**MongoDB Lab Assignments -Day 1**

# MongoDB Exercise in mongo shell

Connect to a running mongo instance, use a database named **mongo\_practice**. Document all your queries in a javascript file to use as a reference.

# Insert Documents

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

Atlas atlas-1ydbxl-shard-0 [primary] myFirstDatabase> use mongo\_practice

switched to db mongo\_practice

Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.createCollection('movies');

{ ok: 1 }

Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.movies.insert({title : "Fight Club", writer : "Chuck", year: 1999, actors:["Brad Pitt","Edward Norton"]});

DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.

{

acknowledged: true,

insertedIds: { '0': ObjectId("61d96996ad64c7e224c6d113") }

}

Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.movies.insertOne({title : "Fight Club", writer : "Chuck", year: 1999, actors:["Brad Pitt","Edward Norton"]});

{

acknowledged: true,

insertedId: ObjectId("61d969afad64c7e224c6d114")

}

Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.movies.insertOne({title : "Pulp Fiction", writer : "Quentin Tarantino", year: 1994, actors:["Jhon Travolta","Uma Thurman"]});

{

acknowledged: true,

insertedId: ObjectId("61d96b99ad64c7e224c6d115")

}

Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.movies.insertOne({title : "Inglorious Basterds", writer : "Quentin Tarantino", year: 2009, actors:["Brad Pitt","Diane Kruger","Eli Roth"]});

{

acknowledged: true,

insertedId: ObjectId("61d96dd6ad64c7e224c6d116")

}

Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.movies.insertOne({title : "The Hobbit: An Unexpected Journey", writer : "J.R.R Tolkein", year: 2012, franchise: "The Hobbit"});

{

acknowledged: true,

insertedId: ObjectId("61d96eb1ad64c7e224c6d117")

}

Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.movies.insertOne({title : "The Hobbit: The Desolution of Smaug", writer : "J.R.R Tolkein", year: 2013, franchise: "The Hobbit"});

{

acknowledged: true,

insertedId: ObjectId("61d96eebad64c7e224c6d118")

}

Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.movies.insertOne({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 Mountion from falling into the hands on the rising darkness"});

{

acknowledged: true,

insertedId: ObjectId("61d97011ad64c7e224c6d119")

}

Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.movies.insert({title : "Pee Wee Herman's Big Adventure"});

{

acknowledged: true,

insertedIds: { '0': ObjectId("61d970a0ad64c7e224c6d11a") }

}

Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.movies.insert({title : "Avatar"});

{

acknowledged: true,

insertedIds: { '0': ObjectId("61d970b4ad64c7e224c6d11b") }

}

|  |
| --- |
| 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    Reference https://www.tutorialspoint.com/mongodb/mongodb\_insert\_document.htm |

# Query / Find Documents

query the **movies** collection to

1. get all documents

Ans 🡪 Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.movies.find();

[

{

\_id: ObjectId("61d969afad64c7e224c6d114"),

title: 'Fight Club',

writer: 'Chuck',

year: 1999,

actors: [ 'Brad Pitt', 'Edward Norton' ]

},

{

\_id: ObjectId("61d96b99ad64c7e224c6d115"),

title: 'Pulp Fiction',

writer: 'Quentin Tarantino',

year: 1994,

actors: [ 'Jhon Travolta', 'Uma Thurman' ]

},

{

\_id: ObjectId("61d96dd6ad64c7e224c6d116"),

title: 'Inglorious Basterds',

writer: 'Quentin Tarantino',

year: 2009,

actors: [ 'Brad Pitt', 'Diane Kruger', 'Eli Roth' ]

},

{

\_id: ObjectId("61d96eb1ad64c7e224c6d117"),

title: 'The Hobbit: An Unexpected Journey',

writer: 'J.R.R Tolkein',

year: 2012,

franchise: 'The Hobbit'

},

{

\_id: ObjectId("61d96eebad64c7e224c6d118"),

title: 'The Hobbit: The Desolution of Smaug',

writer: 'J.R.R Tolkein',

year: 2013,

franchise: 'The Hobbit'

},

{

\_id: ObjectId("61d97011ad64c7e224c6d119"),

title: 'The Hobbit: The Battle of the Five Armies',

writer: 'J.R.R Tolkein',

year: 2012,

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

Ans 🡪 Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.movies.find({writer:"Quentin Tarantino"});

