MongoDB Assignment 1

1.Insert the following documents into a movies collection.

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":"2009", "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":"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."})

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

db.movies.insert({"title":"Avatar"})

**Query / Find Documents**

query the movies collection to

1. get all documents

db.movies.find()

1. get all documents with writer set to "Quentin Tarantino"

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

1. get all documents where actors include "Brad Pitt"

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

1. get all documents with franchise set to "The Hobbit"

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

1. get all movies released in the 90s

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

1. 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({"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."}})

1. 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({"title":"The Hobbit: The Desolation of Smaug"},{$set:{"synopsis":"The dwarves, along with Bilbo Baggins and Gandalf the Grey, continue their request to reclaim Erebor, their homeland, from Smaug. Bilbo Baggins is in possession of mysterious and magical ring"}})

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

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"

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

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

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

1. 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/}}]})

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

db.movies.find({$or:[{synopsis:{$regex:"dwarves"}}, {synopsis:{$regex:"hobbit"}}]})

1. 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("5c9f992ae5c2dfe9b3729c00")})

1. delete the movie "Avatar"

db.movies.remove({\_id:ObjectId("5c9f9936e5c2dfe9b3729c01")})

## Relationships

### Insert the following documents into a users collection

db.users.insert({\_id:1,”username”:"GoodGuyGreg", first\_name:"Good Guy", last\_name:"Greg"})

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

### Insert the following documents into a posts collection

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

db.posts.insert({ “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"})

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

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

db.posts.insert({“username”:"ScumbagSteve", title:"Forks your repo on github", body:"Sets to private"})

### Insert the following documents into a comments collection

db.comments.insert({“username”:"GoodGuyGreg", "comment":"Hope you gota good deal!", "post":"6050baf9bad1c44c1088b64d"}

db.comments.insert({“username”:"GoodGuyGreg", "comment":"What's mine is yours!", "post":"6050bb18bad1c44c1088b64e"})

db.comments.insert({“username”:"GoodGuyGreg", "comment":"Don't violate the licensing agreement!", "post":"6050bb43bad1c44c1088b64f"})

db.comments.insert({“username”:"ScumbagSteve", "comment":"It still isn't clean", "post":"6050b7db412e1de6a71b0d0e"})

db.comments.insert({“username”:"ScumbagSteve", "comment":"Denied your PR cause I found a hack", "post":"6050b8b6412e1de6a71b0d10"})

**Querying related collections**

1. find all users

db.users.find().pretty()

1. find all posts

db.posts.find().pretty()

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

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

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

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

1. find all comments

db.comments.find().pretty()

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

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

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

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

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