Query the movies collection to

1.get all documents

```
> db.Movies.find()
< { _id: ObjectId("638d8f6243519c2487ae18fe"),
    title: 'Fight Club',
    writer: 'Chuck Palahniuko',
    year: '1999',
    actors: [ 'Brad Pitt', 'Edward Norton' ] }
  { _id: ObjectId("638d91c843519c2487ae18ff"),
    title: 'Pulp Fiction',
    writer: 'Quentin Tarantino',
    year: '1994',
    actors: [ 'John Travolta', 'Uma Thurman' ] }
  { _id: ObjectId("638d931443519c2487ae1900"),
    title: 'Inglorious Basterds',
    writer: 'Quentin Tarantio',
    year: '2009',
    actors: [ 'Brad Pitt', 'Diane Kruger', 'Eli Roth' ] }
  { _id: ObjectId("638d945243519c2487ae1902"),
    title: 'The Hobbit:An Unexpected Journey',
    Writer: 'J.R.R.Tolkein',
    year: '2012',
```

```
  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 gold
{ _id: ObjectId("638d94f443519c2487ae1903"),
  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
{ _id: ObjectId("638d991743519c2487ae1905"),
  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 risin
{ _id: ObjectId("638d9aea43519c2487ae1906"),
  title: 'Pee Wee Herman\'s Big Adventure' }
{ _id: ObjectId("638d9b9343519c2487ae1907"), title: 'Avatar' }
```

2.get all documents with writer set to “Quentin Tarantino”

```
> db.Movies.find({writer:"Quentin Tarantino"})
< { _id: ObjectId("638d91c843519c2487ae18ff"),
    title: 'Pulp Fiction',
    writer: 'Quentin Tarantino',
    year: '1994',
    actors: [ 'John Travolta', 'Uma Thurman' ] }
```

3.get all documents where actors include”Brad Pitt”

```
> db.Movies.find({actors:"Brad Pitt"})
< { _id: ObjectId("638d8f6243519c2487ae18fe"),
    title: 'Fight Club',
    writer: 'Chuck Palahniuko',
    year: '1999',
    actors: [ 'Brad Pitt', 'Edward Norton' ] }
  { _id: ObjectId("638d931443519c2487ae1900"),
    title: 'Inglorious Basterds',
    writer: 'Quentin Tarantio',
    year: '2009',
    actors: [ 'Brad Pitt', 'Diane Kruger', 'Eli Roth' ] }
```

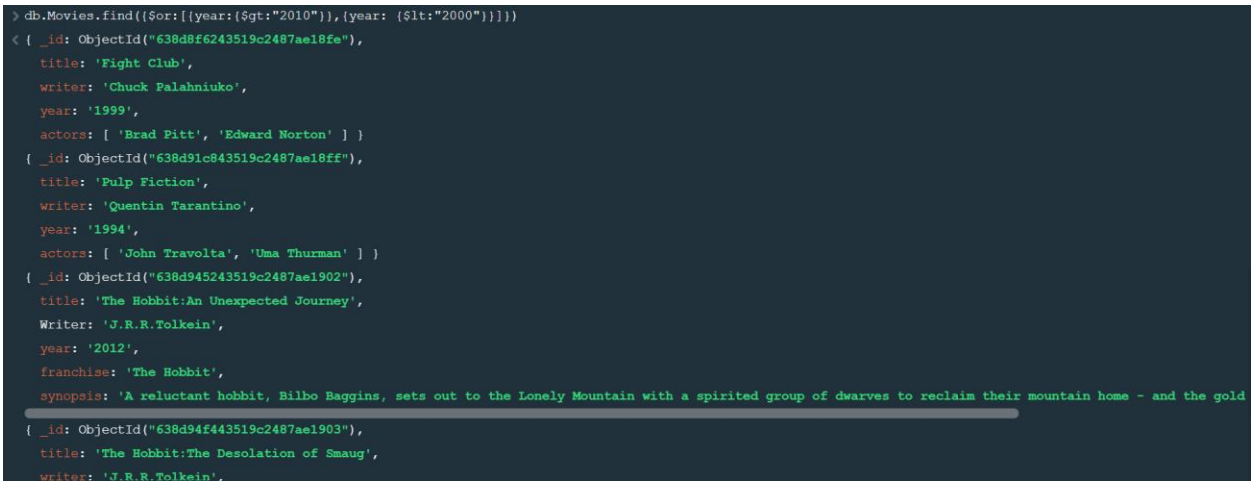
4.get all documents with franchise set to “The Hobbit”

