bilbo Baggins and Gandalf the Grey, continue their quest to reclaim Erebor, their homeland, from Smaug. Bilbo Baggins is
```

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



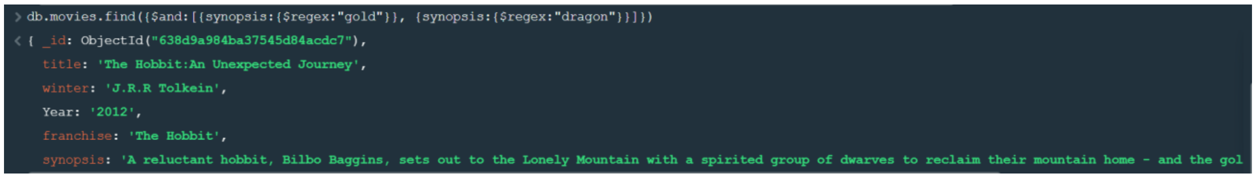
```
> db.movies.find({$and:[{synopsis:{$regex:"Bilbo"}}, {synopsis:{$not:/Gandalf/}}]})
< { _id: ObjectId("638d9a984ba37545d84acdc7"),
    title: 'The Hobbit:An Unexpected Journey',
    winter: 'J.R.R Tolkein',
    Year: '2012',
    franchise: 'The Hobbit',
    synopsis: 'A reluctant hobbit, Bilbo Baggins, sets out to the Lonely Mountain with a spirited group of dwarves to reclaim their mountain home - and the gol
  { _id: ObjectId("638db5294ba37545d84acdca"),
    title: 'The Hobbit:The Battle of the Five Armies',
    winter: 'J.R.R Tolkein',
    year: '2012',
    franchise: 'The Hobbit',
    synopsis: 'Bilbo and Company are forced to engage in a war against an array of combinations and keep the Lonely Mountain from falling into the hands of a r
```

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



```
> db.movies.find({$or:[{synopsis:{$regex:"dwarves"}}, {synopsis:{$regex:"hobbit"}}]})
< { _id: ObjectId("638d9a984ba37545d84acdc7"),
    title: 'The Hobbit:An Unexpected Journey',
    winter: '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 gol
  _id: ObjectId("638d9bc84ba37545d84acdc8"),
  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 i
```

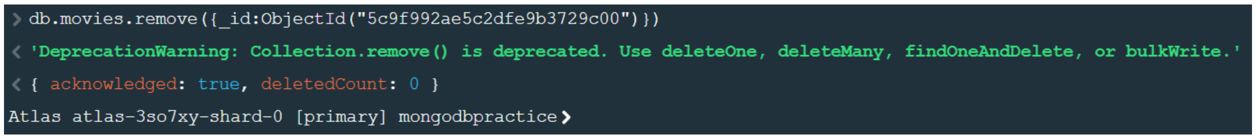
5.find all movies that have a synopsis that contains the word "gold" and "dragon".



```
> db.movies.find({$and:[{synopsis:{$regex:"gold"}}, {synopsis:{$regex:"dragon"}}]})
< { _id: ObjectId("638d9a984ba37545d84acdc7"),
    title: 'The Hobbit:An Unexpected Journey',
    winter: '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 gol
```

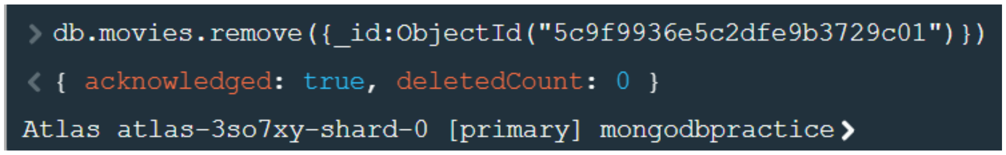
Delete Documents

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



```
> db.movies.remove({_id:ObjectId("5c9f992ae5c2dfe9b3729c00")})
< 'DeprecationWarning: Collection.remove() is deprecated. Use deleteOne, deleteMany, findOneAndDelete, or bulkWrite.'
< { acknowledged: true, deletedCount: 0 }
Atlas atlas-3so7xy-shard-0 [primary] mongodbpractice>
```

2.delete the movie "Ava.

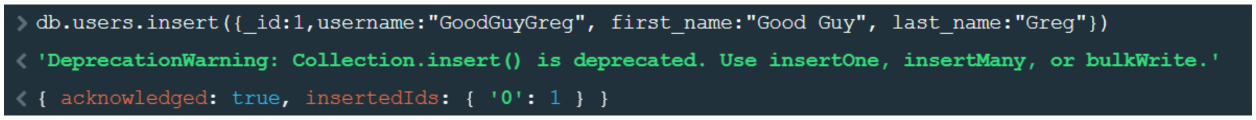


