

Assignment 1

```
Command Prompt - mongod
Microsoft Windows [Version 10.0.19043.1165]
(c) Microsoft Corporation. All rights reserved.

C:\Users\VTAMSHET>mongod
{"t":{"sdate":"2022-06-22T11:05:35.715+05:30"},"s":"I", "c":"CONTROL", "id":23285, "ctx":"main","msg":"Automatically disabling TLS 1.0, to force-enable TLS 1.0 specify --sslDisabledProtocols 'none'"}
{"t":{"sdate":"2022-06-22T11:05:35.719+05:30"},"s":"I", "c":"NETWORK", "id":4915701, "ctx":"main","msg":"Initialized wire specification","attr":{"spec":{"incomingExternalClient":{"minWireVersion":0,"maxWireVersion":13},"incomingInternalClient":{"minWireVersion":0,"maxWireVersion":13},"outgoing":{"minWireVersion":0,"maxWireVersion":13},"isInternalClient":true}}}
{"t":{"sdate":"2022-06-22T11:05:35.719+05:30"},"s":"W", "c":"ASIO", "id":22601, "ctx":"main","msg":"No TransportLayer configured during NetworkInterface startup"}
{"t":{"sdate":"2022-06-22T11:05:35.720+05:30"},"s":"I", "c":"NETWORK", "id":4648602, "ctx":"main","msg":"Implicit TCP FastOpen in use."}
{"t":{"sdate":"2022-06-22T11:05:35.722+05:30"},"s":"W", "c":"ASIO", "id":22601, "ctx":"main","msg":"No TransportLayer configured during NetworkInterface startup"}
{"t":{"sdate":"2022-06-22T11:05:35.723+05:30"},"s":"I", "c":"REPL", "id":5123008, "ctx":"main","msg":"Successfully registered PrimaryOnlyService","attr":{"service":"TenantMigrationDonorService","ns":"config.tenantMigrationDonors"}}
{"t":{"sdate":"2022-06-22T11:05:35.723+05:30"},"s":"I", "c":"REPL", "id":5123008, "ctx":"main","msg":"Successfully registered PrimaryOnlyService","attr":{"service":"TenantMigrationRecipientService","ns":"config.tenantMigrationRecipients"}}
{"t":{"sdate":"2022-06-22T11:05:35.723+05:30"},"s":"I", "c":"CONTROL", "id":5945603, "ctx":"main","msg":"Multi threading initialized"}
{"t":{"sdate":"2022-06-22T11:05:35.726+05:30"},"s":"I", "c":"CONTROL", "id":4615611, "ctx":"initandlisten","msg":"MongoDB starting","attr":{"pid":14812,"port":27017,"dbPath":"C:/data/db/","architecture":"64-bit","host":"INLEN8520050891"}}
{"t":{"sdate":"2022-06-22T11:05:35.727+05:30"},"s":"I", "c":"CONTROL", "id":23398, "ctx":"initandlisten","msg":"Target operating system minimum version","attr":{"targetMinOS":"Windows 7/Windows Server 2008 R2"}}
{"t":{"sdate":"2022-06-22T11:05:35.727+05:30"},"s":"I", "c":"CONTROL", "id":23403, "ctx":"initandlisten","msg":"Build Info","attr":{"buildInfo":{"version":"5.0.9","gitVersion":"6f7dae919422dcdf4892c10ff28cdc721ad00e6","modules":[],"allocator":"tcmalloc","environment":{"distmod":"windows","distarch":"x86_64","target_arch":"x86_64"}}}}}
{"t":{"sdate":"2022-06-22T11:05:35.727+05:30"},"s":"I", "c":"CONTROL", "id":51765, "ctx":"initandlisten","msg":"Operating System","attr":{"os":{"name":"Microsoft Windows 10","version":"10.0 (build 19043)"}}}
{"t":{"sdate":"2022-06-22T11:05:35.728+05:30"},"s":"I", "c":"CONTROL", "id":21951, "ctx":"initandlisten","msg":"Options set by command line","attr":{"options":{}}}
{"t":{"sdate":"2022-06-22T11:05:35.733+05:30"},"s":"E", "c":"CONTROL", "id":20557, "ctx":"initandlisten","msg":"DBException in initAndListen, terminating","attr":{"error":{"NonExistentPath: Data directory C:\\data\\db\\ not found. Create the 'storage.dbPath' option in the configuration file."}}}
{"t":{"sdate":"2022-06-22T11:05:35.733+05:30"},"s":"I", "c":"REPL", "id":4784900, "ctx":"initandlisten","msg":"Stepping down the ReplicationCoordinator for shutdown","attr":{"waitTimeMillis":15000}}
{"t":{"sdate":"2022-06-22T11:05:35.733+05:30"},"s":"I", "c":"COMMAND", "id":4784901, "ctx":"initandlisten","msg":"Shutting down the MirrorMaestro"}
{"t":{"sdate":"2022-06-22T11:05:35.734+05:30"},"s":"I", "c":"SHARDING", "id":4784902, "ctx":"initandlisten","msg":"Shutting down the WaitForMajorityService"}
{"t":{"sdate":"2022-06-22T11:05:35.736+05:30"},"s":"I", "c":"NETWORK", "id":20562, "ctx":"initandlisten","msg":"Shutdown: going to close listening sockets"}
{"t":{"sdate":"2022-06-22T11:05:35.736+05:30"},"s":"I", "c":"NETWORK", "id":4784905, "ctx":"initandlisten","msg":"Shutting down the global connection pool"}
{"t":{"sdate":"2022-06-22T11:05:35.737+05:30"},"s":"I", "c":"CONTROL", "id":4784906, "ctx":"initandlisten","msg":"Shutting down the FlowControlTICKETholder"}
{"t":{"sdate":"2022-06-22T11:05:35.738+05:30"},"s":"I", "c":"-", "id":20520, "ctx":"initandlisten","msg":"Stopping further Flow Control ticket acquisitions."}
{"t":{"sdate":"2022-06-22T11:05:35.739+05:30"},"s":"I", "c":"NETWORK", "id":4784918, "ctx":"initandlisten","msg":"Shutting down the ReplicaSetMonitor"}
{"t":{"sdate":"2022-06-22T11:05:35.746+05:30"},"s":"I", "c":"SHARDING", "id":4784921, "ctx":"initandlisten","msg":"Shutting down the MigrationUtilExecutor"}
{"t":{"sdate":"2022-06-22T11:05:35.748+05:30"},"s":"I", "c":"ASIO", "id":22582, "ctx":"MigrationUtil-TaskExecutor","msg":"Killing all outstanding egress activity."}
{"t":{"sdate":"2022-06-22T11:05:35.749+05:30"},"s":"I", "c":"COMMAND", "id":4784923, "ctx":"initandlisten","msg":"Shutting down the ServiceEntryPoint"}

Command Prompt - mongo
https://community.mongodb.com

The server generated these startup warnings when booting:
2022-06-22T11:06:44.090+05:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
2022-06-22T11:06:44.091+05:30: This server is bound to localhost. Remote systems will be unable to connect to this server. Start the server with --bind_ip <address> to specify which IP addresses it should serve responses from, or with --bind_ip_all to bind to all interfaces. If this behavior is desired, start the server with --bind_ip 127.0.0.1 to disable this warning

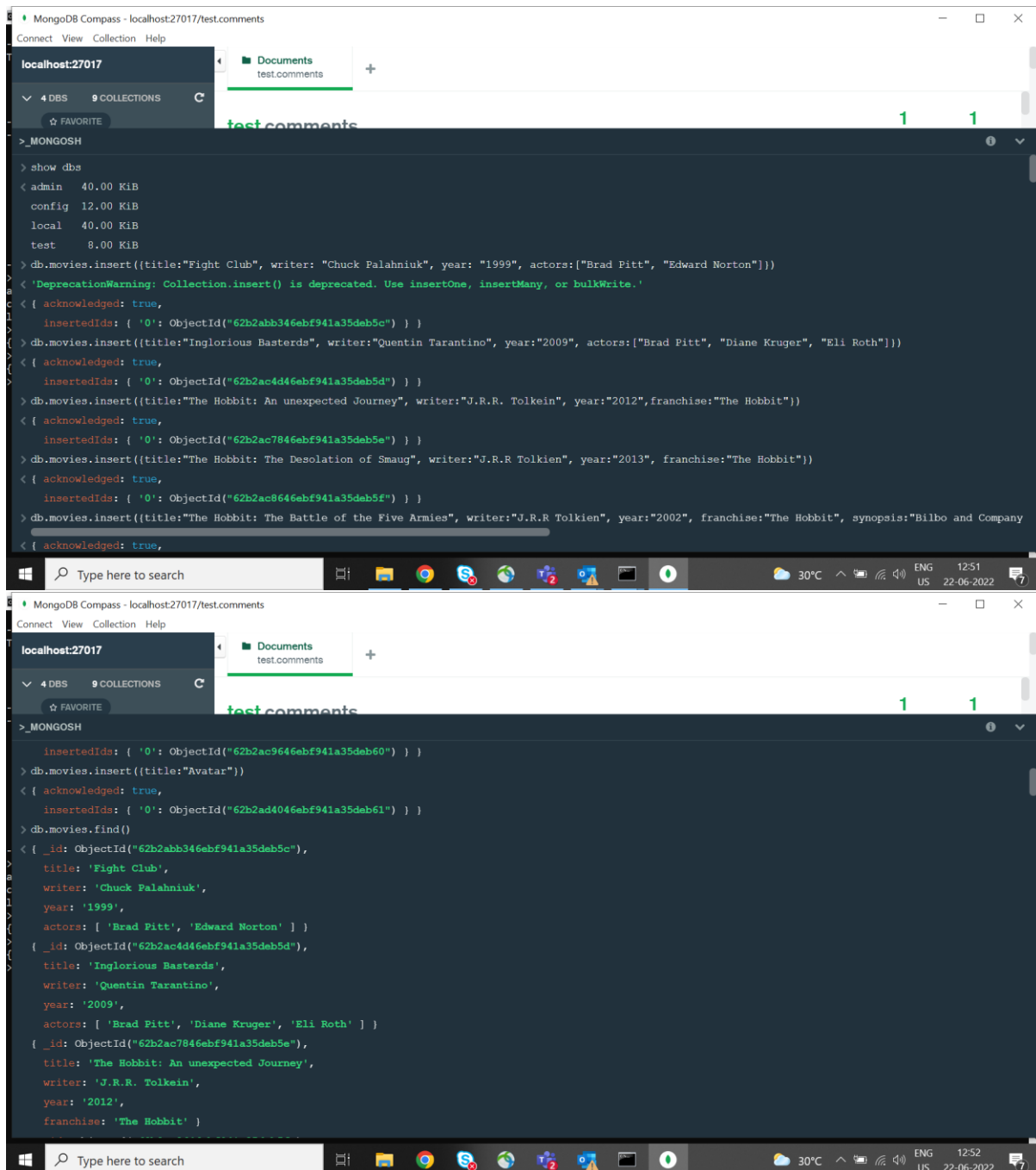
Enable MongoDB's free cloud-based monitoring service, which will then receive and display metrics about your deployment (disk utilization, CPU, operation statistics, etc).

The monitoring data will be available on a MongoDB website with a unique URL accessible to you and anyone you share the URL with. MongoDB may use this information to make product improvements and to suggest MongoDB products and deployment options to you.

To enable free monitoring, run the following command: db.enableFreeMonitoring()
To permanently disable this reminder, run the following command: db.disableFreeMonitoring()

> show dbs
admin 0.000GB
config 0.000GB
local 0.000GB
> db.createCollection("Movies")
{ "ok" : 1 }
> db.createCollection("Users")
{ "ok" : 1 }
>
```

