1st Assignment:

MongoDB Exercise in mongo shell

> use mongo\_practice

Insert Documents:

> db.movies.insertOne({title:'Fight Club',writer:'Chuck Palahniuko',year:1999,actors:['Brad Pitt','Edward Norton']})

> db.movies.find()

{ "\_id" : ObjectId("60df0803d1fd6b476b03f0f6"), "title" : "Fight Club", "writer" : "Chuck Palahniuko", "year" : 1999, "actors" : [ "Brad Pitt", "Edward Norton" ] }

> db.movies.insertMany([{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']}])

> db.movies.insertOne({title:'The Hobbit:An Unexpected Journey',writer:'J.R.R. Tolkein',year:2012,franchise:'The Hobbit'})

> db.movies.insertMany([{title:'The Hobbit:The Desolation of Smaug',writer:'J.R.R. Tolkein',year:2013,franchise:'The Hobbit'},{title:'The Hobbit:The Battle of 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'}])

1. Get All Documents(Query)

> db.movies.find()

{ "\_id" : ObjectId("60df0803d1fd6b476b03f0f6"), "title" : "Fight Club", "writer" : "Chuck Palahniuko", "year" : 1999, "actors" : [ "Brad Pitt", "Edward Norton" ] }

{ "\_id" : ObjectId("60df0d06d1fd6b476b03f0f7"), "title" : "Pulp Fiction", "writer" : "Quentin Tarantino", "year" : 1994, "actors" : [ "John Travolta", "Uma Thurman" ] }

{ "\_id" : ObjectId("60df0d06d1fd6b476b03f0f8"), "title" : "Inglorious Basterds", "writer" : "Quentin Tarantino", "year" : 2009, "actors" : [ "Brad Pitt", "Diane Kruger", "Eli Roth" ] }

{ "\_id" : ObjectId("60df1087d1fd6b476b03f0f9"), "title" : "The Hobbit:An Unexpected Journey", "writer" : "J.R.R. Tolkein", "year" : 2012, "franchise" : "The Hobbit" }

{ "\_id" : ObjectId("60df1a93d1fd6b476b03f102"), "title" : "The Hobbit:The Desolation of Smaug", "writer" : "J.R.R. Tolkein", "year" : 2013, "franchise" : "The Hobbit" }

{ "\_id" : ObjectId("60df1a93d1fd6b476b03f103"), "title" : "The Hobbit:The Battle of 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." }

{ "\_id" : ObjectId("60df1a93d1fd6b476b03f104"), "title" : "Pee Wee Herman's Big Adventure" }

{ "\_id" : ObjectId("60df1a93d1fd6b476b03f105"), "title" : "Avatar" }

2. Get all documents with writer set to “Quentin Tarantino”

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

{ "\_id" : ObjectId("60df0d06d1fd6b476b03f0f7"), "title" : "Pulp Fiction", "writer" : "Quentin Tarantino", "year" : 1994, "actors" : [ "John Travolta", "Uma Thurman" ] }

{ "\_id" : ObjectId("60df0d06d1fd6b476b03f0f8"), "title" : "Inglorious Basterds", "writer" : "Quentin Tarantino", "year" : 2009, "actors" : [ "Brad Pitt", "Diane Kruger", "Eli Roth" ] }

3. Get all documents where actors include “Brad Pitt”

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

{ "\_id" : ObjectId("60df0803d1fd6b476b03f0f6"), "title" : "Fight Club", "writer" : "Chuck Palahniuko", "year" : 1999, "actors" : [ "Brad Pitt", "Edward Norton" ] }

{ "\_id" : ObjectId("60df0d06d1fd6b476b03f0f8"), "title" : "Inglorious Basterds", "writer" : "Quentin Tarantino", "year" : 2009, "actors" : [ "Brad Pitt", "Diane Kruger", "Eli Roth" ] }

