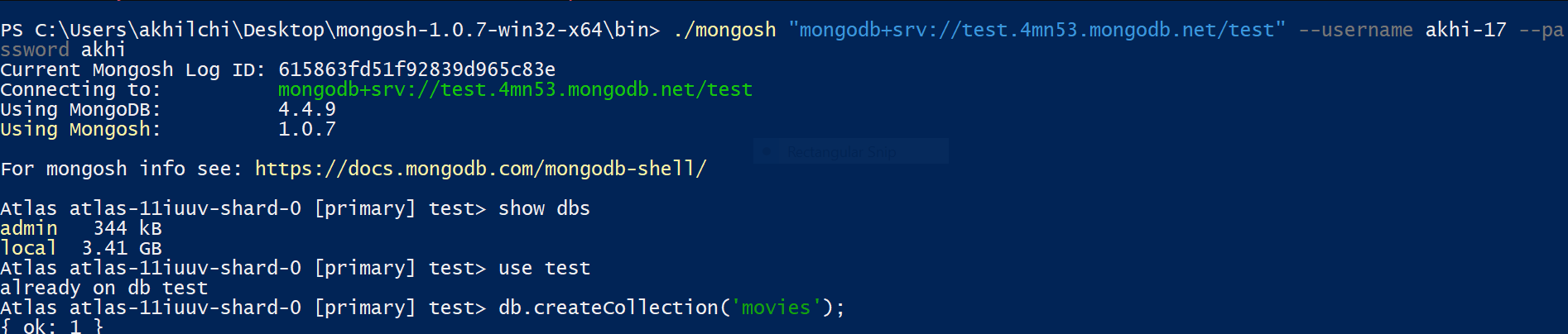
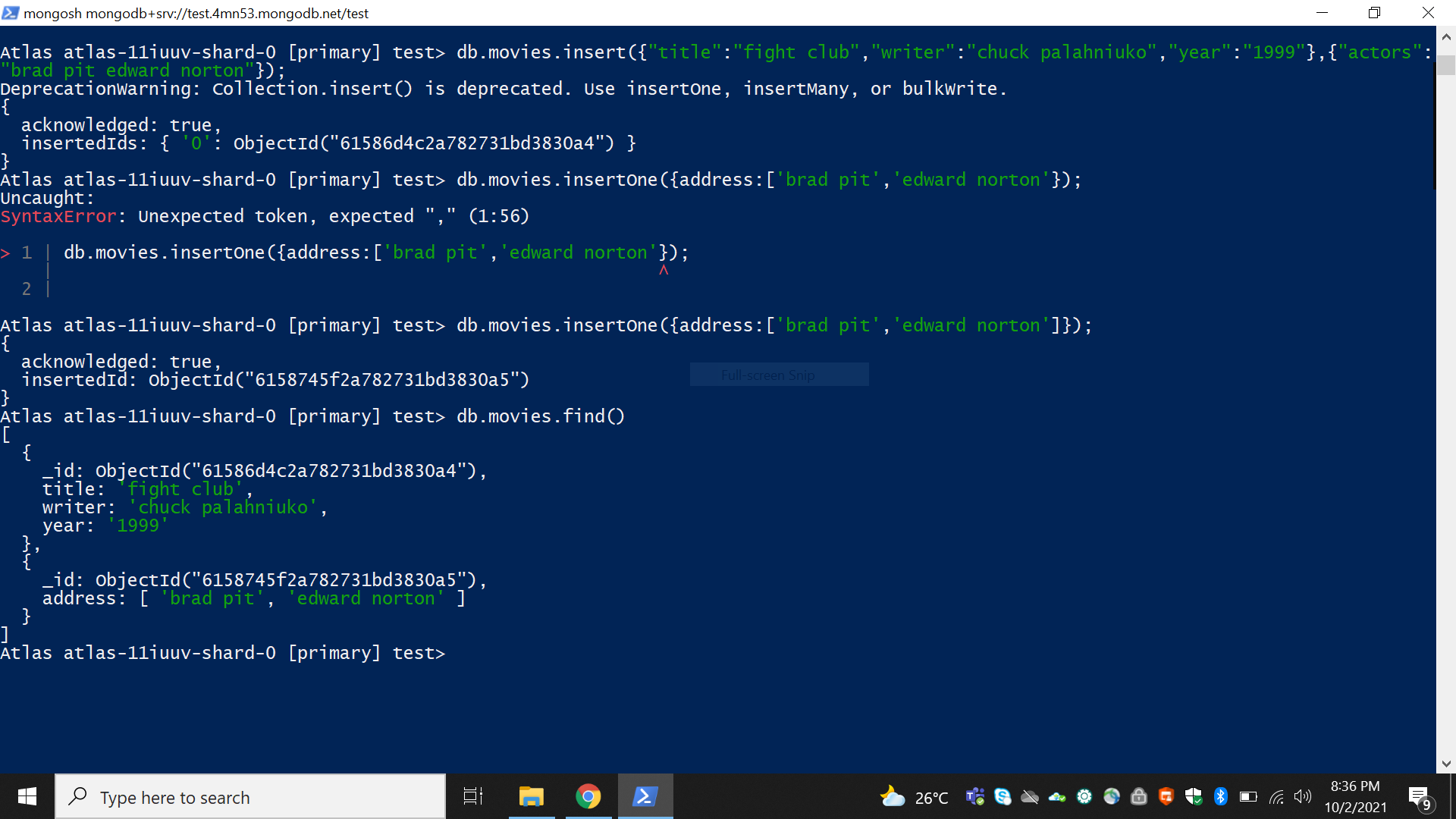
**MongoDB Lab