[

{,

\_id: ObjectId("61d96b99ad64c7e224c6d115"),

title: 'Pulp Fiction',0ad64c7e224c6d11a"),

writer: 'Quentin Tarantino', Adventure"

year: 1994,

actors: [ 'Jhon Travolta', 'Uma Thurman' ] title: 'Avatar' }

},

{as atlas-1ydbxl-shard-0 [primary] mongo\_practice>

\_id: ObjectId("61d96dd6ad64c7e224c6d116"),

title: 'Inglorious Basterds',

writer: 'Quentin Tarantino',

year: 2009,

actors: [ 'Brad Pitt', 'Diane Kruger', 'Eli Roth' ]

}

]

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

Ans 🡪 Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.movies.find({actors:"Brad Pitt"});

[

{

\_id: ObjectId("61d969afad64c7e224c6d114"),

title: 'Fight Club',

writer: 'Chuck',

year: 1999,

actors: [ 'Brad Pitt', 'Edward Norton' ]

},

{

\_id: ObjectId("61d96dd6ad64c7e224c6d116"),

title: 'Inglorious Basterds',

writer: 'Quentin Tarantino',

year: 2009,

actors: [ 'Brad Pitt', 'Diane Kruger', 'Eli Roth' ]

}

]

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

Ans🡪 Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.movies.find({franchise:"The Hobbit"});

[

{

\_id: ObjectId("61d96eb1ad64c7e224c6d117"),

title: 'The Hobbit: An Unexpected Journey',

writer: 'J.R.R Tolkein',

year: 2012,

franchise: 'The Hobbit'

},

{

\_id: ObjectId("61d96eebad64c7e224c6d118"),

title: 'The Hobbit: The Desolution of Smaug',

writer: 'J.R.R Tolkein',

year: 2013,

franchise: 'The Hobbit'

},

{

\_id: ObjectId("61d97011ad64c7e224c6d119"),

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 Mountion from falling into the hands on the rising darkness'

}

]

1. get all movies released in the 90s

Ans-> Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.movies.find({year:({$gt: 1990,$lt: 2000})});

[

{

\_id: ObjectId("61d969afad64c7e224c6d114"),

title: 'Fight Club',

writer: 'Chuck',

year: 1999,

actors: [ 'Brad Pitt', 'Edward Norton' ]

},

{

\_id: ObjectId("61d96b99ad64c7e224c6d115"),

title: 'Pulp Fiction',

writer: 'Quentin Tarantino',

year: 1994,

actors: [ 'Jhon Travolta', 'Uma Thurman' ]

}

]

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

Ans🡪 Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.movies.find({$or:[{year:{$gt: 2010}},{year:{$lt: 2000}}]});

[

{

\_id: ObjectId("61d969afad64c7e224c6d114"),

title: 'Fight Club',

writer: 'Chuck',

year: 1999,

actors: [ 'Brad Pitt', 'Edward Norton' ]

},

{

\_id: ObjectId("61d96b99ad64c7e224c6d115"),

title: 'Pulp Fiction',

writer: 'Quentin Tarantino',

year: 1994,

actors: [ 'Jhon Travolta', 'Uma Thurman' ]

},

{

\_id: ObjectId("61d96eb1ad64c7e224c6d117"),

title: 'The Hobbit: An Unexpected Journey',

writer: 'J.R.R Tolkein',

year: 2012,

franchise: 'The Hobbit'

},

{

\_id: ObjectId("61d96eebad64c7e224c6d118"),

title: 'The Hobbit: The Desolution of Smaug',

writer: 'J.R.R Tolkein',

year: 2013,

franchise: 'The Hobbit'

},

{

\_id: ObjectId("61d97011ad64c7e224c6d119"),

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 Mountion from falling into the hands on the rising darkness'

}

]

Reference:

https://www.tutorialspoint.com/mongodb/mongodb\_query\_document.htm

# 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."

Ans🡪 Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.movies.update({\_id:ObjectId("61d96eb1ad64c7e224c6d117")},{$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."}});

DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany, or bulkWrite.

{

acknowledged: true,

insertedId: null,

matchedCount: 1,

modifiedCount: 1,

upsertedCount: 0

}

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."

