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

**Sol-** db.movies.insertMany([ //all the documents given ])

**Query / Find Documents**

query the **movies** collection to

1. get all documents

**Sol-** db.movies.find()

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

**Sol-** db.movies.find({“writer”:” Quentin Tarantino”})

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

**Sol-** db.movies.find({“actors”:”Brad Pitt”})

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

**Sol-** db.movies.find({“franchise”:”The Hobbit”})

1. get all movies released in the 90s

**Sol-** db.movies.find({"year":{$lt:2000}})

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

**Sol-** 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."}})

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({synopsis:{$regex:"Bilbo"}})

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

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

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

**Sol-** 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"

**Sol-** 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"

Sol- db.movies.find({$or:[{synopsis:{$regex:"gold"}},{synopsis:{$regex:"dragon"}}]})

**Delete Documents**

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

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

1. delete the movie "Avatar"

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

**Relationships**

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

username : GoodGuyGreg

first\_name : "Good Guy"

last\_name : "Greg"

Sol- //Creating a collection

db.createCollection("users")

// now inserting the given document

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

username : ScumbagSteve

full\_name :

first : "Scumbag"

last : "Steve"

Sol- db.users.insert({username:"ScumbagSteve", fullname:{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

**Sol-** //creating a collection

Db.createCollection(“posts”)

// now inserting the given data

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

username : GoodGuyGreg

title : Steals your identity

body : Raises your credit score

Sol- db.posts.insert({ 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

Sol- db.posts.insert({username:"GoodGuyGreg", title:"Reports a bug in your code", body:"Sends you a pull request"})

username : ScumbagSteve

title : Borrows something

body : Sells it

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

username : ScumbagSteve

title : Borrows everything

body : The end

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

username : ScumbagSteve

title : Forks your repo on github

body : Sets to private

Sol- db.posts.insert({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"

Sol- db.comments.insert({username:"GoodGuyGreg", comment:"Hope you got a good deal!", post:ObjectId("60059616c4595ddf8d71c186")})

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"

Sol- db.comments.insert({username:"GoodGuyGreg", comment:"What's mine is yours!", post:ObjectId("60059647c4595ddf8d71c187")})

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

Sol- db.comments.insert({username:"GoodGuyGreg", comment:"Don't violate the licensing agreement!", post:ObjectId("60059658c4595ddf8d71c188")})

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"

Sol- db.comments.insert({username:"ScumbagSteve", comment:"It still isn't clean", post:ObjectId("60059552c4595ddf8d71c183")})

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"

Sol- db.comments.insert({username:"ScumbagSteve", comment:"Denied your PR cause I found a hack", post:ObjectId("600595f0c4595ddf8d71c185")})

**Querying related collections**

1. find all users

Sol- db.users.find()

1. find all posts

Sol- db.posts.find()

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

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

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

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

1. find all comments

Sol- db.comments.find()

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

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

1. 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"