**MongoDB Lab Assignments -Day 1**

**Insert Documents**

****

**Insert the following documents into a movies collection.**

**[**

**title : Fight Club**

**writer : Chuck Palahniuko**

**year : 1999**

**actors : [**

**Brad Pitt**

**Edward Norton**

**]**

>>db.movies.insert({title:"Fight Club", writer: "Chuck Palahniuk", year: "1999", actors:["Brad Pitt", "Edward Norton"]})

**title : Pulp Fiction**

**writer : Quentin Tarantino**

**year : 1994**

**actors : [**

**John Travolta**

**Uma Thurman**

**]**

db.movies.insert({title:"Pulp Fiction", writer:"Quentin Tarantino", year:"2009", actors:["John Travolta", "Uma Thurman"]})

**title : Inglorious Basterds**

**writer : Quentin Tarantino**

**year : 2009**

**actors : [**

**Brad Pitt**

**Diane Kruger**

**Eli Roth**

**]**

**db.movies.insert({title:"Inglorious Basterds", writer:"Quentin Tarantino", year:"2009", actors:["Brad Pitt", "Diane Kruger", "Eli Roth"]})**

**title : The Hobbit: An Unexpected Journey**

**writer : J.R.R. Tolkein**

**year : 2012**

**franchise : The Hobbit**

**db.movies.insert({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**

db.movies.insert({title:"The Hobbit: The Desolation of Smaug", writer:"J.R.R Tolkien", year:"2013", franchise:"The Hobbit"})

**title : The Hobbit: The Battle of the Five Armies**

**writer : J.R.R. Tolkein**

**year : 2012**

**franchise : The Hobbit**

**synopsis : Bilbo and Company are forced to engage in a war against an array of combatants and keep the Lonely Mountain from falling into the hands of a rising darkness.**

db.movies.insert({title:"The Hobbit: The Battle of the Five Armies", writer:"J.R.R Tolkien", year:"2002", franchise:"The Hobbit", synopsis:"Bilbo and Company are forced to engage in a war against an array of combatants and keep the Lonely Mountain from falling into the hands of a rising darkness."})

**title : Pee Wee Herman's Big Adventure**

**title : Avatar**

db.movies.insertMany({title:"Pee Wee Herman's Big Adventures"},({title:"Avatar})

**Query / Find Documents**

1. get all documents

db.movies.find()

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

db.movies.find({writer:"Quentin Tarantino"})

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

db.movies.find({actors:"Brad Pitt"})

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

db.movies.find({franchise:"The Hobbit"})

5. get all movies released in the 90s

db.movies.find({year:{$gt:"1990", $lt:"2000"}})

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

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

**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("2c6f59e6e8c2dfe7b3729bfe")}, {$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. 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("4c6fa52ae6c9dfe9b3729c03")}, {$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. add an actor named "Samuel L. Jackson" to the movie "Pulp Fiction"**

**```**

**db.movies.update({\_id:ObjectId("6c8f953ce9c8dfe9b3729bfc")}, {$push:{actors:"Samuel L. Jackson"}})**

Text Search

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

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

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

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

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/}}]})

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"}}]})

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"}}]})

**Delete Documents**

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

**```**

**db.movies.remove({\_id:ObjectId("9c8f792ae5c2dfe9b3729c00")})**

**```**

**2. delete the movie "Avatar"**

**```**

**db.movies.remove({\_id:ObjectId("8c7f6936e5c2dfe9b3729c01")})**

```

**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"})**

**username : ScumbagSteve**

**full\_name :**

**first : "Scumbag"**

**last : "Steve"**

**db.users.insert({\_id:2, username:"ScumbagSteve", fullname:{first: "Scumbag", last:"Steve"}})**

**Insert the following documents into a `posts` collection**

**username : GoodGuyGreg**

**title : Passes out at party**

**body : Wakes up early and cleans house**

**db.posts.insert({username:"GoodGuyGreg", title:"Passes out at Party", body:"Raises your credit score"})**

**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"})**

**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"})**

**username : ScumbagSteve**

**title : Borrows something**

**body : Sells it**

**db.posts.insert({ username:"ScumbagSteve", title:"Borrows something", body:"Sells it"})**

**username : ScumbagSteve**

**title : Borrows everything**

**body : The end**

**db.posts.insert({ username:"ScumbagSteve", title:"Borrows everything", body:"The end"})**

**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"})**

**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("8ca8b5e76435f98b5901f463")})

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("8ca0b9607435f98b5901f46a")})

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("8ca4b2786435f98b5901f467")})

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("8ca0b9146435f98b5901f466")})

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("8ca0b6257435f98b5901f469")})

**Querying related collections**

**1. find all users**

**db.users.find().pretty()**

**2. find all posts**

**db.posts.find().pretty()**

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

**```**

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

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

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

**5. find all comments**

**db.comments.find().pretty()**

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

**db.comments.find({username:"GoodGuyGreg"})**

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

**db.comments.find({username:"ScumbagSteve"})**

**```**

**8. find all comments belonging to the post "Reports a bug in your code"**

**db.comments.find(**post:ObjectId("8ca0b6257435f98b5901f469")**)**