Assignments -Day 1**

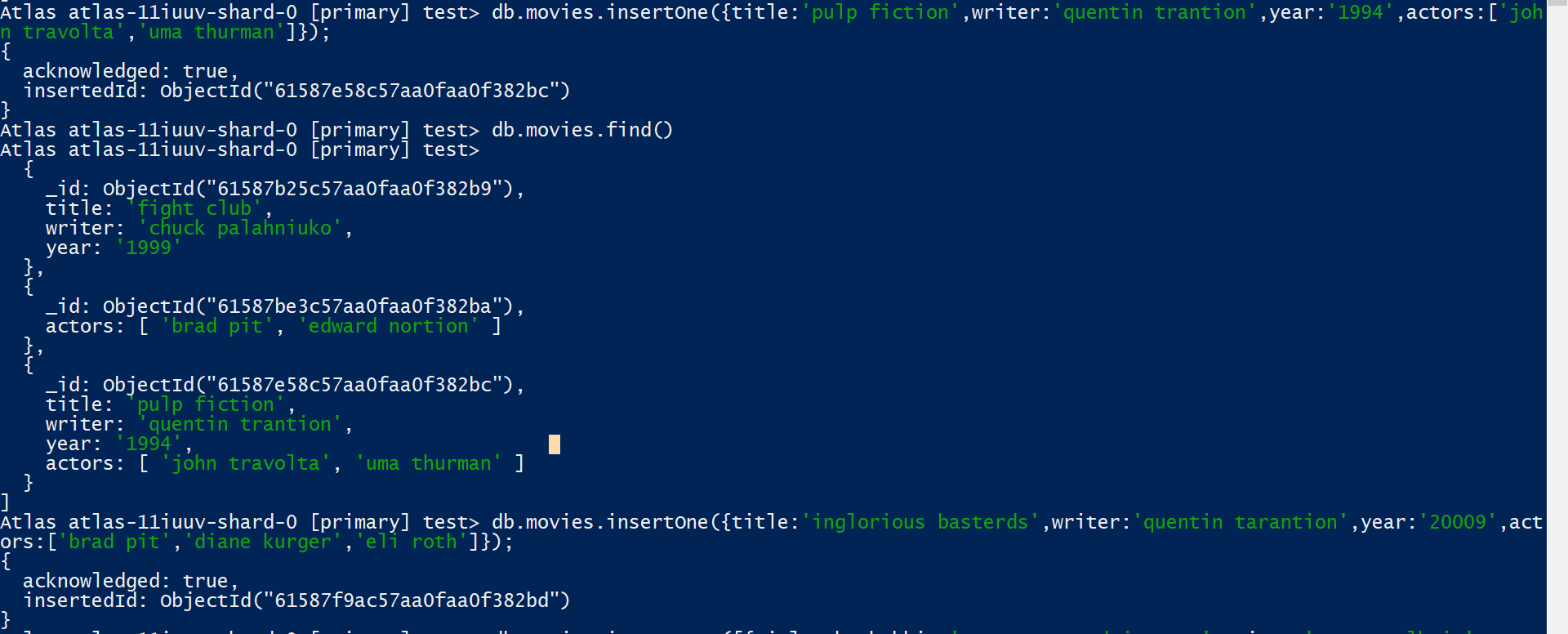
# MongoDB Exercise in mongo shell