Assignment 1



The image displays two screenshots of the MongoDB Compass interface, showing database operations and results.

Top Screenshot: The interface shows the 'test.comments' collection. The command window displays the following commands and results:

```
> show dbs
{ admin: 40.00 KiB, config: 12.00 KiB, local: 40.00 KiB, test: 8.00 KiB }

> db.movies.insert({title:"Fight Club", writer: "Chuck Palahniuk", year: "1999", actors:["Brad Pitt", "Edward Norton"]})
{ acknowledged: true, insertedIds: { '0': ObjectId("62b2abb346ebf941a35deb5c") } }

> db.movies.insert({title:"Inglorious Basterds", writer:"Quentin Tarantino", year:"2009", actors:["Brad Pitt", "Diane Kruger", "Eli Roth"]})
{ acknowledged: true, insertedIds: { '0': ObjectId("62b2ac4d46ebf941a35deb5d") } }

> db.movies.insert({title:"The Hobbit: An unexpected Journey", writer:"J.R.R. Tolkien", year:"2012", franchise:"The Hobbit"})
{ acknowledged: true, insertedIds: { '0': ObjectId("62b2ac8646ebf941a35deb5e") } }

> db.movies.insert({title:"The Hobbit: The Desolation of Smaug", writer:"J.R.R Tolkien", year:"2013", franchise:"The Hobbit"})
{ acknowledged: true, insertedIds: { '0': ObjectId("62b2ac8646ebf941a35deb5f") } }

> db.movies.insert({title:"The Hobbit: The Battle of the Five Armies", writer:"J.R.R Tolkien", year:"2002", franchise:"The Hobbit", synopsis:"Bilbo and Company"})
{ acknowledged: true, insertedIds: { '0': ObjectId("62b2ad4046ebf941a35deb60") } }
```

