**Inserting data into Collections**

> use mongo\_practice

switched to db mongo\_practice

> db.movies.insert(

... {

... "title" : "Fight Club",

... "writer" : "Chuck Palahniuko",

... "year" : 1999,

... "actors" : ["Brad Pitt","Edward Norton"]

... })

WriteResult({ "nInserted" : 1 })

> db.movies.insertMany([

... {

... "title":"Pulp Fiction",

... "writer":"Quentin Tarantino",

... "year":1994,

... "actors":["John Travolta","Uma Thurman"]

... },{

... "title":"Inglorius Bastards",

... "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 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"

... }])

{

"acknowledged" : true,

"insertedIds" : [

ObjectId("5f6165114742e63f7f1e42a0"),

ObjectId("5f6165114742e63f7f1e42a1"),

ObjectId("5f6165114742e63f7f1e42a2"),

ObjectId("5f6165114742e63f7f1e42a3"),

ObjectId("5f6165114742e63f7f1e42a4"),

ObjectId("5f6165114742e63f7f1e42a5"),

ObjectId("5f6165114742e63f7f1e42a6")

]

}

Query / Find Documents:

1. Get all documents

> db.movies.find()

{ "\_id" : ObjectId("5f615e184742e63f7f1e4297"), "title" : "Fight Club", "writer" : "Chuck Palahniuko", "year" : "1999", "actors" : [ "Brad Pitt", "Edward Norton" ] }

{ "\_id" : ObjectId("5f6160bb4742e63f7f1e4298"), "title" : "Pulp Fiction", "writer" : "Quentin Tarantino", "year" : "1994", "actors" : [ "John Travolta", "Uma Thurman" ] }

{ "\_id" : ObjectId("5f6160bb4742e63f7f1e4299"), "title" : "Inglorius Bastards", "writer" : "Quentin Tarantino", "year" : "2009", "actors" : [ "Brad Pitt", "Diane Kruger", "Eli Roth" ] }

{ "\_id" : ObjectId("5f6160bb4742e63f7f1e429a"), "title" : "The Hobbit: An Unexpected Journey", "writer" : "J.R.R. Tolkein", "year" : "2012", "franchise" : "The Hobbit" }

{ "\_id" : ObjectId("5f6160bb4742e63f7f1e429b"), "title" : "The Hobbit: The Desolation of Smaug", "writer" : "J.R.R. Tolkein", "year" : "2013", "franchise" : "The Hobbit" }

{ "\_id" : ObjectId("5f6160bb4742e63f7f1e429c"), "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("5f6160bb4742e63f7f1e429d"), "title" : "Pee Wee Herman's Big Adventure" }

{ "\_id" : ObjectId("5f6160bb4742e63f7f1e429e"), "title" : "Avatar" }

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

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

{ "\_id" : ObjectId("5f6160bb4742e63f7f1e4298"), "title" : "Pulp Fiction", "writer" : "Quentin Tarantino", "year" : "1994", "actors" : [ "John Travolta", "Uma Thurman" ] }

{ "\_id" : ObjectId("5f6160bb4742e63f7f1e4299"), "title" : "Inglorius Bastards", "writer" : "Quentin Tarantino", "year" : "2009", "actors" : [ "Brad Pitt", "Diane Kruger", "Eli Roth" ] }

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

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

{

"\_id" : ObjectId("5f615e184742e63f7f1e4297"),

"title" : "Fight Club",

"writer" : "Chuck Palahniuko",

"year" : "1999",

"actors" : [

"Brad Pitt",

"Edward Norton"

]

}

{

"\_id" : ObjectId("5f6160bb4742e63f7f1e4299"),

"title" : "Inglorius Bastards",

"writer" : "Quentin Tarantino",

"year" : "2009",

"actors" : [

"Brad Pitt",

"Diane Kruger",

"Eli Roth"

]

}

1. Get all documents where franchise is set to “The Hobbit”

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

{

"\_id" : ObjectId("5f6160bb4742e63f7f1e429a"),

"title" : "The Hobbit: An Unexpected Journey",

"writer" : "J.R.R. Tolkein",

"year" : "2012",

"franchise" : "The Hobbit"

}

{

"\_id" : ObjectId("5f6160bb4742e63f7f1e429b"),

"title" : "The Hobbit: The Desolation of Smaug",

"writer" : "J.R.R. Tolkein",

"year" : "2013",

"franchise" : "The Hobbit"

}

{

"\_id" : ObjectId("5f6160bb4742e63f7f1e429c"),

"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."