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

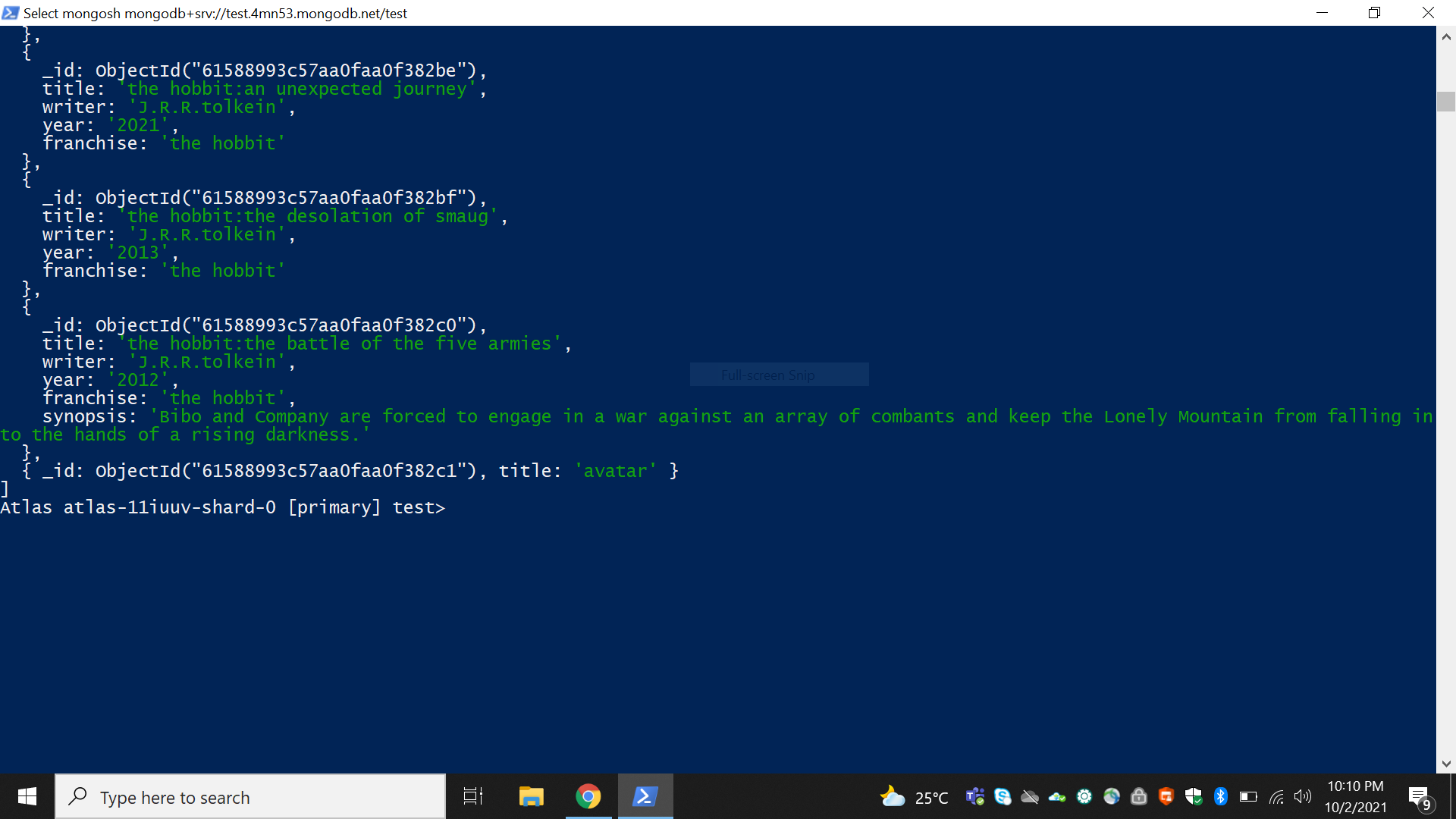