4. Get all documents with franchise “The Hobbit”

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

{ "\_id" : ObjectId("60df1087d1fd6b476b03f0f9"), "title" : "The Hobbit:An Unexpected Journey", "writer" : "J.R.R. Tolkein", "year" : 2012, "franchise" : "The Hobbit" }

{ "\_id" : ObjectId("60df1a93d1fd6b476b03f102"), "title" : "The Hobbit:The Desolation of Smaug", "writer" : "J.R.R. Tolkein", "year" : 2013, "franchise" : "The Hobbit" }

{ "\_id" : ObjectId("60df1a93d1fd6b476b03f103"), "title" : "The Hobbit:The Battle of 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." }

5. Get all movies released in the 90s.

> db.movies.find({year:{$lt:2000}})

{ "\_id" : ObjectId("60df0803d1fd6b476b03f0f6"), "title" : "Fight Club", "writer" : "Chuck Palahniuko", "year" : 1999, "actors" : [ "Brad Pitt", "Edward Norton" ] }

{ "\_id" : ObjectId("60df0d06d1fd6b476b03f0f7"), "title" : "Pulp Fiction", "writer" : "Quentin Tarantino", "year" : 1994, "actors" : [ "John Travolta", "Uma Thurman" ] }

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

> db.movies.find({$or: [{year:{$lt:2000}},{year:{$gt:2010}}]})

{ "\_id" : ObjectId("60df0803d1fd6b476b03f0f6"), "title" : "Fight Club", "writer" : "Chuck Palahniuko", "year" : 1999, "actors" : [ "Brad Pitt", "Edward Norton" ] }

{ "\_id" : ObjectId("60df0d06d1fd6b476b03f0f7"), "title" : "Pulp Fiction", "writer" : "Quentin Tarantino", "year" : 1994, "actors" : [ "John Travolta", "Uma Thurman", "Samuel L. Jackson" ] }

{ "\_id" : ObjectId("60df1087d1fd6b476b03f0f9"), "title" : "The Hobbit:An Unexpected Journey", "writer" : "J.R.R. Tolkein", "year" : 2012, "franchise" : "The Hobbit", "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." }

{ "\_id" : ObjectId("60df1a93d1fd6b476b03f103"), "title" : "The Hobbit:The Battle of 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." }

{ "\_id" : ObjectId("60df1a93d1fd6b476b03f102"), "title" : "The Hobbit:The Desolation of Smaug", "writer" : "J.R.R. Tolkein", "year" : 2013, "franchise" : "The Hobbit", "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." }

Update Document:

1.

> 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."}})

WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

2.

MongoDB Enterprise > 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."}})

WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

3.

MongoDB Enterprise > db.movies.update({title: "Pulp Fiction"},{$push: {actors: "Samuel L. Jackson"}})

WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

Text Search:

1.

> db.movies.find({synopsis:{$regex:"Bilbo"}})

{ "\_id" : ObjectId("60df1087d1fd6b476b03f0f9"), "title" : "The Hobbit:An Unexpected Journey", "writer" : "J.R.R. Tolkein", "year" : 2012, "franchise" : "The Hobbit", "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." }

{ "\_id" : ObjectId("60df1a93d1fd6b476b03f103"), "title" : "The Hobbit:The Battle of 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." }

{ "\_id" : ObjectId("60df1a93d1fd6b476b03f102"), "title" : "The Hobbit:The Desolation of Smaug", "writer" : "J.R.R. Tolkein", "year" : 2013, "franchise" : "The Hobbit", "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." }

2.

> db.movies.find({synopsis:{$regex:"Gandalf"}})

{ "\_id" : ObjectId("60df1a93d1fd6b476b03f102"), "title" : "The Hobbit:The Desolation of Smaug", "writer" : "J.R.R. Tolkein", "year" : 2013, "franchise" : "The Hobbit", "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.