Ans🡪 Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.movies.update({\_id:ObjectId("61d96eebad64c7e224c6d118")},{$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,

insertedId: null,

matchedCount: 1,

modifiedCount: 1,

upsertedCount: 0

}

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

Ans 🡪 Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.movies.update({\_id:ObjectId("61d96b99ad64c7e224c6d115")},{$push:{actors:"Samuel L. Jackson"}});

{

acknowledged: true,

insertedId: null,

matchedCount: 1,

modifiedCount: 1,

upsertedCount: 0

}

Reference:

https://www.tutorialspoint.com/mongodb/mongodb\_update\_document.htm

# Text Search

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

Ans🡪 Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.movies.find({synopsis:{$regex:"Bilbo"}});

[

{

\_id: ObjectId("61d96eb1ad64c7e224c6d117"),

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("61d96eebad64c7e224c6d118"),

title: 'The Hobbit: The Desolution 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("61d97011ad64c7e224c6d119"),

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 Mountion from falling into the hands on the rising darkness'

}

]

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

Ans🡪 Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.movies.find({synopsis:{$regex:"Gandalf"}});

[

{

\_id: ObjectId("61d96eebad64c7e224c6d118"),

title: 'The Hobbit: The Desolution 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"

Ans🡪 Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.movies.find({$and:[{synopsis:{$regex:"Bilbo"}}, {synopsis:{$not:/Gandalf/}}]});

[

{

\_id: ObjectId("61d96eb1ad64c7e224c6d117"),

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("61d97011ad64c7e224c6d119"),

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 Mountion from falling into the hands on the rising darkness'

}

]

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

Ans🡪 Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.movies.find({$or:[{synopsis:{$regex:"dwarves"}}, {synopsis:{$regex:"hobbit"}}]});

[

{

\_id: ObjectId("61d96eb1ad64c7e224c6d117"),

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("61d96eebad64c7e224c6d118"),

title: 'The Hobbit: The Desolution 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"

Ans🡪 Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.movies.find({$and:[{synopsis:{$regex:"gold"}}, {synopsis:{$regex:"dragon"}}]});

[

{

\_id: ObjectId("61d96eb1ad64c7e224c6d117"),

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.'

}

]

Reference: https://www.tutorialspoint.com/mongodb/mongodb\_text\_search.htm

# Delete Documents

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

Ans🡪 Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.movies.remove({\_id:ObjectId("61d970a0ad64c7e224c6d11a")});

{ acknowledged: true, deletedCount: 1 }

1. delete the movie "Avatar"

Ans🡪 Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.movies.remove({\_id:ObjectId("61d970b4ad64c7e224c6d11b")});

{ acknowledged: true, deletedCount: 1 }

Reference: https://www.tutorialspoint.com/mongodb/mongodb\_delete\_document.htm

# Relationships

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

Ans🡪 Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.createCollection('users');

{ ok: 1 }

|  |
| --- |
| username : GoodGuyGreg first\_name : "Good Guy" last\_name : "Greg" username : ScumbagSteve full\_name :  first : "Scumbag" last : "Steve" |

Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.users.insertOne({username: "GoodGuyGreg", first\_name : "Good Guy" ,last\_name : "Greg"});

{

acknowledged: true,

insertedId: ObjectId("61d9864aad64c7e224c6d11c")

}

Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.users.insert({username:"ScumbagSteve", fullname:{first: "Scumbag", last:"Steve"}});

{

acknowledged: true,

insertedIds: { '0': ObjectId("61d986acad64c7e224c6d11d") }

}

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

Ans🡪Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.createCollection('posts');

{ ok: 1 }

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 |

Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.posts.insert({username:"GoodGuyGreg", title:"Passes out at Party", body:"Wakes up early and cleans house"});

{

acknowledged: true,

insertedIds: { '0': ObjectId("61d987b8ad64c7e224c6d11e") }

}

Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.posts.insert({ username:"GoodGuyGreg", title:"Steals your identity", body:"Raises your credit score"});

{

acknowledged: true,

insertedIds: { '0': ObjectId("61d9880bad64c7e224c6d11f") }

}

Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.posts.insert({username:"GoodGuyGreg", title:"Reports a bug in your code", body:"Sends you a pull request"});