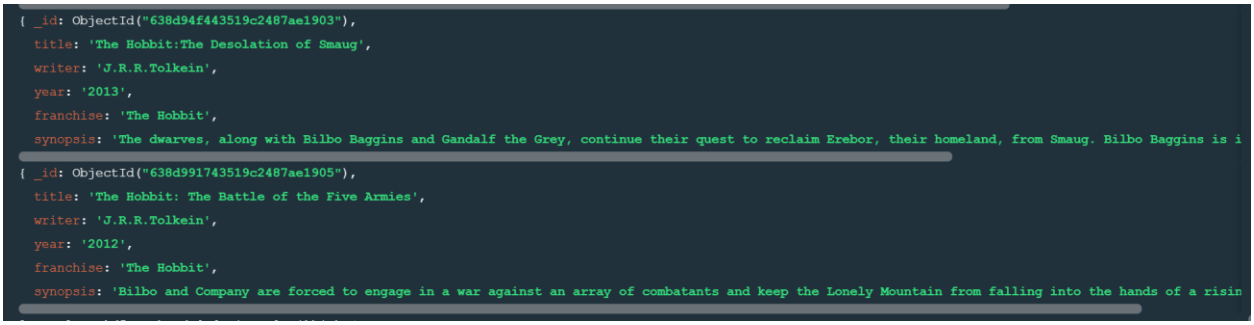
```
db.Movies.find({franchise:"The Hobbit"})
{ _id: ObjectId("638d945243519c2487ae1902"),
  title: 'The Hobbit:An Unexpected Journey',
  Writer: 'J.R.R.Tolkein',
  year: '2012',
  franchise: 'The Hobbit',
  synopsis: 'A reluctant hobbit, Bilbo Baggins, sets out to the Lonely Mountain with a spirited group of dwarves to reclaim their mountain home - and the gold
{ _id: ObjectId("638d94f443519c2487ae1903"),
  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
{ _id: ObjectId("638d991743519c2487ae1905"),
  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 risin
```

5.get all movies released in the 90s.

```
> db.Movies.find({year:{$gt:"1990", $lt:"2000"}})
< { _id: ObjectId("638d8f6243519c2487ae18fe"),
    title: 'Fight Club',
    writer: 'Chuck Palahniuko',
    year: '1999',
    actors: [ 'Brad Pitt', 'Edward Norton' ] }
  { _id: ObjectId("638d91c843519c2487ae18ff"),
    title: 'Pulp Fiction',
    writer: 'Quentin Tarantino',
    year: '1994',
    actors: [ 'John Travolta', 'Uma Thurman' ] }
```

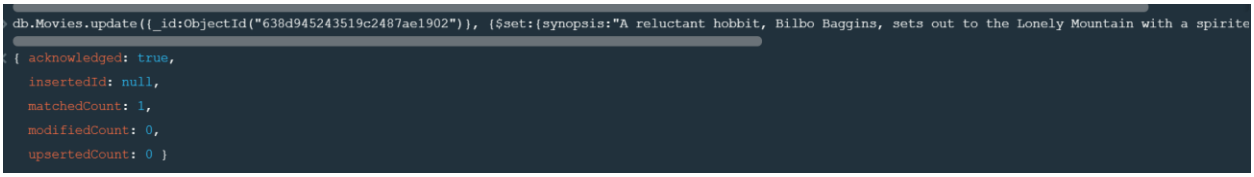
6. get all movies released before the year 2000 or after 2010





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."