```
> db.movies.remove({_id:ObjectId("5c9f9936e5c2dfe9b3729c01")})
< { acknowledged: true, deletedCount: 0 }
Atlas atlas-3so7xy-shard-0 [primary] mongodbpractice>
```

Relationships

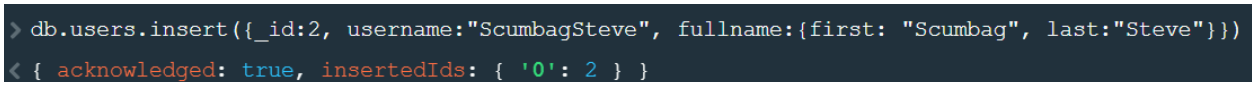
Insert the following documents into a users collection

username : GoodGuyGreg
first_name : "Good Guy"
last_name : "Greg"



```
> db.users.insert({_id:1,username:"GoodGuyGreg", first_name:"Good Guy", last_name:"Greg"})
< 'DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.'
< { acknowledged: true, insertedIds: { '0': 1 } }
```

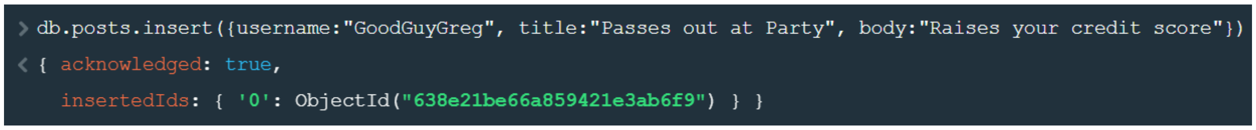
username : ScumbagSteve
full_name :
first : "Scumbag"
last : "Steve"



```
> db.users.insert({_id:2, username:"ScumbagSteve", fullname:{first: "Scumbag", last:"Steve"}})
< { acknowledged: true, insertedIds: { '0': 2 } }
```

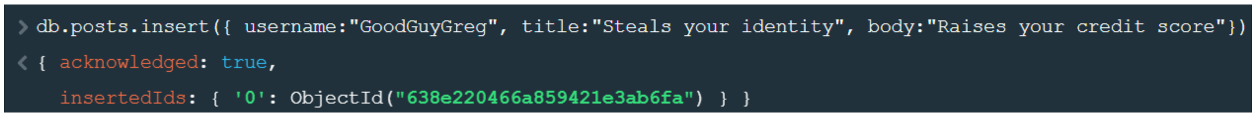
Insert the following documents into a posts collection

username : GoodGuyGreg
title : Passes out at party
body : Wakes up early and cleans house



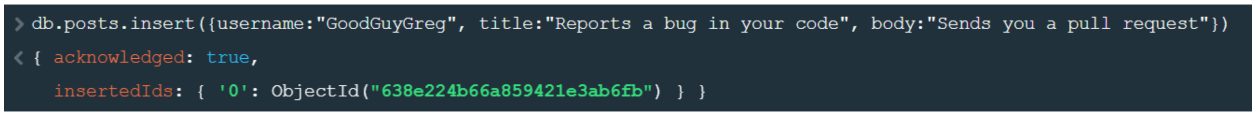
```
> db.posts.insert({username:"GoodGuyGreg", title:"Passes out at Party", body:"Raises your credit score"})
< { acknowledged: true,
    insertedIds: { '0': ObjectId("638e21be66a859421e3ab6f9") } }
```

username : GoodGuyGreg
title : Steals your identity
body : Raises your credit score



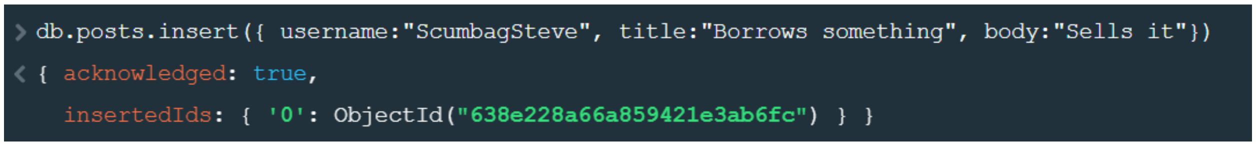
```
> db.posts.insert({ username:"GoodGuyGreg", title:"Steals your identity", body:"Raises your credit score"})
< { acknowledged: true,
    insertedIds: { '0': ObjectId("638e220466a859421e3ab6fa") } }
```

username : GoodGuyGreg
title : Reports a bug in your code
body : Sends you a Pull Request



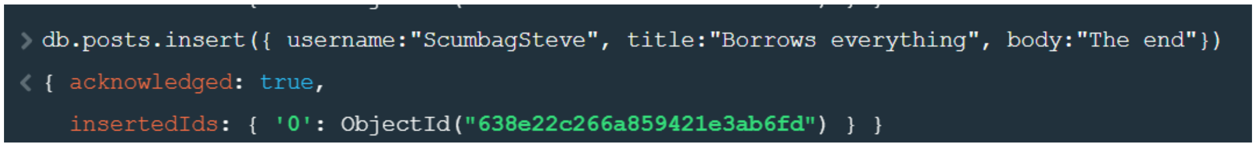
```
> db.posts.insert({username:"GoodGuyGreg", title:"Reports a bug in your code", body:"Sends you a pull request"})
< { acknowledged: true,
    insertedIds: { '0': ObjectId("638e224b66a859421e3ab6fb") } }
```

username : ScumbagSteve
title : Borrows something
body : Sells it



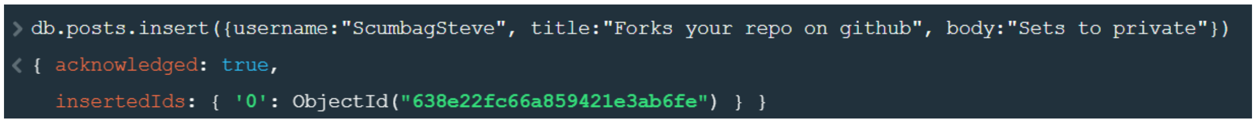
```
> db.posts.insert({ username:"ScumbagSteve", title:"Borrows something", body:"Sells it"})
< { acknowledged: true,
    insertedIds: { '0': ObjectId("638e228a66a859421e3ab6fc") } }
```

username : ScumbagSteve
title : Borrows everything
body : The end



