MongoDB Day 1 Assignment

Query/Find Documents

1. Get all documents.  
   Query – db.movies.find() or db.movies.find().pretty()
2. Get all documents with writer set to “Quentin Tarantino”.  
   Query – db.movies.find({ “Writer”:”Quentin Tarantino”}).pretty()
3. Get all documents where actors include Badd Pitt.   
   Query – db.movies.find({ “actors.actorone”: “Brad Pitt”}).pretty()
4. Get all the documents with franchise set to “The Hobbit”  
   Query – db.movies.find({“franchise”: “The Hobbit”}).pretty()
5. Get all movies released in the 90s  
   Query – db.movies.find({ year: { $gte:1990, $lte:1999}}).pretty()
6. Get all the movies realeased before the year 2000 or after 2010  
   Query – db.movies.find({ $or: [{year: { $lt:2000}}, {year:{$gt:2010}}}]}).pretty()

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”.  
     
   Query – db.movies.update( {"title" : "The Hobbit: the 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. 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”.  
     
   Query –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. Add an Actor named “Samuel L. Jackson” to the movie “Pulp Fiction”   
   Query – db.movies.update({“title”:”Pulp Fiction”}, { $push:{“actors”:”Samuel L. Jackson”}})

Text Search

1. Find all movies that have a synopsis that contains the word “Bilbo”.  
   Query – db.movies.find({synopsis:/Bilbo/}).pretty()
2. Find all the movies that have a synopsis that contains the word “Gandalf”  
   Query – db.movies.find({synopsis:/Gandalf/}).pretty()
3. Find all the movies that have a synopsis that contains the word”Bilbo” and not the word “Gandalf”  
   Query - db.movies.find({$and: [{synopsis:/Bilbo/}, {synopsis : {$not: /Gandalf/}}]}).pretty()
4. Find all movies that have a synopsis that contains the word “dwarves” or “hobbit”  
   Query - db.movies.find({$or: [{synopsis: /hobbit/}, {synopsis: /dwarves/}]}).pretty()
5. Find all movies that have a synopsis that contains the word “gold” and “dragon”  
   Query - db.movies.find({$and: [{synopsis:/gold/}, {synopsis:/dragon/}]}).pretty()

Delete Documents

1. Delete the movie “Pee Wee Herman’s Big Adventure”  
   Query – db.movies.remove({title:”Pee Wee Herman’s Big Adventure”})
2. Delete the movie Avatar  
   Query – db.movies.remove({title:”Avatar”})

Relationships

1. db.users.insert([ { username:"GoodGuyGreg", first\_name:"GoodGuy", last\_name:"Guy" }, { username:"ScumbagSteve", full\_name:{ first:"Scumbag", last:"Steve"}}])
2. db.posts.insert([  
    {   
    \_id:1,  
    username:”GoodGuyGreg”,  
    title: “Passes out at party”,  
    body:”Wakes up early and cleans house”  
    },{  
    \_id:2,  
    username:”GoodGuyGreg”,  
    title: “Steals you identity”,  
    body:”Raises your credit score”  
    },{  
    \_id:3,  
    username:”GoogGuyGreg”,  
    title: “reports a bug in your code”,  
    body:”Sends you a pull request”  
    },{  
    \_id:4,  
    username:”ScumbagSteve”,  
    title: “Borrows Something”,  
    body:”Sells it”  
    },{  
    \_id:5,  
    username:”ScumbagSteve”,  
    title: “Borrows Everything”,  
    body:”The End”  
    },{  
    \_id:6,  
    username:”ScumbagSteve”,  
    title: “Forks your repo on Github”,  
    body:”Sets to private”.  
    }  
   ])
3. db.comments.insert([  
    {  
    username:”GoodGuyGreg”,  
    comment:”Hope you got a good deal”,  
    post:4  
    },{  
    username:”GoodGuyGreg”,  
    comment:”What’s mine is yours!”,  
    post:5  
    },{  
    username:”GoodGuyGreg”,  
    comment:”Don’t violate the licensing agreement!”,  
    post:6  
    },{  
    username:”ScumbagSteve”,  
    comment:”it still isn’t clean”,  
    post:1  
    },{  
    username:”ScumbagSteve”,  
    comment:”Denied your PR cause I found a hack”,  
    post:3  
    }  
    ])

Querying related collections

1. db.users.find()
2. db.posts.find()
3. db.posts.find({username:”GoodGuyGreg”})
4. db.posts.find({username:”ScumbagSteve”})
5. db.comments.find()
6. db.comments.find({username:”GoodGuyGreg”})
7. db.comments.find({username:”ScumbagSteve”})
8. db.comments.find({post: db.posts.findOne({title:”Reports a bug in your code”}).\_id})