Bottom Screenshot: The interface shows the 'test.comments' collection. The command window displays the following commands and results:

```
> db.movies.insert({title:"Avatar"})
{ acknowledged: true, insertedIds: { '0': ObjectId("62b2ad4046ebf941a35deb61") } }

> db.movies.find()
{ '_id': ObjectId("62b2abb346ebf941a35deb5c"), title: 'Fight Club', writer: 'Chuck Palahniuk', year: '1999', actors: [ 'Brad Pitt', 'Edward Norton' ] }, { '_id': ObjectId("62b2ac4d46ebf941a35deb5d"), title: 'Inglorious Basterds', writer: 'Quentin Tarantino', year: '2009', actors: [ 'Brad Pitt', 'Diane Kruger', 'Eli Roth' ] }, { '_id': ObjectId("62b2ac8646ebf941a35deb5e"), title: 'The Hobbit: An unexpected Journey', writer: 'J.R.R. Tolkien', year: '2012', franchise: 'The Hobbit' }
```

Assignment 1

The image displays two screenshots of the MongoDB Compass interface, showing the 'test.comments' collection in the 'localhost:27017' database. The interface includes a sidebar with '4 DBS' and '9 COLLECTIONS', and a main area for the 'test.comments' collection. The data is presented in a JSON format within the MONGODB shell.

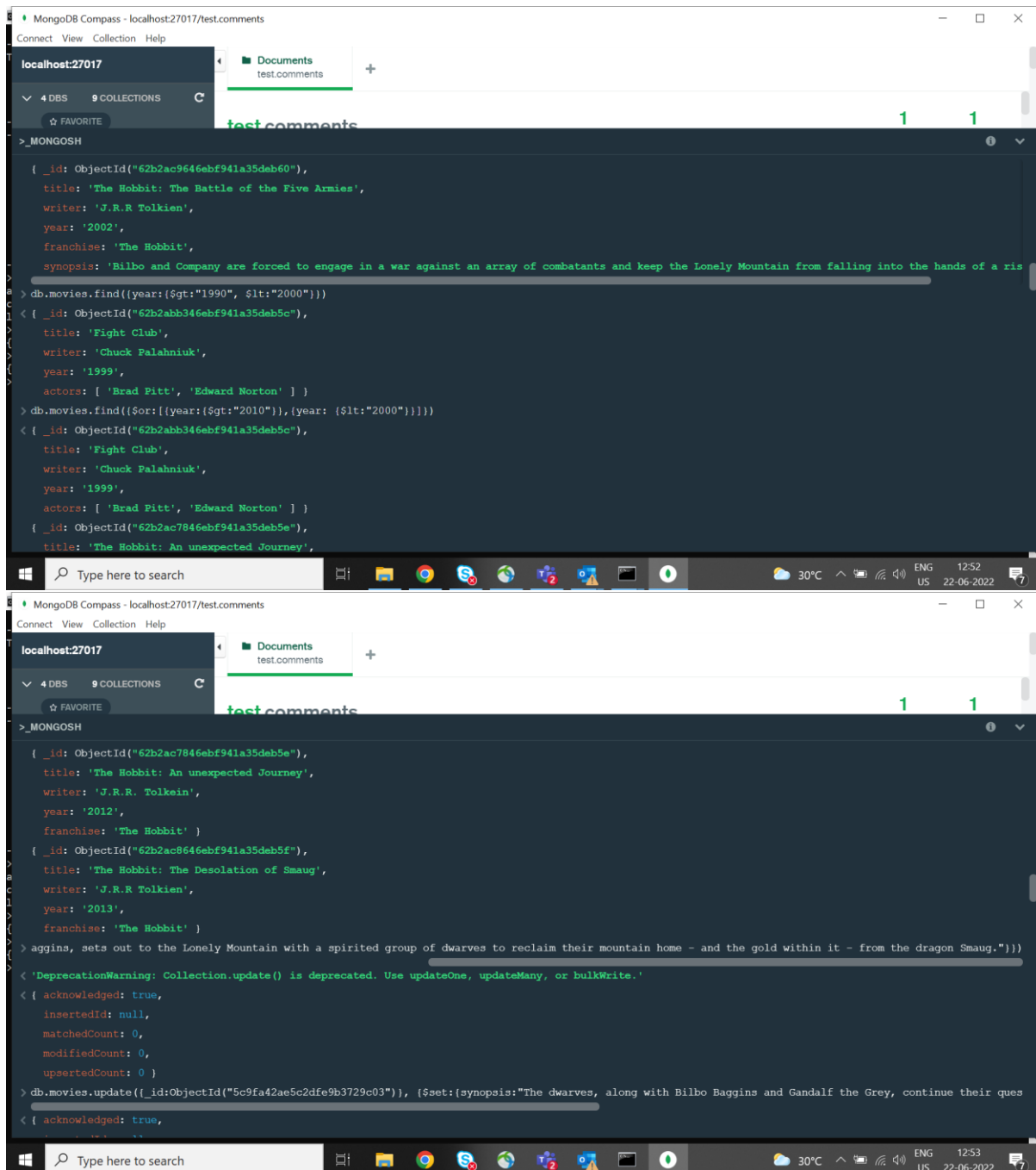
Screenshot 1 (Top): Shows the first three documents in the collection. The first document is for 'The Hobbit: The Desolation of Smaug' (2013), the second is 'The Hobbit: The Battle of the Five Armies' (2012), and the third is 'Avatar'.

```
{ "_id": ObjectId("62b2ac8646ebf941a35deb5f"),  
  title: 'The Hobbit: The Desolation of Smaug',  
  writer: 'J.R.R Tolkien',  
  year: '2013',  
  franchise: 'The Hobbit' }  
{ "_id": ObjectId("62b2ac9646ebf941a35deb60"),  
  title: 'The Hobbit: The Battle of the Five Armies',  
  writer: 'J.R.R Tolkien',  
  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 ris  
{ "_id": ObjectId("62b2ad4046ebf941a35deb61"), title: 'Avatar' }
```