```
> db.posts.insert({ username:"ScumbagSteve", title:"Borrows everything", body:"The end"})
< { acknowledged: true,
    insertedIds: { '0': ObjectId("638e22c266a859421e3ab6fd") } }
```

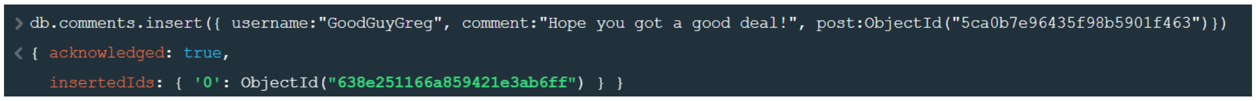
username : ScumbagSteve
title : Forks your repo on github
body : Sets to private



```
> db.posts.insert({username:"ScumbagSteve", title:"Forks your repo on github", body:"Sets to private"})
< { acknowledged: true,
    insertedIds: { '0': ObjectId("638e22fc66a859421e3ab6fe") } }
```

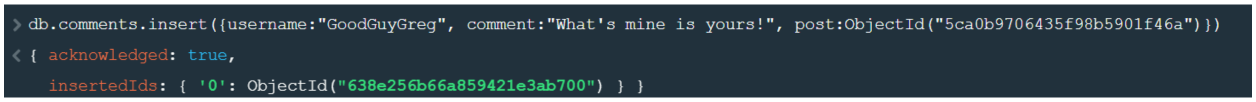
Insert the following documents into a comments collection

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"



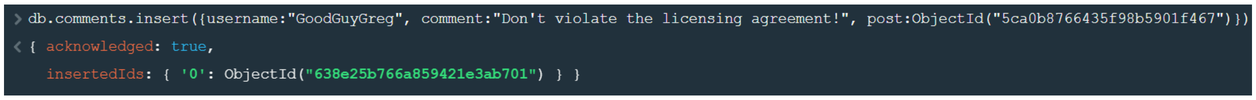
```
> db.comments.insert({ username:"GoodGuyGreg", comment:"Hope you got a good deal!", post:ObjectId("5ca0b7e96435f98b5901f463")})
< { acknowledged: true,
    insertedIds: { '0': ObjectId("638e251166a859421e3ab6ff") } }
```

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"



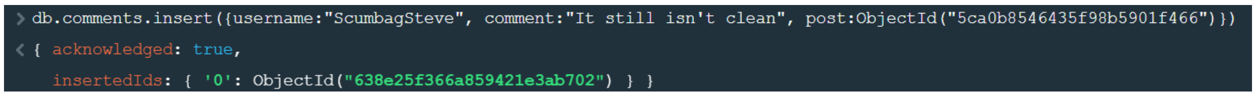
```
> db.comments.insert({username:"GoodGuyGreg", comment:"What's mine is yours!", post:ObjectId("5ca0b9706435f98b5901f46a")})