2. 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("638d94f443519c2487ae1903")}, {$set:{synopsis:"The dwarves, along with Bilbo Baggins and Gandalf the Grey, continue their q
< { acknowledged: true,
    insertedId: null,
    matchedCount: 0,
    modifiedCount: 0,
    upsertedCount: 0 }
```

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

```
> db.movies.update({_id:ObjectId("638d91c843519c2487ae18ff")}, {$push:{actors:"Samuel L. Jackson"}})
< { acknowledged: true,
    insertedId: null,
    matchedCount: 0,
    modifiedCount: 0,
    upsertedCount: 0 }
```

Text Search

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

```
> db.Movies.find({synopsis:{$regex:"Bilbo"}})
< { _id: ObjectId("638d945243519c2487ae1902"),
    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 Mountain with a spirited group of dwarves to reclaim their mountain home - and the gold
  { _id: ObjectId("638d94f443519c2487ae1903"),
    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
  _id: ObjectId("638d991743519c2487ae1905"),
  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
```

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

```
> db.Movies.find({synopsis:{$regex:"Gandalf"}})
< { _id: ObjectId("638d94f443519c2487ae1903"),
    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
```

3. 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("638d945243519c2487ae1902"),
    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 Mountain with a spirited group of dwarves to reclaim their mountain home - and the gold
  { _id: ObjectId("638d991743519c2487ae1905"),
    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 risin
```

4. 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("638d945243519c2487ae1902"),
    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 Mountain with a spirited group of dwarves to reclaim their mountain home - and the gol
  { _id: ObjectId("638d94f443519c2487ae1903"),
    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
```

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("638d945243519c2487ae1902"),
    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 Mountain with a spirited group of dwarves to reclaim their mountain home - and the gol
Atlas atlas-skd5aa-shard-0 [primary] Likhitha>
```

Delete Documents

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

```
> db.Movies.remove({_id:ObjectId("638d9aea43519c2487ae1906")})
< 'DeprecationWarning: Collection.remove() is deprecated. Use deleteOne, deleteMany, findOneAndDelete, or bulkWrite.'
< { acknowledged: true, deletedCount: 1 }
```

2. delete the movie "Avatar"

```
> db.Movies.remove({_id:ObjectId("638d9b9343519c2487ae1907")})
< { acknowledged: true, deletedCount: 1 }
Atlas atlas-skd5aa-shard-0 [primary] Likhitha>
```

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("638dfef0601b9318fb0bd478") } }
```

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

```
> db.posts.insert({username:"GoodGuyGreg", title:"Reports a bug in your code", body:"Sends you a pull request"})
< { acknowledged: true,
    insertedIds: { '0': ObjectId("638dff64601b9318fb0bd479") } }
```

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("638dffc0601b9318fb0bd47a") } }
```

username : ScumbagSteve
title : Borrows something
body : Sells it

```
db.posts.insert({ username:"ScumbagSteve", title:"Borrows something", body:"Sells it"})
{ acknowledged: true,
  insertedIds: { '0': ObjectId("638e0022601b9318fb0bd47b") } }
```

username : ScumbagSteve
title : Borrows everything
body : The end

```
> db.posts.insert({ username:"ScumbagSteve", title:"Borrows everything", body:"The end"})
< { acknowledged: true,
    insertedIds: { '0': ObjectId("638e0095601b9318fb0bd47c") } }
```

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("638e0129601b9318fb0bd47d") } }
```

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("638e08ac601b9318fb0bd47f") } }
```

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("638e0973601b9318fb0bd480") } }
```

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("638e098a601b9318fb0bd481") } }
```

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("638e099d601b9318fb0bd482") } }
```

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("638e09aa601b9318fb0bd483") } }
```

Querying related collections

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("638def0601b9318fb0bd478"),
    username: 'GoodGuyGreg',
    title: 'Passes out at Party',
    body: 'Raises your credit score' }
  { _id: ObjectId("638dff64601b9318fb0bd479"),
    username: 'GoodGuyGreg',
    title: 'Reports a bug in your code',
    body: 'Sends you a pull request' }
  { _id: ObjectId("638dffc0601b9318fb0bd47a"),
    username: 'GoodGuyGreg',
    title: 'Reports a bug in your code',
    body: 'Sends you a pull request' }
  { _id: ObjectId("638e0022601b9318fb0bd47b"),
    username: 'ScumbagSteve',
    title: 'Borrows something',
    body: 'Sells it' }
```