Screenshot 2 (Bottom): Shows the next three documents in the collection. The first is 'Inglorious Basterds' (2009), the second is 'The Hobbit: An unexpected Journey' (2012), and the third is 'The Hobbit: The Desolation of Smaug' (2013).

```
> db.movies.find({writer:"Quentin Tarantino"})  
< { "_id": ObjectId("62b2ac4d46ebf941a35deb5d"),  
  title: 'Inglorious Basterds',  
  writer: 'Quentin Tarantino',  
  year: '2009',  
  actors: [ 'Brad Pitt', 'Diane Kruger', 'Eli Roth' ] }  
> db.movies.find({actors:"Brad Pitt"})  
< { "_id": ObjectId("62b2abb346ebf941a35deb5c"),  
  title: 'Fight Club',  
  writer: 'Chuck Palahniuk',  
  year: '1999',  
  actors: [ 'Brad Pitt', 'Edward Norton' ] }  
{ "_id": ObjectId("62b2ac4d46ebf941a35deb5d"),  
  title: 'Inglorious Basterds',  
  writer: 'Quentin Tarantino',  
  year: '2009',  
  actors: [ 'Brad Pitt', 'Diane Kruger', 'Eli Roth' ] }  
> db.movies.find({franchise:"The Hobbit"})  
< { "_id": ObjectId("62b2ac7846ebf941a35deb5e"),  
  title: 'The Hobbit: An unexpected Journey',  
  writer: 'J.R.R. Tolkein',  
  year: '2012',  
  franchise: 'The Hobbit' }  
{ "_id": ObjectId("62b2ac8646ebf941a35deb5f"),  
  title: 'The Hobbit: The Desolation of Smaug',  
  writer: 'J.R.R Tolkien',  
  year: '2013',  
  franchise: 'The Hobbit' }
```

Assignment 1



The image shows two screenshots of the MongoDB Compass interface, specifically the MONGODB shell, demonstrating database operations on a local MongoDB instance at localhost:27017.

Top Screenshot:

- The interface shows the 'test.comments' collection selected.
- The MONGODB shell contains the following commands and results:
 - `> { _id: ObjectId("62b2ac9646ebf941a35deb60"), title: 'The Hobbit: The Battle of the Five Armies', writer: 'J.R.R. Tolkien', year: '2002', 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 ris`
 - `> db.movies.find({year:{$gt:"1990", $lt:"2000"}})` returns one document:
 - `< { _id: ObjectId("62b2abb346ebf941a35deb5c"), title: 'Fight Club', writer: 'Chuck Palahniuk', year: '1999', actors: ['Brad Pitt', 'Edward Norton'] }`
 - `> db.movies.find({$or:[{year:{$gt:"2010"}},{year:{$lt:"2000"}}]})` returns one document:
 - `< { _id: ObjectId("62b2abb346ebf941a35deb5c"), title: 'Fight Club', writer: 'Chuck Palahniuk', year: '1999', actors: ['Brad Pitt', 'Edward Norton'] }`
 - `> { _id: ObjectId("62b2ac7846ebf941a35deb5e"), title: 'The Hobbit: An unexpected Journey',`

Bottom Screenshot:

- The MONGODB shell continues with the following commands and results:
 - `> { _id: ObjectId("62b2ac7846ebf941a35deb5e"), title: 'The Hobbit: An unexpected Journey', writer: 'J.R.R. Tolkein', year: '2012', franchise: 'The Hobbit' }`
 - `> { _id: ObjectId("62b2ac8646ebf941a35deb5f"), title: 'The Hobbit: The Desolation of Smaug', writer: 'J.R.R. Tolkien', year: '2013', franchise: 'The Hobbit' }`
 - `> aggrins, 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: 0, modifiedCount: 0, upsertedCount: 0 }`
 - `> db.movies.update({_id:ObjectId("5c9fa42ae5c2dfe9b3729c03")}, {$set:{synopsis:"The dwarves, along with Bilbo Baggins and Gandalf the Grey, continue their ques`
 - `< { acknowledged: true,`

Assignment 1

The image shows two screenshots of the MongoDB Compass interface, specifically the MONGODB shell, demonstrating database operations on a local MongoDB instance at localhost:27017.

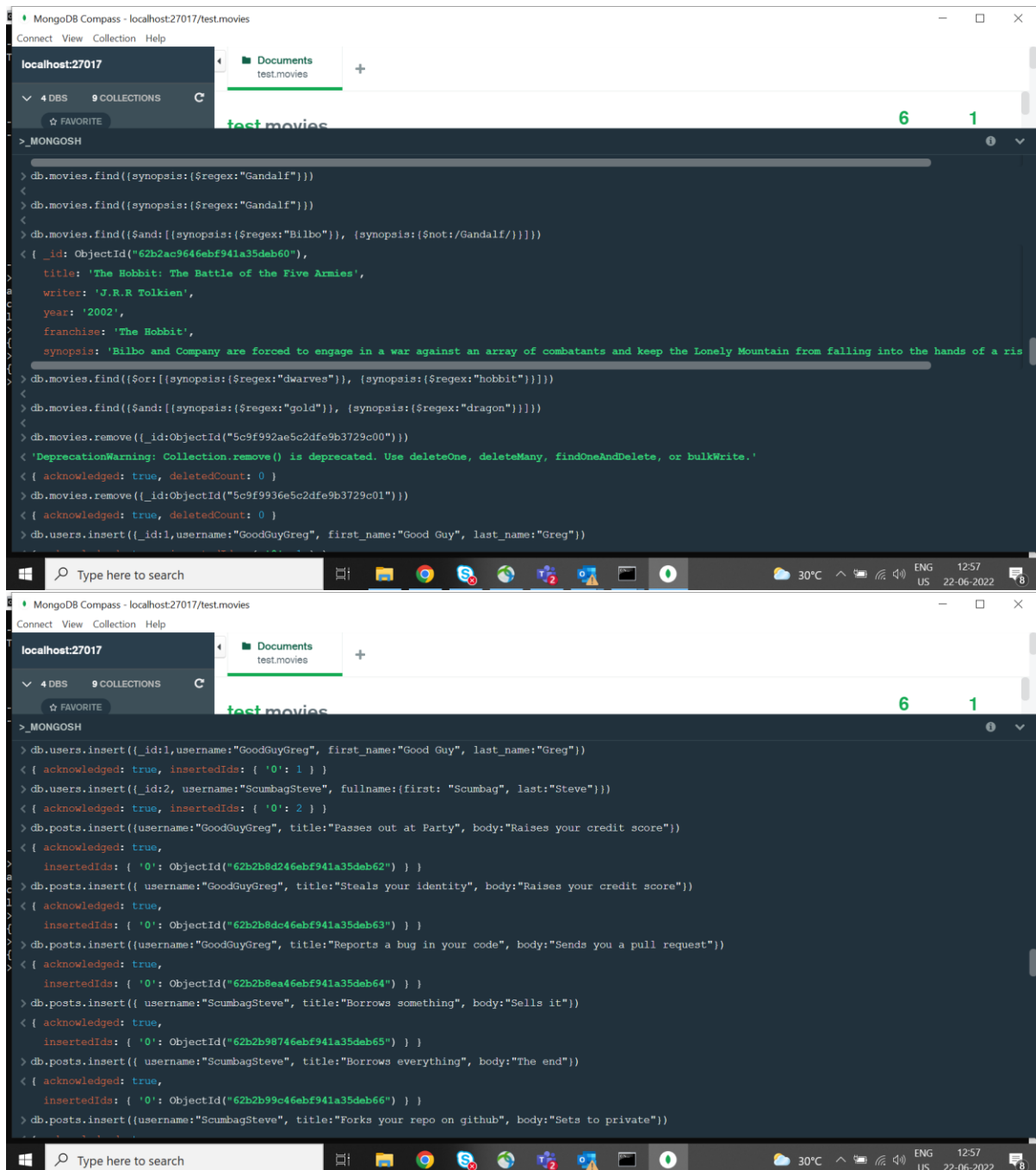
Top Screenshot:

- The interface shows the 'test.movies' collection with 6 documents and 1 index.
- The MONGODB shell contains the following commands and results:
 - `db.movies.find({$and:[{synopsis:{$regex:"Bilbo"}}, {synopsis:{$not:/Gandalf/}}])` returns one document: `{_id: ObjectId("62b2ac9646ebf941a35deb60"), title: 'The Hobbit: The Battle of the Five Armies', writer: 'J.R.R Tolkien', year: '2002', franchise: 'The Hobbit', synopsis: 'Bilbo and Company are forced to engage in a war against an array of combatants and keep the Lonely Mountain from falling into the hands of a ris`
 - `db.movies.find({$or:[{synopsis:{$regex:"dwarves"}}, {synopsis:{$regex:"hobbit"}}])` returns an empty array.
 - `db.movies.find({$and:[{synopsis:{$regex:"gold"}}, {synopsis:{$regex:"dragon"}}])` returns an empty array.
 - `db.movies.remove({_id:ObjectId("5c9f992ae5c2dfe9b3729c00")})` returns `{ acknowledged: true, deletedCount: 0 }`.
 - `db.movies.remove({_id:ObjectId("5c9f9936e5c2dfe9b3729c01")})` returns `{ acknowledged: true, deletedCount: 0 }`.
 - `db.users.insert({_id:1, username:"GoodGuyGreg", first_name:"Good Guy", last_name:"Greg"})` returns `{ acknowledged: true, insertedIds: { '0': 1 } }`.
 - `db.users.insert({_id:2, username:"ScumbagSteve", fullname:{first: "Scumbag", last:"Steve"}})` returns `{ acknowledged: true, insertedIds: { '0': 2 } }`.
 - `db.posts.insert({username:"GoodGuyGreg", title:"Passes out at Party", body:"Raises your credit score"})` returns `{ acknowledged: true, insertedIds: { '0': 2 } }`.

Bottom Screenshot:

- The interface shows the 'test.movies' collection with 6 documents and 1 index.
- The MONGODB shell contains the following commands and results:
 - `db.movies.update({_id:ObjectId("5c9fa42ae5c2dfe9b3729c03")}, {$set:{synopsis:"The dwarves, along with Bilbo Baggins and Gandalf the Grey, continue their ques` returns `{ acknowledged: true, insertedId: null, matchedCount: 0, modifiedCount: 0, upsertedCount: 0 }`.
 - `db.movies.update({_id:ObjectId("5c9f983ce5c2dfe9b3729bfc")}, {$push:{actors:"Samuel L. Jackson"}})` returns `{ acknowledged: true, insertedId: null, matchedCount: 0, modifiedCount: 0, upsertedCount: 0 }`.
 - `db.movies.find({synopsis:{$regex:"Bilbo"}})` returns one document: `{_id: ObjectId("62b2ac9646ebf941a35deb60"), title: 'The Hobbit: The Battle of the Five Armies', writer: 'J.R.R Tolkien', year: '2002', 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 ris`

Assignment 1



The image shows two screenshots of the MongoDB Compass interface, specifically the MONGOSh terminal window, demonstrating database operations on a local MongoDB instance at localhost:27017.

Top Screenshot:

- The interface shows the 'test.movies' collection selected.
- The MONGOSh terminal displays the following commands and results:
 - `db.movies.find({synopsis:{$regex:"Gandalf"}})` returns an empty array `< >`.
 - `db.movies.find({synopsis:{$regex:"Gandalf"}})` returns an empty array `< >`.
 - `db.movies.find({$and:[{synopsis:{$regex:"Bilbo"}}, {synopsis:{$not:/Gandalf/}]})` returns a document:

```
{
  "_id": ObjectId("62b2ac9646ebf941a35deb60"),
  "title": "The Hobbit: The Battle of the Five Armies",
  "writer": "J.R.R Tolkien",
  "year": "2002",
  "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 ris"
}
```
 - `db.movies.find({$or:[{synopsis:{$regex:"dwarves"}}, {synopsis:{$regex:"hobbit"}]})` returns an empty array `< >`.
 - `db.movies.find({$and:[{synopsis:{$regex:"gold"}}, {synopsis:{$regex:"dragon"}]})` returns an empty array `< >`.
 - `db.movies.remove({_id:ObjectId("5c9f992ae5c2dfe9b3729c00")})` returns `< { acknowledged: true, deletedCount: 0 } >`. A deprecation warning is shown: `< DeprecationWarning: Collection.remove() is deprecated. Use deleteOne, deleteMany, findOneAndDelete, or bulkWrite. >`.
 - `db.movies.remove({_id:ObjectId("5c9f9936e5c2dfe9b3729c01")})` returns `< { acknowledged: true, deletedCount: 0 } >`.
 - `db.users.insert({_id:1, username:"GoodGuyGreg", first_name:"Good Guy", last_name:"Greg"})` is executed.