< { acknowledged: true,
    insertedIds: { '0': ObjectId("638e256b66a859421e3ab700") } }
```

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



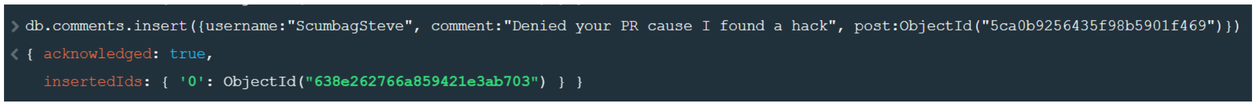
```
> db.comments.insert({username:"GoodGuyGreg", comment:"Don't violate the licensing agreement!", post:ObjectId("5ca0b8766435f98b5901f467")})
< { acknowledged: true,
    insertedIds: { '0': ObjectId("638e25b766a859421e3ab701") } }
```

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"



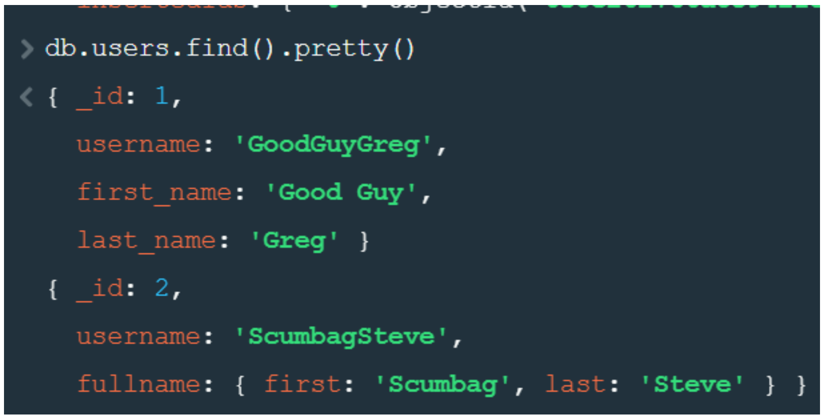
```
> db.comments.insert({username:"ScumbagSteve", comment:"It still isn't clean", post:ObjectId("5ca0b8546435f98b5901f466")})
< { acknowledged: true,
    insertedIds: { '0': ObjectId("638e25f366a859421e3ab702") } }
```

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"



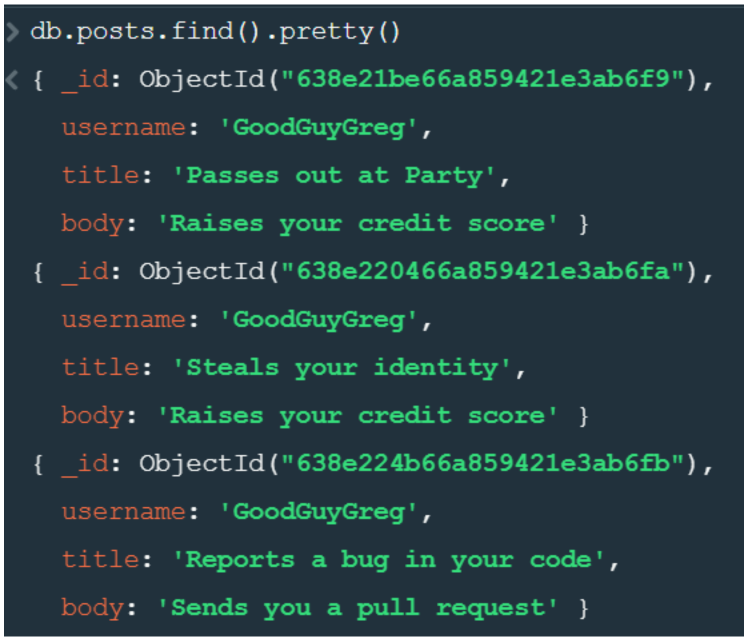
```
> db.comments.insert({username:"ScumbagSteve", comment:"Denied your PR cause I found a hack", post:ObjectId("5ca0b9256435f98b5901f469")})
< { acknowledged: true,
    insertedIds: { '0': ObjectId("638e262766a859421e3ab703") } }
```

1.find all users

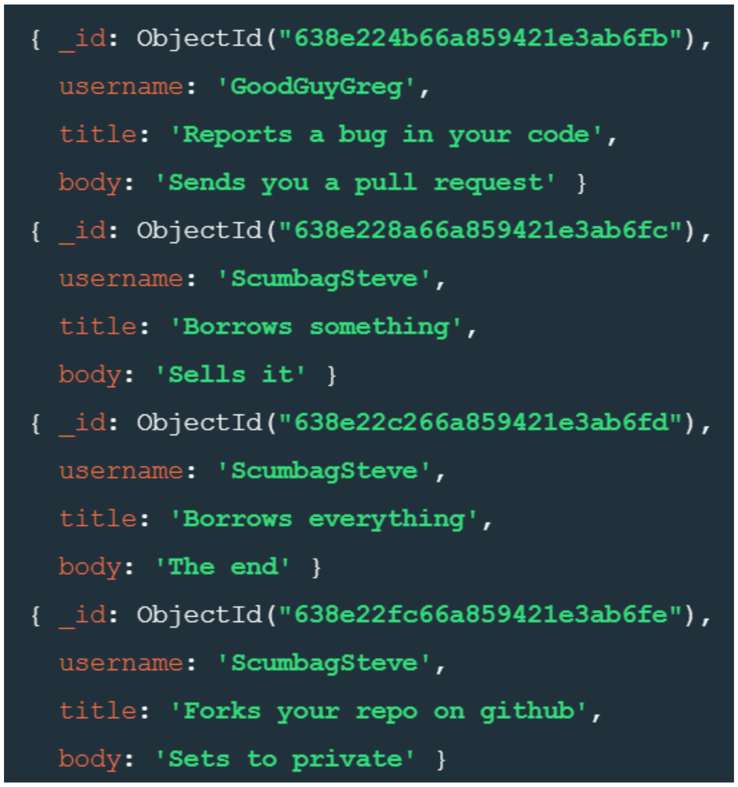


