# 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.

Ans.

Use mongo\_practice

db.movies.insert([{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 engagein a war against an array of combatants and keep theLonely Mountain from falling into the hands of a rising darkness"},

{title : "Pee Wee Herman's Big Adventure"},

{title : "Avatar"}])

Reference https[://ww](http://www.tutorialspoint.com/mongodb/mongodb_insert_document.htm)w.t[utorialspoint.com/mongodb/mongodb\_insert\_document.htm](http://www.tutorialspoint.com/mongodb/mongodb_insert_document.htm)

# Query / Find Documents

query the **movies** collection to

1. get all documents

ans. db.movies.find()

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

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

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

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

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

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

1. get all movies released in the 90s

ans. db.movies.find({year:{$lt:2000}})

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

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

Reference: https[://ww](http://www.tutorialspoint.com/mongodb/mongodb_query_document.htm)w.t[utorialspoint.com/mongodb/mongodb\_query\_document.htm](http://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."}})

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."

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."}})

1. 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[://ww](http://www.tutorialspoint.com/mongodb/mongodb_update_document.htm)w.t[utorialspoint.com/mongodb/mongodb\_update\_document.htm](http://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.movie.createIndex({synopsis:"text"})

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

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

ans.

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

1. find all movies that have a synopsis that contains the word "Bilbo" and not the word "Gandalf"

ans.

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

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

ans

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

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

ans

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

Reference: https[://w](http://www.tutorialspoint.com/mongodb/mongodb_text_search.htm)ww[.t](http://www.tutorialspoint.com/mongodb/mongodb_text_search.htm)u[torialspoint.com/mongodb/mongodb\_text\_search.htm](http://www.tutorialspoint.com/mongodb/mongodb_text_search.htm)

# Delete Documents

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

ans

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

1. delete the movie "Avatar"

ans.

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

Reference: https[://ww](http://www.tutorialspoint.com/mongodb/mongodb_delete_document.htm)w.t[utorialspoint.com/mongodb/mongodb\_delete\_document.htm](http://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"

Ans

db.users.insert({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

Ans

db.posts.insert([{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 : ObjectId("61f102a2f65a66591acfe602"),ObjectId("61f102a2f65a66591acfe602"),ObjectId("61f102a2f65a66591acfe602")

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"

ans

db.comments.insertMany([{username:"GoodGuyGerg",comment:"Hope you got a good deal!",post:ObjectId("61ef7dbbc20642806207cea8")},{ username: "GoodGuyGerg", comment: "What's mine is yours!", post: ObjectId("61ef7dbbc20642806207cea9")},{ username: "GoodGuyGreg", comment: "Don't violate the licensing agreement!", post: ObjectId("61ef7dbbc20642806207ceaa")},{ username: "ScumbagSteve", comment: "It still isn't clean", post: ObjectId("61eef0847cffc5cbdb7b68bd") },{ username: "ScumbagSteve", comment: "Denied your PR cause I found a hack", post: ObjectId("61eef0847cffc5cbdb7b68bf")}])

# Querying related collections

1. find all users

ans

db.users.find()

1. find all posts

ans

db.users.posts()

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

ans

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

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

ans

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

1. find all comments

ans

db.comments.find()

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

ans

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

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

ans

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

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

ans.

db.posts.find({title:"Reports a bug in your code"}) db.comments.find({post: ObjectId("61eef0847cffc5cbdb7b68bf")})References: <https://docs.mongodb.com/manual/reference/method/db.collection.find/>

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@