> db.movies.find({$and:[{synopsis:{$regex:"Bilbo"}}, {synopsis:{$not:/Gandalf/}}]})

{ "\_id" : ObjectId("60df1087d1fd6b476b03f0f9"), "title" : "The Hobbit:An Unexpected Journey", "writer" : "J.R.R. Tolkein", "year" : 2012, "franchise" : "The Hobbit", "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." }

{ "\_id" : ObjectId("60df1a93d1fd6b476b03f103"), "title" : "The Hobbit:The Battle of 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." }

4.

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

{ "\_id" : ObjectId("60df1087d1fd6b476b03f0f9"), "title" : "The Hobbit:An Unexpected Journey", "writer" : "J.R.R. Tolkein", "year" : 2012, "franchise" : "The Hobbit", "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." }

{ "\_id" : ObjectId("60df1a93d1fd6b476b03f102"), "title" : "The Hobbit:The Desolation of Smaug", "writer" : "J.R.R. Tolkein", "year" : 2013, "franchise" : "The Hobbit", "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." }

5.

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

{ "\_id" : ObjectId("60df1087d1fd6b476b03f0f9"), "title" : "The Hobbit:An Unexpected Journey", "writer" : "J.R.R. Tolkein", "year" : 2012, "franchise" : "The Hobbit", "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." }

Delete Document:

1.

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

WriteResult({ "nRemoved" : 1 })

2.

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

WriteResult({ "nRemoved" : 1 })

Relationships

Users:

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

>db.users.insertOne({username:"ScumbagSteve",full\_name:{first:"Scumbag",last:"Steve"}})

Posts:

> db.posts.insertOne({username:"GoodGuyGreg",title:"Passes out at party",body:"Wakes up early and cleans house"})

> db.posts.insertMany([{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"}])

Comments:

> db.comments

mongo\_practice.comments

> db.comments.insert({username:"GoodGuyGreg",comment:"Hope you got a good deal!",post:ObjectId("60e005db94de16fe49e16758")})

WriteResult({ "nInserted" : 1 })

> db.comments.insert({username:"GoodGuyGreg",comment:"What's mine is yours!",post:ObjectId("60e005db94de16fe49e16759")})

WriteResult({ "nInserted" : 1 })

> db.comments.insert({username:"GoodGuyGreg",comment:"Don't violate the licensing agreement!",post:ObjectId("60e005db94de16fe49e1675a")})

WriteResult({ "nInserted" : 1 })

> db.comments.insert({username:"ScumbagSteve",comment:"It still isn't clean",post:ObjectId("60e001d494de16fe49e16755")})

WriteResult({ "nInserted" : 1 })

> db.comments.insert({username:"ScumbagSteve",comment:"Denied your PR cause I found a hack",post:ObjectId("60e005db94de16fe49e16757")})

WriteResult({ "nInserted" : 1 })

Querying related collections:

1) Find all users

> db.users.find()

{ "\_id" : ObjectId("60dffdcc94de16fe49e16753"), "username" : "GoodGuyGreg", "first\_name" : "Good Guy", "last\_name" : "Greg" }

{ "\_id" : ObjectId("60e000c294de16fe49e16754"), "username" : "ScumbagSteve", "full\_name" : { "first" : "Scumbag", "last" : "Steve" } }

2)

> db.posts.find()

{ "\_id" : ObjectId("60e001d494de16fe49e16755"), "username" : "GoodGuyGreg", "title" : "Passes out at party", "body" : "Wakes up early and cleans house" }

{ "\_id" : ObjectId("60e005db94de16fe49e16759"), "username" : "ScumbagSteve", "title" : "Borrows everything", "body" : "The end" }

{ "\_id" : ObjectId("60e005db94de16fe49e16758"), "username" : "ScumbagSteve", "title" : "Borrows something", "body" : "Sells it" }

{ "\_id" : ObjectId("60e005db94de16fe49e16757"), "username" : "GoodGuyGreg", "title" : "Reports a bug in your code", "body" : "Sends you a Pull Request" }

{ "\_id" : ObjectId("60e005db94de16fe49e16756"), "username" : "GoodGuyGreg", "title" : "Steals your identity", "body" : "Raises your credit score" }