```
> db.users.find().pretty()
< { _id: 1,
    username: 'GoodGuyGreg',
    first_name: 'Good Guy',
    last_name: 'Greg' }
  { _id: 2,
    username: 'ScumbagSteve',
    fullname: { first: 'Scumbag', last: 'Steve' } }
```

2.find all posts

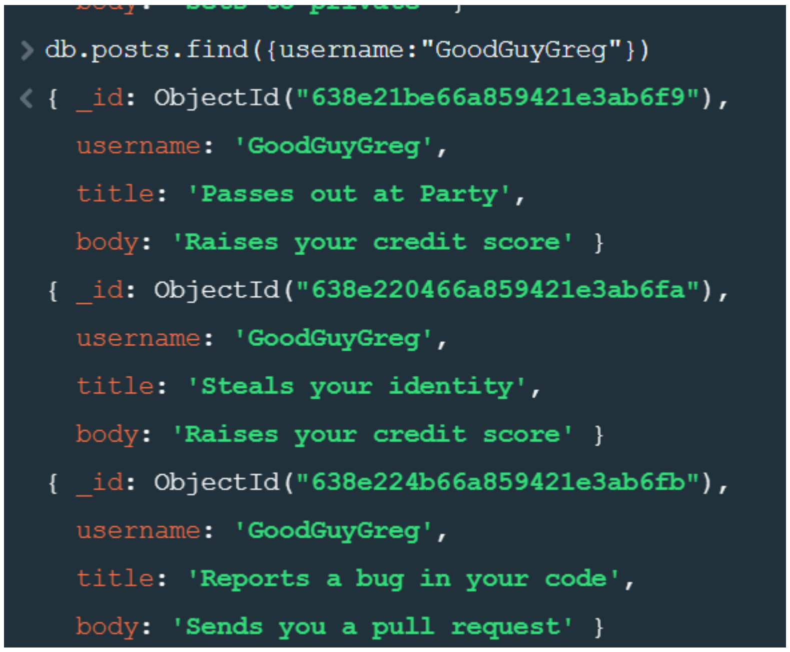