{

acknowledged: true,

insertedIds: { '0': ObjectId("61d98830ad64c7e224c6d120") }

}

Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.posts.insert({ username:"ScumbagSteve", title:"Borrows something", body:"Sells it"});

{

acknowledged: true,

insertedIds: { '0': ObjectId("61d9886bad64c7e224c6d121") }

}

Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.posts.insert({ username:"ScumbagSteve", title:"Borrows everything", body:"The end"});

{

acknowledged: true,

insertedIds: { '0': ObjectId("61d9888dad64c7e224c6d122") }

}

Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.posts.insert({username:"ScumbagSteve", title:"Forks your repo on github", body:"Sets to private"});

{

acknowledged: true,

insertedIds: { '0': ObjectId("61d988bbad64c7e224c6d123") }

}

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

Ans🡪 Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.createCollection('comments');

{ ok: 1 }

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"

Ans 🡪Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.comments.insert({ username:"GoodGuyGreg", comment:"Hope you got a good deal!", post:ObjectId("61d9886bad64c7e224c6d121")});

{

acknowledged: true,

insertedIds: { '0': ObjectId("61d989b4ad64c7e224c6d124") }

}

Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.comments.insert({username:"GoodGuyGreg", comment:"What's mine is yours!", post:ObjectId("61d9888dad64c7e224c6d122")});

{

acknowledged: true,

insertedIds: { '0': ObjectId("61d98a15ad64c7e224c6d125") }

}

Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.comments.insert({username:"GoodGuyGreg", comment:"Don't violate the licensing agreement!", post:ObjectId("61d988bbad64c7e224c6d123")});

{

acknowledged: true,

insertedIds: { '0': ObjectId("61d98a63ad64c7e224c6d126") }

}

Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.comments.insert({username:"ScumbagSteve", comment:"It still isn't clean", post:ObjectId("61d987b8ad64c7e224c6d11e")});

{

acknowledged: true,

insertedIds: { '0': ObjectId("61d98ad3ad64c7e224c6d127") }

}

Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.comments.insert({username:"ScumbagSteve", comment:"Denied your PR cause I found a hack", post:ObjectId("61d98830ad64c7e224c6d120")});

{

acknowledged: true,

insertedIds: { '0': ObjectId("61d98b3dad64c7e224c6d128") }

}

# Querying related collections

1. find all users

Ans🡪 Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.users.find().pretty();

[

{

\_id: ObjectId("61d9864aad64c7e224c6d11c"),

username: 'GoodGuyGreg',

first\_name: 'Good Guy',

last\_name: 'Greg'

},

{

\_id: ObjectId("61d986acad64c7e224c6d11d"),

username: 'ScumbagSteve',

fullname: { first: 'Scumbag', last: 'Steve' }

}

]

1. find all posts

Ans🡪 Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.posts.find().pretty();

[

{

\_id: ObjectId("61d987b8ad64c7e224c6d11e"),

username: 'GoodGuyGreg',

title: 'Passes out at Party',

body: 'Wakes up early and cleans house'

},

{

\_id: ObjectId("61d9880bad64c7e224c6d11f"),

username: 'GoodGuyGreg',

title: 'Steals your identity',

body: 'Raises your credit score'

},

{

\_id: ObjectId("61d98830ad64c7e224c6d120"),

username: 'GoodGuyGreg',

title: 'Reports a bug in your code',

body: 'Sends you a pull request'

},

{

\_id: ObjectId("61d9886bad64c7e224c6d121"),

username: 'ScumbagSteve',

title: 'Borrows something',

body: 'Sells it'

},

{

\_id: ObjectId("61d9888dad64c7e224c6d122"),

username: 'ScumbagSteve',

title: 'Borrows everything',

body: 'The end'

},

{

\_id: ObjectId("61d988bbad64c7e224c6d123"),

username: 'ScumbagSteve',

title: 'Forks your repo on github',

body: 'Sets to private'

}

]

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

Ans🡪 Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.posts.find({username:"GoodGuyGreg"});

[

{

\_id: ObjectId("61d987b8ad64c7e224c6d11e"),

username: 'GoodGuyGreg',

title: 'Passes out at Party',

body: 'Wakes up early and cleans house'

