Name: Gollapudi Manideepak
College: LPU

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
```

Name: Gollapudi Manideepak
College: LPU

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

Solution:

```
Use MongoAssignment
db.Movies.insertMany([{"title": "Fight Club",
"Writer": "Chuck Palahniuko",
"Actors": ["Brad Pitt", "Edward"],
"Year": 1999},
 {"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" }
])
```

Name: Gollapudi Manideepak
College: LPU

Query / Find Documents

query the movies collection to

1. **get all documents**

   Sol: db.Movies.find()
2. **get all documents with writer set to "Quentin Tarantino"**
   Sol: db.Movies.find({Writer: "Quentin Tarantino"})
3. **get all documents where actors include "Brad Pitt"**
   Sol: db.Movies.find({Actors:"Brad Pitt"})
4. **get all documents with franchise set to "The Hobbit"**
   Sol: db.Movies.find({Franchise: "The Hobbit"})
5. **get all movies released in the 90s**
   Sol: db.Movies.find({$or:[{Year:{$lt:2000}},{Year:{$gte:1990}}]})
6. **get all movies released before the year 2000 or after 2010**
   Sol: db.Movies.find(db.Movies.find({$or:[{Year:{$lt:2000}},{Year:{$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."**

   Sol: db.Movies.updateOne({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."**

   Sol: db.Movies.updateOne({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."}})

Name: Gollapudi Manideepak
College: LPU

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

   Sol: .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"**

   Sol: db.Movies.find({$text:{$search:"Bilbo"}})
2. **find all movies that have a synopsis that contains the word "Gandalf"**

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

   Sol: b.Movies.find({$text: {$search: "Bilbo -Gandalf"}})
4. **find all movies that have a synopsis that contains the word "dwarves" or "hobbit"**

   Sol: db.Movies.find({$text: {$search: "dwarves hobbit"}})
5. **find all movies that have a synopsis that contains the word "gold" and "dragon"**

   Sol: db.Movies.find({$text: {$search: "gold dragon"}})

Delete Documents

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

   Sol: db.Movies.remove({title: "Pee Wee Herman's Big Adventure"})
2. **delete the movie "Avatar"**

   Sol: db.movies.remove({title: "Avatar"})

Relationships

```
username : GoodGuyGreg
first_name : "Good Guy"
last_name : "Greg"
username : ScumbagSteve
full_name :
  first : "Scumbag"
  last : "Steve"
```

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

Sol: db.users.insertMany([

{ username: "GoodGuyGreg",

first_name: "Good Guy",

last_name: "Greg" },

{ username: "ScumbagSteve",

Name: Gollapudi Manideepak
College: LPU

```
full_name: { first: "Scumbag", last: "Steve" }
 }
]);
```

Insert the following documents into a posts collection

username : GoodGuyGreg
title:Passesoutatparty
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

Sol:
```
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" }
 ]);
```

Name: Gollapudi Manideepak
College: LPU

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 t

where [post_obj_id] is the ObjectId of the posts document: "Reports a bug in your code"

Sol:
```
db.comments.insertMany([
{ username: "GoodGuyGreg",
comment: "Hope you got a good deal!",
post: ObjectId("61f0384a6f358a5a660abcfe") },
 { username: "GoodGuyGreg",
comment: "Don't violate the licensing agreement!",
post: ObjectId("61f038d36f358a5a660abd00") },
 { username: "GoodGuyGreg",
comment: "Don't violate the licensing agreement!",
 post: ObjectId("61f039146f358a5a660abd02") },
{ username: "ScumbagSteve",
comment: "It still isn't clean",
post: ObjectId("61f036637b5febb0e0a31436") },
{ username: "ScumbagSteve",
comment: "Denied your PR cause I found a hack",
 post: ObjectId("61f037e36f358a5a660abcfd") }
])
```

Name: Gollapudi Manideepak
College: LPU

Querying related collections

1. **find all users**

   Sol: db.users.find()

2. **find all posts**

   Sol: db.posts.find()

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

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

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

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

5. **find all comments**

   Sol: db.comments.find()

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

   Sol: db.comments.find({ username: 'GoodGuyGreg' })

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

   Sol: db.comments.find({ username: 'ScumbagSteve'})

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

   Sol: db.posts.aggregate([{$match: { title: 'Reports a bug in your code' }},{$lookup: {from: 'comments',localField: '_id',foreignField: 'post',as: 'comments'}}])

Thank You!!