}

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

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

{

"\_id" : ObjectId("5f6164f54742e63f7f1e429f"),

"title" : "Fight Club",

"writer" : "Chuck Palahniuko",

"year" : 1999,

"actors" : [

"Brad Pitt",

"Edward Norton"

]

}

{

"\_id" : ObjectId("5f6165114742e63f7f1e42a0"),

"title" : "Pulp Fiction",

"writer" : "Quentin Tarantino",

"year" : 1994,

"actors" : [

"John Travolta",

"Uma Thurman"

]

}

1. Get all the movies release before 2000 or after 2010

> db.movies.find(

... {'$or': [

... {"year":{'$lt':2000}},

... {"year":{'$gt':2010}}

... ]}

... ).pretty()

{

"\_id" : ObjectId("5f6164f54742e63f7f1e429f"),

"title" : "Fight Club",

"writer" : "Chuck Palahniuko",

"year" : 1999,

"actors" : [

"Brad Pitt",

"Edward Norton"

]

}

{

"\_id" : ObjectId("5f6165114742e63f7f1e42a0"),

"title" : "Pulp Fiction",

"writer" : "Quentin Tarantino",

"year" : 1994,

"actors" : [

"John Travolta",

"Uma Thurman"

]

}

{

"\_id" : ObjectId("5f6165114742e63f7f1e42a2"),

"title" : "The Hobbit: An Unexpected Journey",

"writer" : "J.R.R. Tolkein",

"year" : 2012,

"franchise" : "The Hobbit"

}

{

"\_id" : ObjectId("5f6165114742e63f7f1e42a3"),

"title" : "The Hobbit: The Desolation of Smaug",

"writer" : "J.R.R. Tolkein",

"year" : 2013,

"franchise" : "The Hobbit"

}

{

"\_id" : ObjectId("5f6165114742e63f7f1e42a4"),

"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."

}

Update Documents:

1. Add a Synopsis to “The Hobbit: An Unexpected Journey”

> db.movies.updateOne(

... {"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."}}

... )

{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }

1. add a synopsis to "The Hobbit: The Desolation of Smaug"

db.movies.updateOne(

... {"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."}}

... )

{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }

1. add an actor named "Samuel L. Jackson" to the movie "Pulp Fiction"

> db.movies.updateOne(

... {"title":"Pulp Fiction"},

... {$addToSet: { actors: "Samuel L. Jackson" }}

... )

{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }

**Text Search**

> db.movies.createIndex({synopsis:"text"})

{

"createdCollectionAutomatically" : false,

"numIndexesBefore" : 1,

"numIndexesAfter" : 2,

"ok" : 1

}

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

> db.movies.find({$text:{$search:"Bilbo"}}).pretty()

{

"\_id" : ObjectId("5f6165114742e63f7f1e42a3"),

"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."

}

{

"\_id" : ObjectId("5f6165114742e63f7f1e42a4"),

"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("5f6165114742e63f7f1e42a2"),

"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."

}

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

> db.movies.find({$text:{$search:"Gandalf"}}).pretty()

{

"\_id" : ObjectId("5f6165114742e63f7f1e42a3"),

"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."

}

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

> db.movies.find({$text:{$search:'Bilbo -Gandalf'}}).pretty()

{

"\_id" : ObjectId("5f6165114742e63f7f1e42a4"),

"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("5f6165114742e63f7f1e42a2"),

"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."