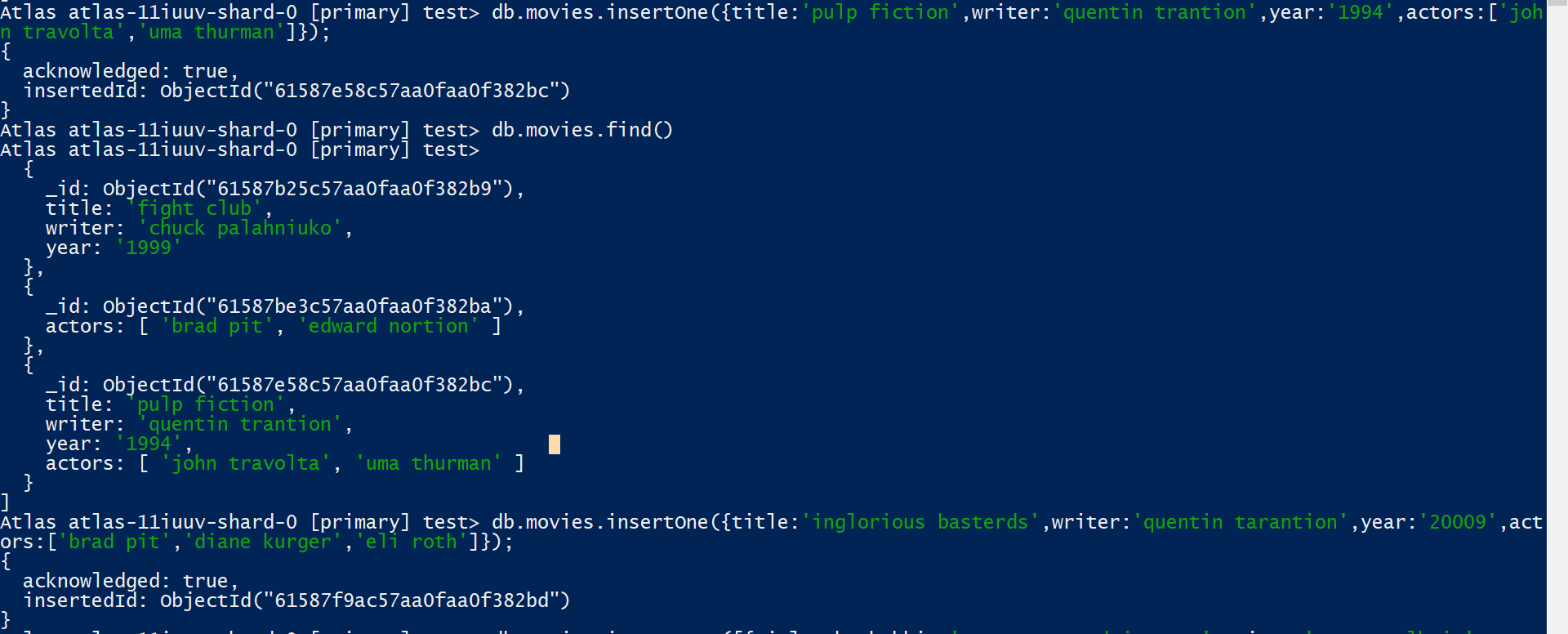
Insert the following documents into a **movies** collection. 

