**MongoDB Lab Assignments –Day 1**

**MongoDB Exercise in mongo shell**

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

**Query / Find Documents**

Query the **movies** collection to

1. **Get all documents**

use mongo\_practice

db.movies.find()

1. **Get all documents with writer set to “Quentin Tarantino.**

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

1. **Get all documents where actors include “Brad Pitt”.**

db.movies.find({actors: “Brad Pitt”})

1. **Get all documents with franchise set to “The Hobbit”.**

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

1. **Get all movies released in the 90s.**

db.movies.find({year: {$gt: 1990, $lt: 2000}})

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

db.movies.find({$or: [{year:{$gt:2020}}, {year:{$lt: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({\_id:ObjectId("5c9f98e5e5c2dfe9b3729bfe")}, {$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."}})

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.”**

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”, {$set: {actors: “Samuel L. Jackson”}})

**Text Search**

1. **Find all movies that have a synopsis that contains the word “Bilbo”.**

db.movies.find({synopsis: {$regex: “Bilbo”}})

1. **Find all movies that have a synopsis that contains the world “Gandalf”.**

db.movies.find({synopsis:{$regex: “Gandalf”}})

1. **Find all movies that have a synopsis that contains the word “Bilbo” and not the word “Gandalf”.**

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”.**

db.movies.find({$or:[{synopsis:{$regex:”dwarves”}}, {synopsis:{$regex:”hobbit”}}]})

1. **Find all movies that have a synopsis that contain the word “gold” and “dragon”.**

db.movies.find({$and:[{synopsis:{$regex:”gold”}}, {synopsis:{$regex:”dragon”}}]})

**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”})

**Query related collections:**

1. **Find all users**

db.users.find()

1. **Find all posts**

db.posts.find()

1. **Find all posts that were authored by “GoodGuyGreg”.**

db.posts.find({username: “GoodGuyGreg”})

1. **Find all posts that were authored by “ScumbagSteve”.**

db.posts.find({username: “ScumbagSteve”})

1. **Find all comments**

db.comments.find()

1. **Find all comments that were authored by “GoodGuyGreg”.**

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

1. **Find all comments that were authored by “ScumbagSteve”.**

db.comments.find({username: “ScumbagSteve”})

1. **Find all comments belonging to the post “Reports a bug in your code”.**

db.comments.find(post: ObjectId(“"615947c5eaf0b0eb8c8be182"”)})