Bottom Screenshot:

- The MONGOSh terminal displays the following commands and results:
 - `db.users.insert({_id:1, username:"GoodGuyGreg", first_name:"Good Guy", last_name:"Greg"})` returns `< { acknowledged: true, insertedIds: { '0': 1 } } >`.
 - `db.users.insert({_id:2, username:"ScumbagSteve", fullname:{first: "Scumbag", last:"Steve"}})` returns `< { acknowledged: true, insertedIds: { '0': 2 } } >`.
 - `db.posts.insert({username:"GoodGuyGreg", title:"Passes out at Party", body:"Raises your credit score"})` returns `< { acknowledged: true, insertedIds: { '0': ObjectId("62b2b8d246ebf941a35deb62") } } >`.
 - `db.posts.insert({username:"GoodGuyGreg", title:"Steals your identity", body:"Raises your credit score"})` returns `< { acknowledged: true, insertedIds: { '0': ObjectId("62b2b8dc46ebf941a35deb63") } } >`.
 - `db.posts.insert({username:"GoodGuyGreg", title:"Reports a bug in your code", body:"Sends you a pull request"})` returns `< { acknowledged: true, insertedIds: { '0': ObjectId("62b2b8ea46ebf941a35deb64") } } >`.
 - `db.posts.insert({username:"ScumbagSteve", title:"Borrows something", body:"Sells it"})` returns `< { acknowledged: true, insertedIds: { '0': ObjectId("62b2b98746ebf941a35deb65") } } >`.
 - `db.posts.insert({username:"ScumbagSteve", title:"Borrows everything", body:"The end"})` returns `< { acknowledged: true, insertedIds: { '0': ObjectId("62b2b99c46ebf941a35deb66") } } >`.
 - `db.posts.insert({username:"ScumbagSteve", title:"Forks your repo on github", body:"Sets to private"})` is executed.

Assignment 1

The image consists of two screenshots of the MongoDB Compass interface, showing a local connection to localhost:27017. The interface includes a sidebar with '4 DBS' and '9 COLLECTIONS', and a main area for the 'test.movies' database.

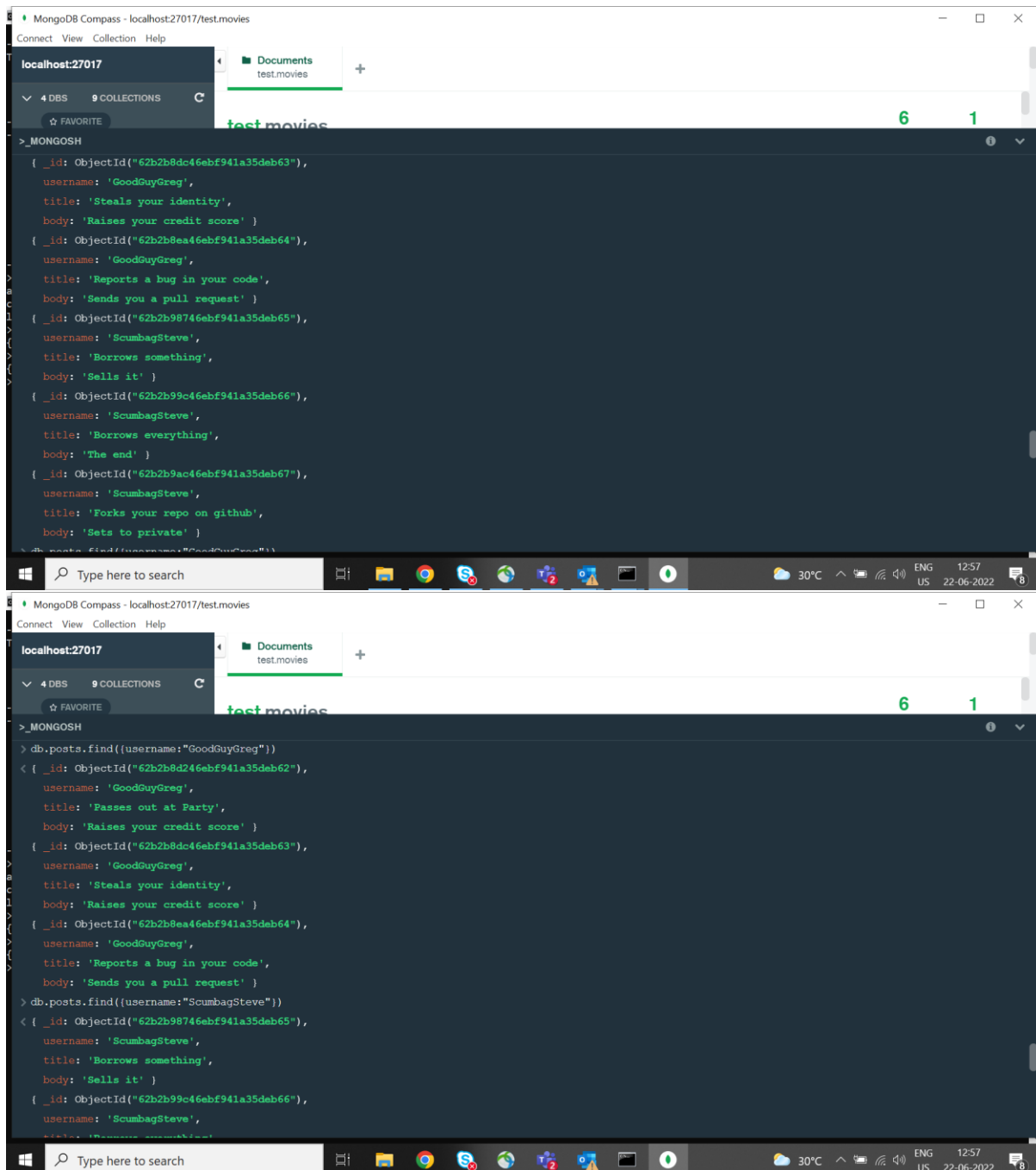
Top Screenshot: The 'test.movies' database is selected. The 'MONGOSH' terminal shows a series of insert and find operations. The results pane on the right shows 6 documents.

```
> db.posts.insert({username:"ScumbagSteve", title:"Forks your repo on github", body:"Sets to private"})
< { acknowledged: true,
  insertedIds: { '0': ObjectId("62b2b9ac46ebf941a35deb67") } }
> db.comments.insert({ username:"GoodGuyGreg", comment:"Hope you got a good deal!", post:ObjectId("5ca0b7e96435f98b5901f463")})
< { acknowledged: true,
  insertedIds: { '0': ObjectId("62b2ba2246ebf941a35deb68") } }
> db.comments.insert({username:"GoodGuyGreg", comment:"What's mine is yours!", post:ObjectId("5ca0b9706435f98b5901f46a")})
< { acknowledged: true,
  insertedIds: { '0': ObjectId("62b2ba3e46ebf941a35deb69") } }
> db.comments.insert({username:"GoodGuyGreg", comment:"Don't violate the licensing agreement!", post:ObjectId("5ca0b8766435f98b5901f467")})
< { acknowledged: true,
  insertedIds: { '0': ObjectId("62b2ba5746ebf941a35deb6a") } }
> db.comments.insert({username:"ScumbagSteve", comment:"It still isn't clean", post:ObjectId("5ca0b8546435f98b5901f466")})
< { acknowledged: true,
  insertedIds: { '0': ObjectId("62b2ba6b46ebf941a35deb6b") } }
> db.comments.insert({username:"ScumbagSteve", comment:"Denied your PR cause I found a hack", post:ObjectId("5ca0b9256435f98b5901f469")})
< { acknowledged: true,
  insertedIds: { '0': ObjectId("62b2ba7a46ebf941a35deb6c") } }
> db.users.find().pretty()
< { _id: 1,
```

