**MongoDB Basic Assignment Assignment**

Creator: Jitendra Singh Barod

**Query / Find Documents**

**query the movies collection to:**

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({$and:[{year:{$gt:1989}},{year:{$lte:2000}}]})

6.get all movies released before the year 2000 or after 20101

db.movies.find({$or:[{year:{$lte:2000}},{year:{$gt: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."

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."}},{upsert:false,multi:false})

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({title:"The Hobbit The Desolation of smaug"},{$set:{"synopsis":'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"'}},{upsert:true,multi:false})

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

db.movies.findOneAndUpdate({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({$text:{$search:"Bilbo"}}

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

db.movies.find({$text:{$search:"Gandalf"}}

3.find all movies that have a synopsis that contains the word "Bilbo" and not the word "Gandalf"-db.movies.find({$text:{$search:"Bilbo -Gandalf"}})

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

db.movies.find({$text:{$search: "dwarves hobbit"}})

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

db.movies.find({$text:{$search: "gold , dragon"}})

**DELETE DOCUMENTS**

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

db.movies.deleteOne({title:"Pee Wee Hermen's Big Adventure"})

2.delete the movie "Avatar"

db.movies.deleteOne({\_id:"61f023e73ac49c860c01fa07"})

**Relationships**

**Insert the following documents into a users collection**

username : GoodGuyGreg

first\_name : "Good Guy"

last\_name : "Greg"

username : ScumbagSteve

full\_name :

first : "Scumbag"

last : "Steve"

db.users.insertMany([{username:"GoodGuyGreg",first\_name:"Good Guy",last\_name:"Greg"},{username:"Scumbagsteve",full\_name:{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

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

username : ScumbagSteve

title : Borrows something

body : Sells it

username : ScumbagSteve

title : Borrows everything

body : The end

username : ScumbagSteve

title : Forks your repo on github

body : Sets to private

db.posts.insertMany([

{

username: "GoodGuyGreg",

title: "Passes out at party",

body: "Wakes up early and cleans house"

},

{

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"

},

{

username: "ScumbagSteve",

title: "Borrows something",

body: "Sells it"

},

{

username: "ScumbagSteve",

title: "Borrows everything",

body: "The end"

},

{

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"

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"

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

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"

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.insertMany([

{

username: "GoodGuyGreg",

comment: "Hope you got a good deal!",

post: ObjectId("61f1252b06be1ef3f8c349f4")

},

{

username: "GoodGuyGreg",

comment: "what's mine is yours",

post: ObjectId("61f1252b06be1ef3f8c349f5")

},

{

username: "GoodGuyGreg",

comment: "Don't violate the licensing agreement!",

post: ObjectId("61f1252b06be1ef3f8c349f6")

},

{

username: "ScumbagSteve",

comment: "It still isn't clean",

post: ObjectId("61f1252b06be1ef3f8c349f1")

},

{

username: "ScumbagSteve",

comment: "Denied your PR cause I found a hack",

post: ObjectId("61f1252b06be1ef3f8c349f3")

}

]);

**Querying related collections**

1. find all users

db.users.find()

2. find all posts

db.posts.find()

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()

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.posts.aggregate([{$match: { title: 'Reports a bug in your code'

}},{$lookup: {from: 'comments',localField: '\_id',foreignField: 'post',as:

'comments'}}])