Basic Assignment :

Connect to a running mongo instance, use a database named mongo\_practice.

* use mongo\_practise

Insert the following documents into a movies collection.

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"

    },

    {

        title : "Pee Wee Herman's Big Adventure"

    },

    {

        title : "Avatar"

    }

])

Query / Find Documents

query the movies collection to

1. get all documents

* db.Movies.find().pretty()

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

* db.Movies.find({writer: "Quentin Tarantino"}).pretty()

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

* db.Movies.find({actors: "Brad Pitt"}).pretty()

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

* db.Movies.find({franchise: "The Hobbit"}).pretty()

1. get all movies released in the 90s

* db.Movies.find({$and: [{year: {$gt: "1900"}}, {year: {$lt: "2000"}}]})

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

* 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."

-> 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."

-> 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"

* 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"

* db.Movies.find({synopsis: /Bilbo./}).pretty()

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

* db.Movies.find({synopsis: /.Gandalf./}).pretty()

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

* db.Movies.find({$and: [{synopsis: /Bilbo/}, {synopsis: {$not: /Gandalf/}}]}).pretty()

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

* db.Movies.find({$or: [{synopsis: /dwarves/}, {synopsis: /hobbit/}]}).pretty()

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

* db.Movies.find({$and: [{synopsis: /gold/}, {synopsis:  /dragon/}]}).pretty()

Delete Documents

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

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

1. delete the movie "Avatar"

* db.Movies.remove({title: "Avatar"})

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"

db.users.insertOne([

    {

        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

db.posts.insertMany([

    {

        \_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

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"

db.comments.insertMany([

    {

        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

* db.users.find({}).pretty()

1. find all posts

* db.posts.find({}).pretty()

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

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

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

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

1. find all comments

* db.comments.find({}).pretty()

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

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

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

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

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

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