},

{

\_id: ObjectId("61d9880bad64c7e224c6d11f"),

username: 'GoodGuyGreg',

title: 'Steals your identity',

body: 'Raises your credit score'

},

{

\_id: ObjectId("61d98830ad64c7e224c6d120"),

username: 'GoodGuyGreg',

title: 'Reports a bug in your code',

body: 'Sends you a pull request'

}

]

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

Ans🡪 Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.posts.find({username:"ScumbagSteve"});

[

{

\_id: ObjectId("61d9886bad64c7e224c6d121"),

username: 'ScumbagSteve',

title: 'Borrows something',

body: 'Sells it'

},

{

\_id: ObjectId("61d9888dad64c7e224c6d122"),

username: 'ScumbagSteve',

title: 'Borrows everything',

body: 'The end'

},

{

\_id: ObjectId("61d988bbad64c7e224c6d123"),

username: 'ScumbagSteve',

title: 'Forks your repo on github',

body: 'Sets to private'

}

]

1. find all comments

Ans🡪 Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.comments.find().pretty();

[

{

\_id: ObjectId("61d989b4ad64c7e224c6d124"),

username: 'GoodGuyGreg',

comment: 'Hope you got a good deal!',

post: ObjectId("61d9886bad64c7e224c6d121")

},

{

\_id: ObjectId("61d98a15ad64c7e224c6d125"),

username: 'GoodGuyGreg',

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

post: ObjectId("61d9888dad64c7e224c6d122")

},

{

\_id: ObjectId("61d98a63ad64c7e224c6d126"),

username: 'GoodGuyGreg',

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

post: ObjectId("61d988bbad64c7e224c6d123")

},

{

\_id: ObjectId("61d98ad3ad64c7e224c6d127"),

username: 'ScumbagSteve',

comment: "It still isn't clean",

post: ObjectId("61d987b8ad64c7e224c6d11e")

},

{

\_id: ObjectId("61d98b3dad64c7e224c6d128"),

username: 'ScumbagSteve',

comment: 'Denied your PR cause I found a hack',

post: ObjectId("61d98830ad64c7e224c6d120")

}

]

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

Ans🡪 Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.comments.find({username:"GoodGuyGreg"});

[

{

\_id: ObjectId("61d989b4ad64c7e224c6d124"),

username: 'GoodGuyGreg',

comment: 'Hope you got a good deal!',

post: ObjectId("61d9886bad64c7e224c6d121")

},

{

\_id: ObjectId("61d98a15ad64c7e224c6d125"),

username: 'GoodGuyGreg',

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

post: ObjectId("61d9888dad64c7e224c6d122")

},

{

\_id: ObjectId("61d98a63ad64c7e224c6d126"),

username: 'GoodGuyGreg',

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

post: ObjectId("61d988bbad64c7e224c6d123")

}

]

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

Ans🡪 Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.comments.find({username:"ScumbagSteve"});

[

{

\_id: ObjectId("61d98ad3ad64c7e224c6d127"),

username: 'ScumbagSteve',

comment: "It still isn't clean",

post: ObjectId("61d987b8ad64c7e224c6d11e")

},

{

\_id: ObjectId("61d98b3dad64c7e224c6d128"),

username: 'ScumbagSteve',

comment: 'Denied your PR cause I found a hack',

post: ObjectId("61d98830ad64c7e224c6d120")

}

]

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

Ans🡪 Atlas atlas-1ydbxl-shard-0 [primary] mongo\_practice> db.posts.aggregate([

... {

..... $match: { title: 'Reports a bug in your code' }

..... },

... {

..... $lookup: {

....... from: 'comments',

....... localField: '\_id',

....... foreignField: 'post',

....... as: 'comments'

....... }

..... }

... ]).pretty();

[

{

\_id: ObjectId("61d98830ad64c7e224c6d120"),

username: 'GoodGuyGreg',

title: 'Reports a bug in your code',

body: 'Sends you a pull request',

comments: [

{

\_id: ObjectId("61d98b3dad64c7e224c6d128"),

username: 'ScumbagSteve',

comment: 'Denied your PR cause I found a hack',

post: ObjectId("61d98830ad64c7e224c6d120")

}

]

}

]

References:

<https://docs.mongodb.com/manual/reference/method/db.collection.find/>

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@