```
    body: 'Sells it' }
  { _id: ObjectId("638e0095601b9318fb0bd47c"),
    username: 'ScumbagSteve',
    title: 'Borrows everything',
    body: 'The end' }
  { _id: ObjectId("638e0129601b9318fb0bd47d"),
    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("638def0601b9318fb0bd478"),
    username: 'GoodGuyGreg',
    title: 'Passes out at Party',
    body: 'Raises your credit score' }
  { _id: ObjectId("638dff64601b9318fb0bd479"),
    username: 'GoodGuyGreg',
    title: 'Reports a bug in your code',
    body: 'Sends you a pull request' }
  { _id: ObjectId("638dffc0601b9318fb0bd47a"),
    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("638e0022601b9318fb0bd47b"),
    username: 'ScumbagSteve',
    title: 'Borrows something',
    body: 'Sells it' }
  { _id: ObjectId("638e0095601b9318fb0bd47c"),
    username: 'ScumbagSteve',
    title: 'Borrows everything',
    body: 'The end' }
  { _id: ObjectId("638e0129601b9318fb0bd47d"),
    username: 'ScumbagSteve',
    title: 'Forks your repo on github',
    body: 'Sets to private' }
```

5. find all comments

```
> db.comments.find().pretty()
< { _id: ObjectId("638e08a8601b9318fb0bd47e"),
    username: 'GoodGuyGreg',
    comment: 'Hope you got a good deal!',
    post: ObjectId("5ca0b7e96435f98b5901f463") }
  { _id: ObjectId("638e08ac601b9318fb0bd47f"),
    username: 'GoodGuyGreg',
    comment: 'Hope you got a good deal!',
    post: ObjectId("5ca0b7e96435f98b5901f463") }
  { _id: ObjectId("638e0973601b9318fb0bd480"),
    username: 'GoodGuyGreg',
    comment: 'What\'s mine is yours!',
    post: ObjectId("5ca0b9706435f98b5901f46a") }
  { _id: ObjectId("638e098a601b9318fb0bd481"),
    username: 'GoodGuyGreg',
    comment: 'Don\'t violate the licensing agreement!',
    post: ObjectId("5ca0b8766435f98b5901f467") }
  { _id: ObjectId("638e099d601b9318fb0bd482"),
    username: 'ScumbagSteve',
```

```
{ _id: ObjectId("638e099d601b9318fb0bd482"),
  username: 'ScumbagSteve',
  comment: 'It still isn\'t clean',
  post: ObjectId("5ca0b8546435f98b5901f466") }
{ _id: ObjectId("638e09aa601b9318fb0bd483"),
  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("638e08a8601b9318fb0bd47e"),
    username: 'GoodGuyGreg',
    comment: 'Hope you got a good deal!',
    post: ObjectId("5ca0b7e96435f98b5901f463") }
  { _id: ObjectId("638e08ac601b9318fb0bd47f"),
    username: 'GoodGuyGreg',
    comment: 'Hope you got a good deal!',
    post: ObjectId("5ca0b7e96435f98b5901f463") }
  { _id: ObjectId("638e0973601b9318fb0bd480"),
    username: 'GoodGuyGreg',
    comment: 'What\'s mine is yours!',
    post: ObjectId("5ca0b9706435f98b5901f46a") }
  { _id: ObjectId("638e098a601b9318fb0bd481"),
    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("638e099d601b9318fb0bd482"),
  username: 'ScumbagSteve',
  comment: 'It still isn\'t clean',
  post: ObjectId("5ca0b8546435f98b5901f466") }
{ _id: ObjectId("638e09aa601b9318fb0bd483"),
  username: 'ScumbagSteve',
  comment: 'Denied your PR cause I found a hack',
  post: ObjectId("5ca0b9256435f98b5901f469") }
```