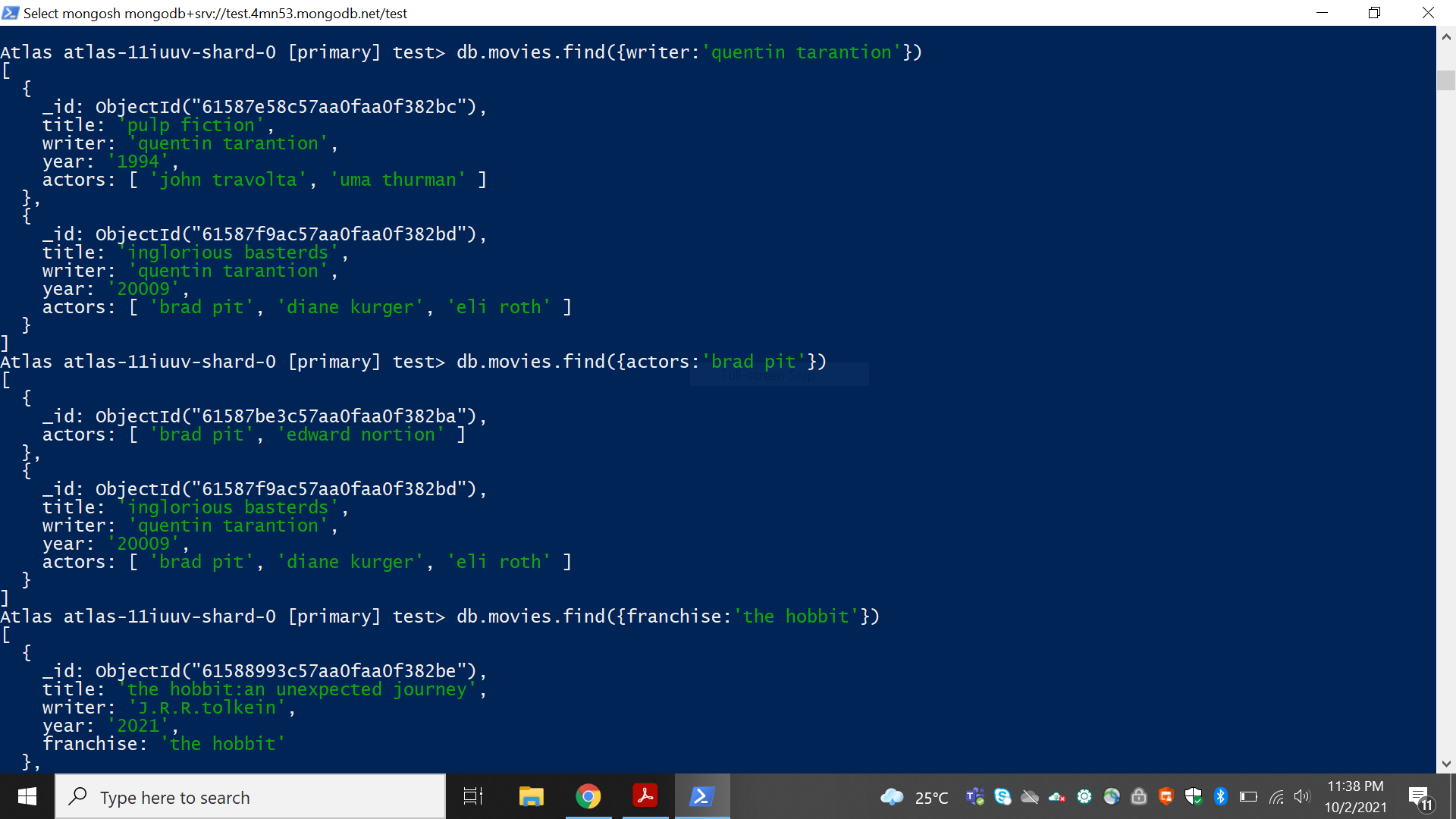


|  |
| --- |
| title : Fight Club writer : Chuck Palahniuko year : 1999 actors : [  Brad Pitt  Edward Norton  ]    title : Pulp Fiction