```
> db.posts.find().pretty()
< { _id: ObjectId("638e21be66a859421e3ab6f9"),
    username: 'GoodGuyGreg',
    title: 'Passes out at Party',
    body: 'Raises your credit score' }
  { _id: ObjectId("638e220466a859421e3ab6fa"),
    username: 'GoodGuyGreg',
    title: 'Steals your identity',
    body: 'Raises your credit score' }
  { _id: ObjectId("638e224b66a859421e3ab6fb"),
    username: 'GoodGuyGreg',
    title: 'Reports a bug in your code',
    body: 'Sends you a pull request' }
```



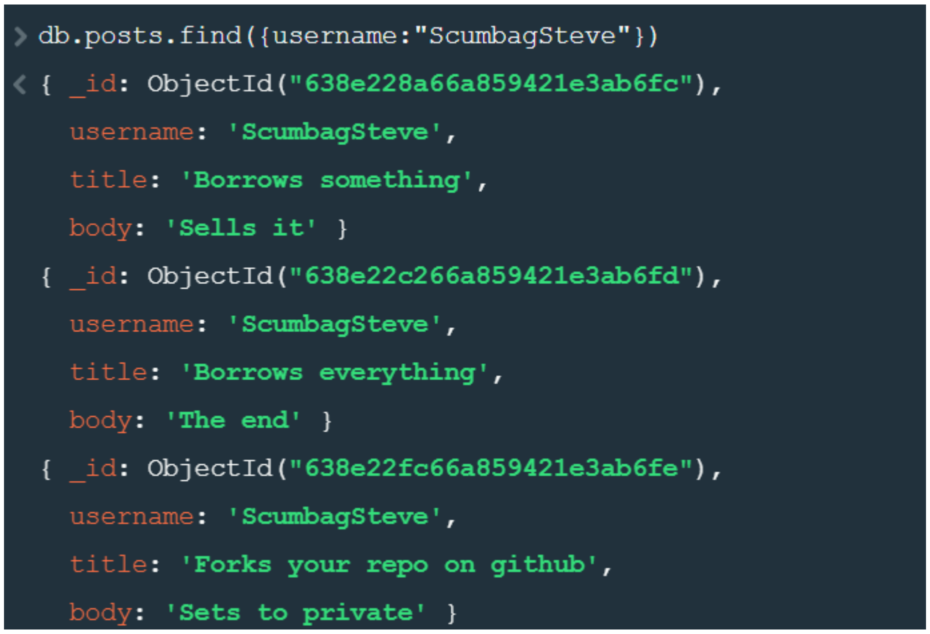
```
  { _id: ObjectId("638e224b66a859421e3ab6fb"),
    username: 'GoodGuyGreg',
    title: 'Reports a bug in your code',
    body: 'Sends you a pull request' }
  { _id: ObjectId("638e228a66a859421e3ab6fc"),
    username: 'ScumbagSteve',
    title: 'Borrows something',
    body: 'Sells it' }
  { _id: ObjectId("638e22c266a859421e3ab6fd"),
    username: 'ScumbagSteve',
    title: 'Borrows everything',
    body: 'The end' }
  { _id: ObjectId("638e22fc66a859421e3ab6fe"),
    username: 'ScumbagSteve',
    title: 'Forks your repo on github',
    body: 'Sets to private' }
```

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



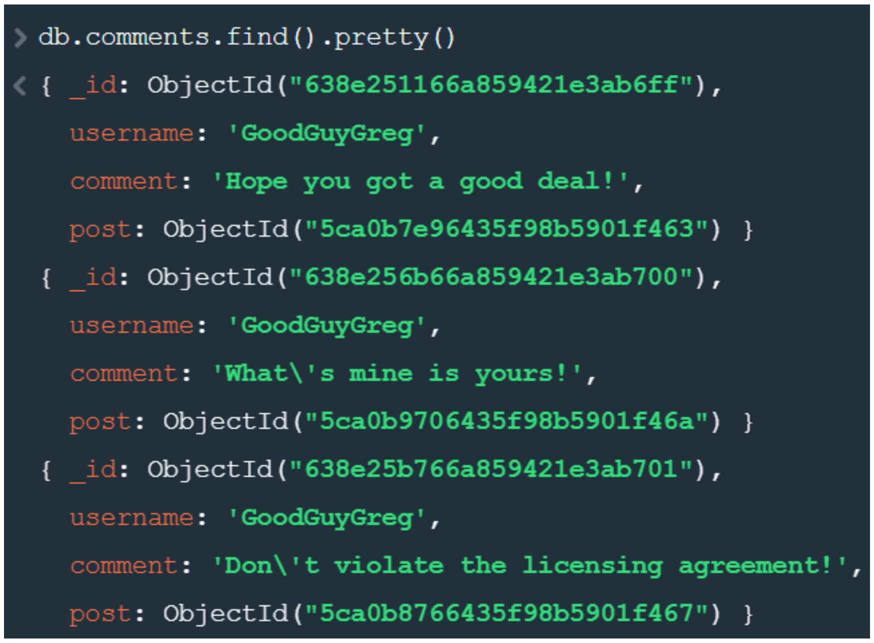
```
> db.posts.find({username:"GoodGuyGreg"})
< { _id: ObjectId("638e21be66a859421e3ab6f9"),
    username: 'GoodGuyGreg',
    title: 'Passes out at Party',
    body: 'Raises your credit score' }
  { _id: ObjectId("638e220466a859421e3ab6fa"),
    username: 'GoodGuyGreg',
    title: 'Steals your identity',
    body: 'Raises your credit score' }
  { _id: ObjectId("638e224b66a859421e3ab6fb"),
    username: 'GoodGuyGreg',
    title: 'Reports a bug in your code',
    body: 'Sends you a pull request' }
```

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

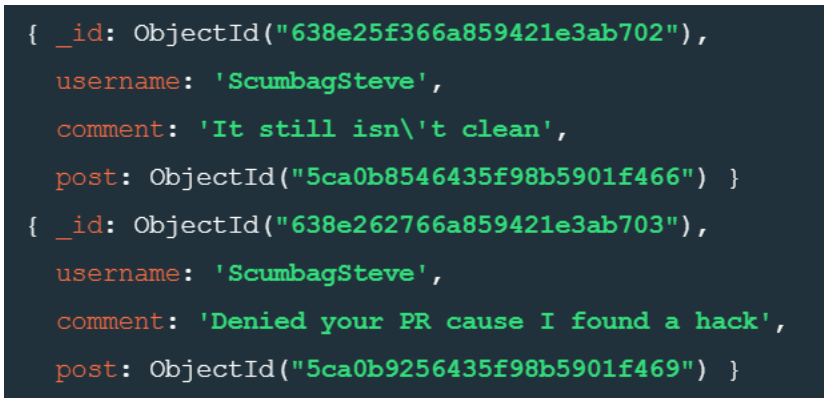