Bottom Screenshot: The 'MONGOSH' terminal shows the results of the find operations. The results pane on the right shows 6 documents.

```
> db.users.find().pretty()
< { _id: 1,
  username: 'GoodGuyGreg',
  first_name: 'Good Guy',
  last_name: 'Greg' }
{ _id: 2,
  username: 'ScumbagSteve',
  fullname: { first: 'Scumbag', last: 'Steve' } }
> db.posts.find().pretty()
< { _id: ObjectId("62b2b8d246ebf941a35deb62"),
  username: 'GoodGuyGreg',
  title: 'Passes out at Party',
  body: 'Raises your credit score' }
{ _id: ObjectId("62b2b8dc46ebf941a35deb63"),
  username: 'GoodGuyGreg',
  title: 'Steals your identity',
  body: 'Raises your credit score' }
{ _id: ObjectId("62b2b8ea46ebf941a35deb64"),
  username: 'GoodGuyGreg',
  title: 'Reports a bug in your code',
  body: 'Sends you a null request' }
```


Assignment 1



Assignment 1

The image displays two screenshots of the MongoDB Compass application interface, showing a local connection to localhost:27017. The interface includes a sidebar with '4 DBS' and '9 COLLECTIONS', and a main area for the 'test.movies' collection. The top screenshot shows the results of a query: `db.comments.find().pretty()`. The bottom screenshot shows the results of two specific queries: `db.comments.find({username:"GoodGuyGreg"})` and `db.comments.find({username:"ScumbagSteve"})`.

Top Screenshot Query Results:

```
> db.comments.find().pretty()
< { _id: ObjectId("62b2ba2246ebf941a35deb68"),
  username: 'GoodGuyGreg',
  comment: 'Hope you got a good deal!',
  post: ObjectId("5ca0b7e96435f98b5901f463") }
{ _id: ObjectId("62b2ba3e46ebf941a35deb69"),
  username: 'GoodGuyGreg',
  comment: 'What\'s mine is yours!',
  post: ObjectId("5ca0b9706435f98b5901f46a") }
{ _id: ObjectId("62b2ba5746ebf941a35deb6a"),
  username: 'GoodGuyGreg',
  comment: 'Don\'t violate the licensing agreement!',
  post: ObjectId("5ca0b8766435f98b5901f467") }
{ _id: ObjectId("62b2ba6b46ebf941a35deb6b"),
  username: 'ScumbagSteve',
  comment: 'It still isn\'t clean',
  post: ObjectId("5ca0b8546435f98b5901f466") }
{ _id: ObjectId("62b2ba7a46ebf941a35deb6c"),
  username: 'ScumbagSteve',
  comment: 'Denied your PR cause I found a hack',
  post: ObjectId("5ca0b8546435f98b5901f466") }
```

Bottom Screenshot Query Results:

```
> db.comments.find({username:"GoodGuyGreg"})
< { _id: ObjectId("62b2ba2246ebf941a35deb68"),
  username: 'GoodGuyGreg',
  comment: 'Hope you got a good deal!',
  post: ObjectId("5ca0b7e96435f98b5901f463") }
{ _id: ObjectId("62b2ba3e46ebf941a35deb69"),
  username: 'GoodGuyGreg',
  comment: 'What\'s mine is yours!',
  post: ObjectId("5ca0b9706435f98b5901f46a") }
{ _id: ObjectId("62b2ba5746ebf941a35deb6a"),
  username: 'GoodGuyGreg',
  comment: 'Don\'t violate the licensing agreement!',
  post: ObjectId("5ca0b8766435f98b5901f467") }

> db.comments.find({username:"ScumbagSteve"})
< { _id: ObjectId("62b2ba6b46ebf941a35deb6b"),
  username: 'ScumbagSteve',
  comment: 'It still isn\'t clean',
  post: ObjectId("5ca0b8546435f98b5901f466") }
{ _id: ObjectId("62b2ba7a46ebf941a35deb6c"),
  username: 'ScumbagSteve',
  comment: 'Denied your PR cause I found a hack',
  post: ObjectId("5ca0b8546435f98b5901f466") }
```

Assignment 1

MongoDB Compass - localhost:27017/test.movies

Connect View Collection Help

localhost:27017

4 DBS 9 COLLECTIONS

FAVORITE

My Queries

Databases

Filter your data

admin

config

local

test

- Movies
- Users
- comments
- movies
- posts
- users

Documents test.movies

test.movies

6 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER { field: 'value' }

ADD DATA VIEW

Displaying documents 1 - 6 of 6

REFRESH

```
{
  "_id": ObjectId("62b2abb346ebf941a35deb5c"),
  "title": "Fight Club",
  "writer": "Chuck Palahniuk",
  "year": "1999",
  "actors": Array
}
```

```
{
  "_id": ObjectId("62b2ac1d46ebf941a35deb5d"),
  "title": "Inglorious Basterds",
  "writer": "Quentin Tarantino",
  "year": "2009",
  "actors": Array
}
```

```
{
  "_id": ObjectId("62b2ac7846ebf941a35deb5e"),
  "title": "The Hobbit: An unexpected Journey",
  "writer": "J.R.R. Tolkein",
  "year": "2012"
}
```

