**MongoDB Assignment- 01**

**Connect to a running mongo instance, use a database named mongo practice.**

Show dbs;

Use practice;

db.createCollection(“movies”);

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

db.movies.insert([

{

title : "Fight Club",

writer : "Chuck Palahniuk",

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**

1. **get all documents**

db.movies.find({}).pretty()

**2. get all documents with writer set to "Quentin Tarantino"**

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

**3. get all documents where actors include "Brad Pitt"**

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

**4. get all documents with nise set to "The Hobbit"**

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

**5. get all movies released in the 90s**

db.movies.find({year: {$gte: 1990, $lte: 1999}}).pretty()

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

db.movies.find({$or:[{year:{$lte:2000}}, {year:{$gte:2010}}]}).pretty()

**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."}})

**3. 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()

**2 find all movies that have a synopsis that contains the word "Gandalf"**

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

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

db.movies.find({$and: [{synopsis: /.Bilbo./}, {synopsis: /.^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({$or: [{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"})

**Relationship:**

db.createCollection(“users”);

db.createCollection(“posts”);

db.createCollection(“comments”);

**Insert the following document into a user collection.**

db.users.insert([

{

username : "GoodGuyGreg",

first\_name : "Good Guy",

last\_name : "Greg"

},

{

username : "ScumbagSteve",

full\_name : {

first : "Scumbag",

last : "Steve",

}

}

])

**Insert the following document into a posts collection.**

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 document into a comments collection.**

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

},

])

**1. find all users**

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

**2. find all posts**

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

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

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

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

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

**5. find all comments**

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

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

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

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

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

**8. 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})