```
> db.posts.find({username:"ScumbagSteve"})
< { _id: ObjectId("638e228a66a859421e3ab6fc"),
    username: 'ScumbagSteve',
    title: 'Borrows something',
    body: 'Sells it' }
  { _id: ObjectId("638e22c266a859421e3ab6fd"),
    username: 'ScumbagSteve',
    title: 'Borrows everything',
    body: 'The end' }
  { _id: ObjectId("638e22fc66a859421e3ab6fe"),
    username: 'ScumbagSteve',
    title: 'Forks your repo on github',
    body: 'Sets to private' }
```

5.find all comments

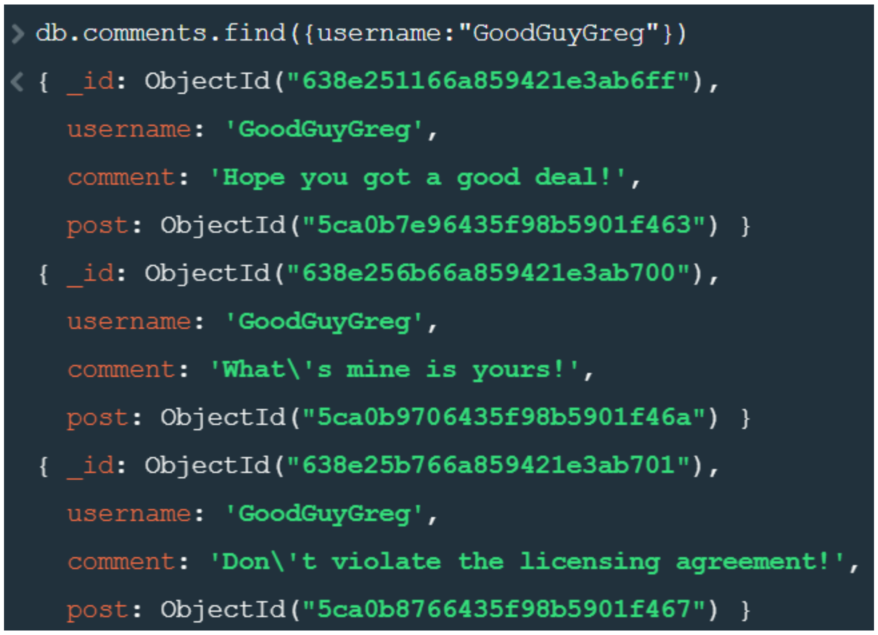