{ "\_id" : ObjectId("60e005db94de16fe49e1675a"), "username" : "ScumbagSteve", "title" : "Forks your repo on github", "body" : "Sets to private" }

3)

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

{ "\_id" : ObjectId("60e001d494de16fe49e16755"), "username" : "GoodGuyGreg", "title" : "Passes out at party", "body" : "Wakes up early and cleans house" }

{ "\_id" : ObjectId("60e005db94de16fe49e16757"), "username" : "GoodGuyGreg", "title" : "Reports a bug in your code", "body" : "Sends you a Pull Request" }

{ "\_id" : ObjectId("60e005db94de16fe49e16756"), "username" : "GoodGuyGreg", "title" : "Steals your identity", "body" : "Raises your credit score" }

4)

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

{ "\_id" : ObjectId("60e005db94de16fe49e16759"), "username" : "ScumbagSteve", "title" : "Borrows everything", "body" : "The end" }

{ "\_id" : ObjectId("60e005db94de16fe49e16758"), "username" : "ScumbagSteve", "title" : "Borrows something", "body" : "Sells it" }

{ "\_id" : ObjectId("60e005db94de16fe49e1675a"), "username" : "ScumbagSteve", "title" : "Forks your repo on github", "body" : "Sets to private" }

5)

> db.comments.find()

{ "\_id" : ObjectId("60e01c1b94de16fe49e1675b"), "username" : "GoodGuyGreg", "comment" : "Hope you got a good deal!", "post" : ObjectId("60e005db94de16fe49e16758") }

{ "\_id" : ObjectId("60e01dd094de16fe49e1675c"), "username" : "GoodGuyGreg", "comment" : "What's mine is yours!", "post" : ObjectId("60e005db94de16fe49e16759") }

{ "\_id" : ObjectId("60e01e9394de16fe49e1675d"), "username" : "GoodGuyGreg", "comment" : "Don't violate the licensing agreement!", "post" : ObjectId("60e005db94de16fe49e1675a") }

{ "\_id" : ObjectId("60e01f1e94de16fe49e1675e"), "username" : "ScumbagSteve", "comment" : "It still isn't clean", "post" : ObjectId("60e001d494de16fe49e16755") }

{ "\_id" : ObjectId("60e01ffd94de16fe49e1675f"), "username" : "ScumbagSteve", "comment" : "Denied your PR cause I found a hack", "post" : ObjectId("60e005db94de16fe49e16757") }

6)

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

{ "\_id" : ObjectId("60e01c1b94de16fe49e1675b"), "username" : "GoodGuyGreg", "comment" : "Hope you got a good deal!", "post" : ObjectId("60e005db94de16fe49e16758") }

{ "\_id" : ObjectId("60e01dd094de16fe49e1675c"), "username" : "GoodGuyGreg", "comment" : "What's mine is yours!", "post" : ObjectId("60e005db94de16fe49e16759") }

{ "\_id" : ObjectId("60e01e9394de16fe49e1675d"), "username" : "GoodGuyGreg", "comment" : "Don't violate the licensing agreement!", "post" : ObjectId("60e005db94de16fe49e1675a") }

7)

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

{ "\_id" : ObjectId("60e01f1e94de16fe49e1675e"), "username" : "ScumbagSteve", "comment" : "It still isn't clean", "post" : ObjectId("60e001d494de16fe49e16755") }

{ "\_id" : ObjectId("60e01ffd94de16fe49e1675f"), "username" : "ScumbagSteve", "comment" : "Denied your PR cause I found a hack", "post" : ObjectId("60e005db94de16fe49e16757") }

8)

> db.comments.find({post:ObjectId("60e005db94de16fe49e16757")})

{ "\_id" : ObjectId("60e01ffd94de16fe49e1675f"), "username" : "ScumbagSteve", "comment" : "Denied your PR cause I found a hack", "post" : ObjectId("60e005db94de16fe49e16757") }