**MongoDB Lab Assignments -Day 1**

**MongoDB Exercise in mongo shell**

Connect to a running mongo instance, use a database named **mongo\_practice**. Document all your queries in a javascript file to use as a reference.

**Insert Documents**

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

title : Fight Club

writer : Chuck Palahniuko

year : 1999

actors : [

Brad Pitt

Edward Norton

]

title : Pulp Fiction

writer : Quentin Tarantino

year : 1994

actors : [

John Travolta

Uma Thurman

]

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

title : The Hobbit: The Desolation of Smaug

writer : J.R.R. Tolkein

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.

title : Pee Wee Herman's Big Adventure

title : Avatar

Reference

https://www.tutorialspoint.com/mongodb/mongodb\_insert\_document.htm **Query / Find Documents**

query the **movies** collection to

1. get all documents

Ans:db.movies.find()

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

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

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

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

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

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

5. get all movies released in the 90s

Ans:db.movies.find({$and:[{year:{$gt:1900}},{year:{$lt:2000}}]})

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

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

Reference:

https://www.tutorialspoint.com/mongodb/mongodb\_query\_document.htm **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."

Ans: 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." }})

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."

Ans: 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"

Ans db.movies.update({title:"Pulp Fiction" },{$push:{actors:{ "Samuel L. Jackson"}}})

Reference:

https://www.tutorialspoint.com/mongodb/mongodb\_update\_document.htm **Text Search**

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

Ans: db.movies.createIndex{synopsis:”text”}

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

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

Ans: 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"

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

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

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

Reference: https://www.tutorialspoint.com/mongodb/mongodb\_text\_search.htm **Delete Documents**

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

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

2. delete the movie "Avatar"

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

Reference:

https://www.tutorialspoint.com/mongodb/mongodb\_delete\_document.htm **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"

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

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"

**Querying related collections**

1. find all users

>Ans:db.users.find()

2. find all posts

Ans:db.post.find()

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

Ans:db.post.find({username:"GoodGuyGreg"} )

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

Ans:db.post.find({username:"ScumbagSteve"} )

5. find all comments

Ans:db.comments.find()

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

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

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

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

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

Ans:db.comments.find({post :"Reports a bug in your code" })

References:

https://docs.mongodb.com/manual/reference/method/db.collection.find/ @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@