1)Insert the following documents into a movies collection.

Ans) db.movies.insertMany([{"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"}

])

**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"})

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}}]})

**Update Document**

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"}, {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"}, {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"}})

**Text Search**

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

ans) 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 -Gandalf"}})

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"}})

**Delete Documents**

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

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

1. delete the movie "Avatar"

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

**Relationships**

1)Insert the following documents into a users collection

Ans) db.users.insertMany([{

"username" : "GoodGuyGreg",

"first\_name" : "Good Guy",

"last\_name" : "Greg",

"username" : "ScumbagSteve",

"full\_name" :"steve",

"first" : "Scumbag",

"last" : "Steve",

}])

2) Insert the following documents into a posts collection

Ans)

|  |
| --- |
| db.posts.insert([ |
|  | { |
|  | \_id: 1, |
|  | username : "GoodGuyGreg", |
|  | title : "Passes out at party", |
|  | body : "Wakes up early and cleans house", |
|  | }, |
|  |  |
|  | { |
|  | \_id: 2, |
|  | username : "GoodGuyGreg", |
|  | title : "Steals your identity", |
|  | body : "Raises your credit score", |
|  | }, |
|  |  |
|  | { |
|  | \_id: 3, |
|  | username : "GoodGuyGreg", |
|  | title : "Reports a bug in your code", |
|  | body : "Sends you a Pull Request", |
|  | }, |
|  |  |
|  | { |
|  | \_id: 4, |
|  | username : "ScumbagSteve", |
|  | title : "Borrows something", |
|  | body : "Sells it", |
|  | }, |
|  |  |
|  | { |
|  | \_id: 5, |
|  | username : "ScumbagSteve", |
|  | title : "Borrows everything", |
|  | body : "The end", |
|  | }, |
|  |  |
|  | { |
|  | \_id: 6, |
|  | username : "ScumbagSteve", |
|  | title : "Forks your repo on github", |
|  | body : "Sets to private", |
|  | }, |
|  | ]) |

Insert the following documents into a comments collection

Ans)

|  |
| --- |
| db.comments.insert([ |
|  | { |
|  | username : "GoodGuyGreg", |
|  | comment : "Hope you got a good deal!", |
|  | post : 4, |
|  | }, |
|  | { |
|  | username : "GoodGuyGreg", |
|  | comment : "What's mine is yours!", |
|  | post : 5, |
|  | }, |
|  | { |
|  | username : "GoodGuyGreg", |
|  | comment : "Don't violate the licensing agreement!", |
|  | post : 6, |
|  | }, |
|  | { |
|  | username : "ScumbagSteve", |
|  | comment : "It still isn't clean", |
|  | post : 1, |
|  | }, |
|  | { |
|  | username : "ScumbagSteve", |
|  | comment : "Denied your PR cause I found a hack", |
|  | post : 3 |
|  | }, |
|  | ]) |

**Querying related collections**

1. find all users

ans) db.users.find()

1. find all posts

ans) db.posts.find()

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.comments.find({post: db.posts.findOne({title: "Reports a bug in your code"}).\_id})