}

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

> db.movies.find({$text:{$search:'dwarves, hobbit'}}).pretty()

{

"\_id" : ObjectId("5f6165114742e63f7f1e42a2"),

"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("5f6165114742e63f7f1e42a3"),

"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."

}

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

> db.movies.find({$text:{$search:'gold, dragon'}}).pretty()

{

"\_id" : ObjectId("5f6165114742e63f7f1e42a2"),

"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 Documents**

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

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

WriteResult({ "nRemoved" : 0 })

2. delete the movie "Avatar"

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

WriteResult({ "nRemoved" : 0 })

**Relationships:**

Insert the following documents into a **users** collection

> db.users.insertMany([

... {

... "username" : "GoodGuyGreg",

... "first\_name" : "Good Guy",

... "last\_name" : "Greg"

... },{

...

... "username" : "ScumbagSteve",

... "full\_name" : {

... "first" : "Scumbag",

... "last" : "Steve"

... }

... }])

{

"acknowledged" : true,

"insertedIds" : [

ObjectId("5f622b20d333e3be77ecc31f"),

ObjectId("5f622b20d333e3be77ecc320")

]

}

Insert the following documents into a **users** collection

> db.posts.insertMany([

... {

... "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"

... }])

{

"acknowledged" : true,

"insertedIds" : [

ObjectId("5f622c58d333e3be77ecc321"),

ObjectId("5f622c58d333e3be77ecc322"),

ObjectId("5f622c58d333e3be77ecc323"),

ObjectId("5f622c58d333e3be77ecc324"),

ObjectId("5f622c58d333e3be77ecc325"),

ObjectId("5f622c58d333e3be77ecc326")

]

}

Insert the following documents into a **comments** collection

> db.comments.insertMany([

... {

... "username" : "GoodGuyGreg",

... "comment" : "Hope you got a good deal!",

... "post" : "5f622c58d333e3be77ecc324"

... },{

... "username" : "GoodGuyGreg",

... "comment" : "What's mine is yours!",

... "post" : "5f622c58d333e3be77ecc325"

... },{

... "username" : "GoodGuyGreg",

... "comment" : "Don't violate the licensing agreement!",

... "post" : "5f622c58d333e3be77ecc326"

... },{

... "username" : "ScumbagSteve",

... "comment" : "It still isn't clean",

... "post" : "5f622c58d333e3be77ecc321"

... }

... ])

{

"acknowledged" : true,

"insertedIds" : [

ObjectId("5f623f94d333e3be77ecc327"),

ObjectId("5f623f94d333e3be77ecc328"),

ObjectId("5f623f94d333e3be77ecc329"),

ObjectId("5f623f94d333e3be77ecc32a")

]

}

**Querying related collections**

1. find all users

> db.users.find()

{ "\_id" : ObjectId("5f622b20d333e3be77ecc31f"), "username" : "GoodGuyGreg", "first\_name" : "Good Guy", "last\_name" : "Greg" }

{ "\_id" : ObjectId("5f622b20d333e3be77ecc320"), "username" : "ScumbagSteve", "full\_name" : { "first" : "Scumbag", "last" : "Steve" } }

1. find all posts

> db.posts.find()

{ "\_id" : ObjectId("5f622c58d333e3be77ecc321"), "username" : "GoodGuyGreg", "title" : "Passes out at party", "body" : "Wakes up early and cleans house" }

{ "\_id" : ObjectId("5f622c58d333e3be77ecc322"), "username" : "GoodGuyGreg", "title" : "Steals your identity", "body" : "Raises your credit score" }

{ "\_id" : ObjectId("5f622c58d333e3be77ecc323"), "username" : "GoodGuyGreg", "title" : "Reports a bug in your code", "body" : "Sends you a Pull Request" }

{ "\_id" : ObjectId("5f622c58d333e3be77ecc324"), "username" : "ScumbagSteve", "title" : "Borrows something", "body" : "Sells it" }