>_MONGOSH

Type here to search

MongoDB Compass - localhost:27017/test.movies

Connect View Collection Help

localhost:27017

4 DBS 9 COLLECTIONS

FAVORITE

My Queries

Databases

Filter your data

admin

config

local

test

- Movies
- Users
- comments
- movies
- posts
- users

Documents test.movies

test.movies

6 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER { field: 'value' }

ADD DATA VIEW

Displaying documents 1 - 6 of 6

REFRESH

```
{
  "_id": ObjectId("62b2ac8646ebf941a35deb5f"),
  "title": "The Hobbit: The Desolation of Smaug",
  "writer": "J.R.R. Tolkien",
  "year": "2013",
  "franchise": "The Hobbit"
}
```

```
{
  "_id": ObjectId("62b2ac9646ebf941a35deb60"),
  "title": "The Hobbit: The Battle of the Five Armies",
  "writer": "J.R.R. Tolkien",
  "year": "2012",
  "franchise": "The Hobbit",
  "synopsis": "Bilbo and Company are forced to engage in a war against an array of co..."
}
```

```
{
  "_id": ObjectId("62b2ad4046ebf941a35deb61"),
  "title": "Avatar"
}
```

>_MONGOSH

Type here to search

Assignment 1

MongoDB Compass - localhost:27017/test.users

Connect View Collection Help

localhost:27017

4 DBS 9 COLLECTIONS

FAVORITE

My Queries

Databases

Filter your data

admin

config

local

test

Movies

Users

comments

movies

posts

users

+

Documents test.users

test.users

2 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER { field: 'value' }

OPTIONS FIND RESET

ADD DATA VIEW

Displaying documents 1 - 2 of 2

REFRESH

```
{
  "_id": 1,
  "username": "GoodGuyGreg",
  "first_name": "Good Guy",
  "last_name": "Greg"
}
```

```
{
  "_id": 2,
  "username": "ScumbagSteve",
  "fullname": Object
}
```

>_MONGOSH

MongoDB Compass - localhost:27017/test.posts

Connect View Collection Help

localhost:27017

4 DBS 9 COLLECTIONS

FAVORITE

My Queries

Databases

Filter your data

admin

config

local

test

Movies

Users

comments

movies

posts

users

+

Documents test.posts

test.posts

6 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER { field: 'value' }

OPTIONS FIND RESET

ADD DATA VIEW

Displaying documents 1 - 6 of 6

REFRESH

```
{
  "_id": ObjectId('62b2b8d46ebf941a35deb62'),
  "username": "GoodGuyGreg",
  "title": "PASSES out at Party",
  "body": "Raises your credit score"
}
```

```
{
  "_id": ObjectId('62b2b8dc46ebf941a35deb63'),
  "username": "GoodGuyGreg",
  "title": "Steals your identity",
  "body": "Raises your credit score"
}
```

```
{
  "_id": ObjectId('62b2b8ea46ebf941a35deb64'),
  "username": "GoodGuyGreg",
  "title": "Reports a bug in your code",
  "body": "Sends you a pull request"
}
```

>_MONGOSH

Assignment 1

MongoDB Compass - localhost:27017/test.posts

Connect View Collection Help

localhost:27017

4 DBS 9 COLLECTIONS

FAVORITE

My Queries

Databases

Filter your data

admin

config

local

test

Movies

Users

comments

movies

posts

users

Documents

test.posts

test.posts

6 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER { field: 'value' }

OPTIONS FIND RESET

ADD DATA VIEW

Displaying documents 1 - 6 of 6

REFRESH

```
{ "_id": ObjectId("62b2b99746ebf941a35deb65"), "username": "ScumbagSteve", "title": "Borrows something", "body": "Sells it" }
```

```
{ "_id": ObjectId("62b2b99c46ebf941a35deb66"), "username": "ScumbagSteve", "title": "Borrows everything", "body": "The end" }
```

```
{ "_id": ObjectId("62b2b9ac46ebf941a35deb67"), "username": "ScumbagSteve", "title": "Forks your repo on github", "body": "Sets to private" }
```

>_MONGOSH

Type here to search

MongoDB Compass - localhost:27017/test.comments

Connect View Collection Help

localhost:27017

4 DBS 9 COLLECTIONS

FAVORITE

My Queries

Databases

Filter your data

admin

config

local

test

Movies

Users

comments

movies

posts

users

Documents

test.comments

test.comments

1 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER { field: 'value' }

OPTIONS FIND RESET

ADD DATA VIEW

Displaying documents 1 - 5 of 5

REFRESH

```
{ "_id": ObjectId("62b2ba2246ebf941a35deb68"), "username": "GoodGuyDreg", "comment": "Hope you got a good deal!", "post": ObjectId("5ca0b7e96435f98b5901f463") }
```

```
{ "_id": ObjectId("62b2ba3e46ebf941a35deb69"), "username": "GoodGuyDreg", "comment": "What's mine is yours!", "post": ObjectId("5ca0b706435f98b5901f46a") }
```

```
{ "_id": ObjectId("62b2ba5746ebf941a35deb6a"), "username": "GoodGuyDreg", "comment": "Don't violate the licensing agreement!", "post": ObjectId("5ca0b766435f98b5901f467") }
```

>_MONGOSH

Type here to search

Assignment 1

MongoDB Compass - localhost:27017/test.comments

Connect View Collection Help

localhost:27017

4 DBS 9 COLLECTIONS

FAVORITE

My Queries

Databases

Filter your data

admin

config

local

test

Movies

Users

comments

movies

posts

users

Documents

test.comments

test.comments

1 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER { field: 'value' }

OPTIONS FIND RESET

ADD DATA VIEW

Displaying documents 1 - 5 of 5

REFRESH

```
{ "_id": ObjectId("62b2ba5746ebf941a35deb6a"), "username": "GoodGuyGreg", "comment": "Don't violate the licensing agreement!", "post": ObjectId("5ca0b8766435f98b5901f467") }
```

```
{ "_id": ObjectId("62b2ba6b46ebf941a35deb6b"), "username": "ScumbagSteve", "comment": "It still isn't clean", "post": ObjectId("5ca0b8546435f98b5901f466") }
```

```
{ "_id": ObjectId("62b2ba7a46ebf941a35deb6c"), "username": "ScumbagSteve", "comment": "Denied your PR cause I found a hack", "post": ObjectId("5ca0b9256435f98b5901f469") }
```

>_MONGOSH

Type here to search

30°C 12:59 22-06-2022