```
> db.comments.find().pretty()
< { _id: ObjectId("638e251166a859421e3ab6ff"),
    username: 'GoodGuyGreg',
    comment: 'Hope you got a good deal!',
    post: ObjectId("5ca0b7e96435f98b5901f463") }
  { _id: ObjectId("638e256b66a859421e3ab700"),
    username: 'GoodGuyGreg',
    comment: 'What\'s mine is yours!',
    post: ObjectId("5ca0b9706435f98b5901f46a") }
  { _id: ObjectId("638e25b766a859421e3ab701"),
    username: 'GoodGuyGreg',
    comment: 'Don\'t violate the licensing agreement!',
    post: ObjectId("5ca0b8766435f98b5901f467") }
```

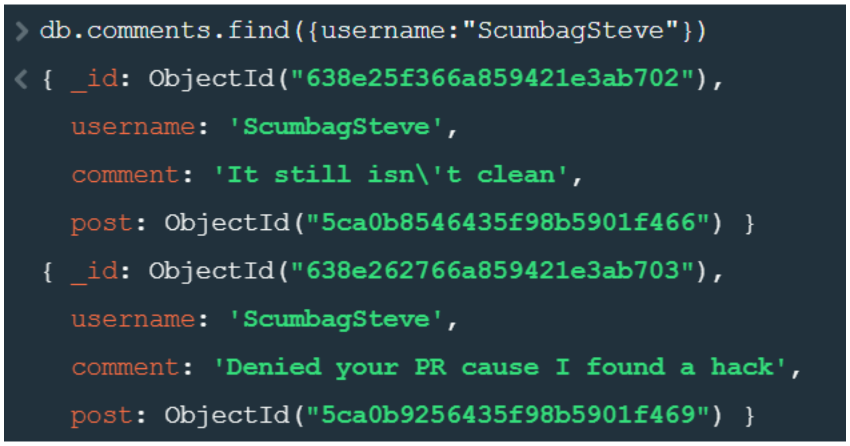


```
{ _id: ObjectId("638e25f366a859421e3ab702"),
  username: 'ScumbagSteve',
  comment: 'It still isn\'t clean',
  post: ObjectId("5ca0b8546435f98b5901f466") }
{ _id: ObjectId("638e262766a859421e3ab703"),
  username: 'ScumbagSteve',
  comment: 'Denied your PR cause I found a hack',
  post: ObjectId("5ca0b9256435f98b5901f469") }
```

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


```
> db.comments.find({username:"GoodGuyGreg"})
< { _id: ObjectId("638e251166a859421e3ab6ff"),
    username: 'GoodGuyGreg',
    comment: 'Hope you got a good deal!',
    post: ObjectId("5ca0b7e96435f98b5901f463") }
  { _id: ObjectId("638e256b66a859421e3ab700"),
    username: 'GoodGuyGreg',
    comment: 'What\'s mine is yours!',
    post: ObjectId("5ca0b9706435f98b5901f46a") }
  { _id: ObjectId("638e25b766a859421e3ab701"),
    username: 'GoodGuyGreg',
    comment: 'Don\'t violate the licensing agreement!',
    post: ObjectId("5ca0b8766435f98b5901f467") }
```

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


```
> db.comments.find({username:"ScumbagSteve"})
< { _id: ObjectId("638e25f366a859421e3ab702"),
    username: 'ScumbagSteve',
    comment: 'It still isn\'t clean',
    post: ObjectId("5ca0b8546435f98b5901f466") }
  { _id: ObjectId("638e262766a859421e3ab703"),
    username: 'ScumbagSteve',
    comment: 'Denied your PR cause I found a hack',
    post: ObjectId("5ca0b9256435f98b5901f469") }
```