{ "\_id" : ObjectId("5f622c58d333e3be77ecc325"), "username" : "ScumbagSteve", "title" : "Borrows everything", "body" : "The end" }

{ "\_id" : ObjectId("5f622c58d333e3be77ecc326"), "username" : "ScumbagSteve", "title" : "Forks your repo on github", "body" : "Sets to private" }

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

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

{ "\_id" : ObjectId("5f622c58d333e3be77ecc321"), "username" : "GoodGuyGreg", "title" : "Passes out at party", "body" : "Wakes up early and cleans house" }

{ "\_id" : ObjectId("5f622c58d333e3be77ecc322"), "username" : "GoodGuyGreg", "title" : "Steals your identity", "body" : "Raises your credit score" }

{ "\_id" : ObjectId("5f622c58d333e3be77ecc323"), "username" : "GoodGuyGreg", "title" : "Reports a bug in your code", "body" : "Sends you a Pull Request" }

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

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

{ "\_id" : ObjectId("5f622c58d333e3be77ecc324"), "username" : "ScumbagSteve", "title" : "Borrows something", "body" : "Sells it" }

{ "\_id" : ObjectId("5f622c58d333e3be77ecc325"), "username" : "ScumbagSteve", "title" : "Borrows everything", "body" : "The end" }

{ "\_id" : ObjectId("5f622c58d333e3be77ecc326"), "username" : "ScumbagSteve", "title" : "Forks your repo on github", "body" : "Sets to private" }

1. find all comments

> db.comments.find()

{ "\_id" : ObjectId("5f623f94d333e3be77ecc327"), "username" : "GoodGuyGreg", "comment" : "Hope you got a good deal!", "post" : "5f622c58d333e3be77ecc324" }

{ "\_id" : ObjectId("5f623f94d333e3be77ecc328"), "username" : "GoodGuyGreg", "comment" : "What's mine is yours!", "post" : "5f622c58d333e3be77ecc325" }

{ "\_id" : ObjectId("5f623f94d333e3be77ecc329"), "username" : "GoodGuyGreg", "comment" : "Don't violate the licensing agreement!", "post" : "5f622c58d333e3be77ecc326" }

{ "\_id" : ObjectId("5f623f94d333e3be77ecc32a"), "username" : "ScumbagSteve", "comment" : "It still isn't clean", "post" : "5f622c58d333e3be77ecc321" }

{ "\_id" : ObjectId("5f6240bdd333e3be77ecc32b"), "username" : "ScumbagSteve", "comment" : "Denied your PR cause I found a hack", "post" : "5f622c58d333e3be77ecc323" }

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

{ "\_id" : ObjectId("5f623f94d333e3be77ecc327"), "username" : "GoodGuyGreg", "comment" : "Hope you got a good deal!", "post" : "5f622c58d333e3be77ecc324" }

{ "\_id" : ObjectId("5f623f94d333e3be77ecc328"), "username" : "GoodGuyGreg", "comment" : "What's mine is yours!", "post" : "5f622c58d333e3be77ecc325" }

{ "\_id" : ObjectId("5f623f94d333e3be77ecc329"), "username" : "GoodGuyGreg", "comment" : "Don't violate the licensing agreement!", "post" : "5f622c58d333e3be77ecc326" }

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

{ "\_id" : ObjectId("5f623f94d333e3be77ecc32a"), "username" : "ScumbagSteve", "comment" : "It still isn't clean", "post" : "5f622c58d333e3be77ecc321" }

{ "\_id" : ObjectId("5f6240bdd333e3be77ecc32b"), "username" : "ScumbagSteve", "comment" : "Denied your PR cause I found a hack", "post" : "5f622c58d333e3be77ecc323" }

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

> db.comments.find({post: "5f622c58d333e3be77ecc323"})

{ "\_id" : ObjectId("5f6240bdd333e3be77ecc32b"), "username" : "ScumbagSteve", "comment" : "Denied your PR cause I found a hack", "post" : "5f622c58d333e3be77ecc323" }