writer : Quentin Tarantino year : 1994 actors : [  John Travolta  Uma Thurman  ]    title : Inglorious Basterds writer : Quentin Tarantino year : 2009 actors : [  Brad 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 
2. get all documents with writer set to "Quentin Tarantino"

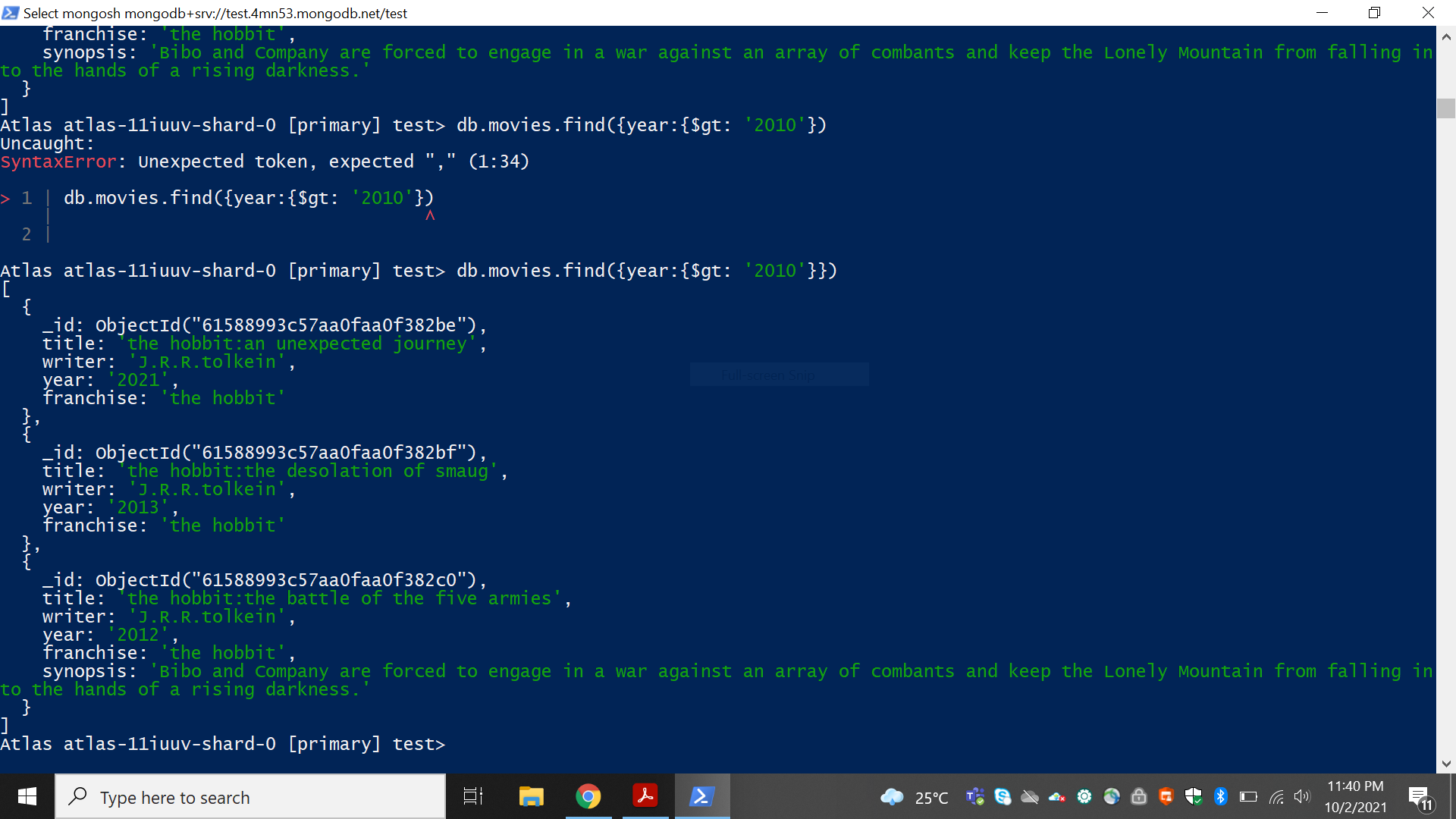
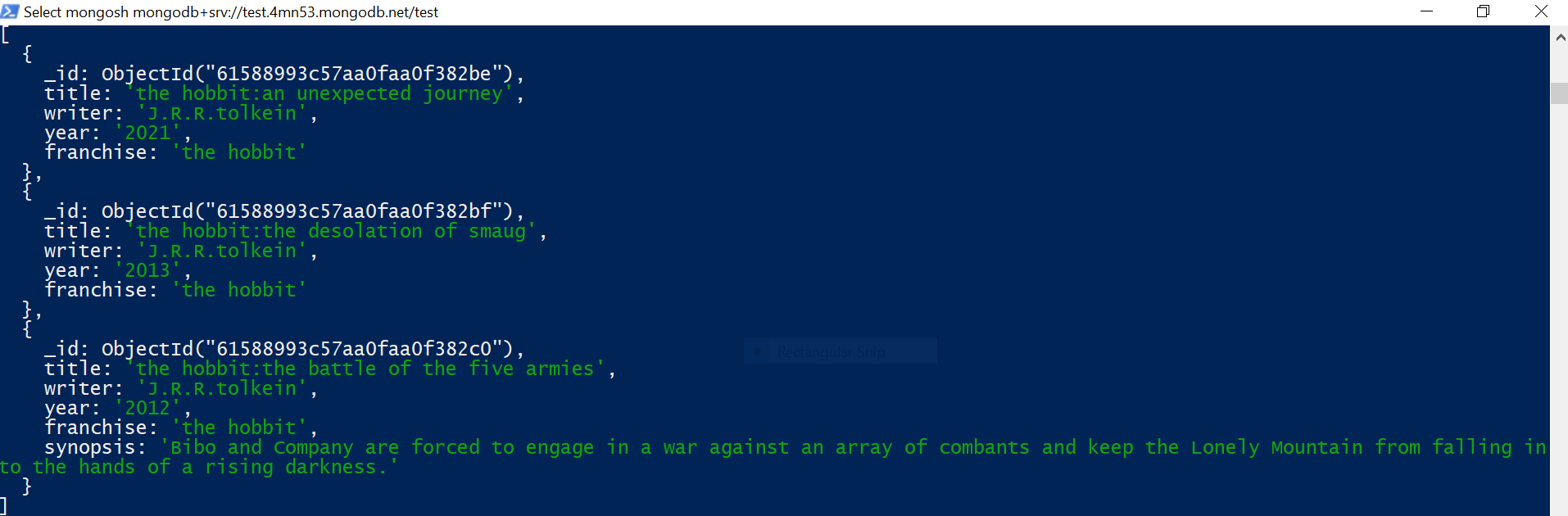
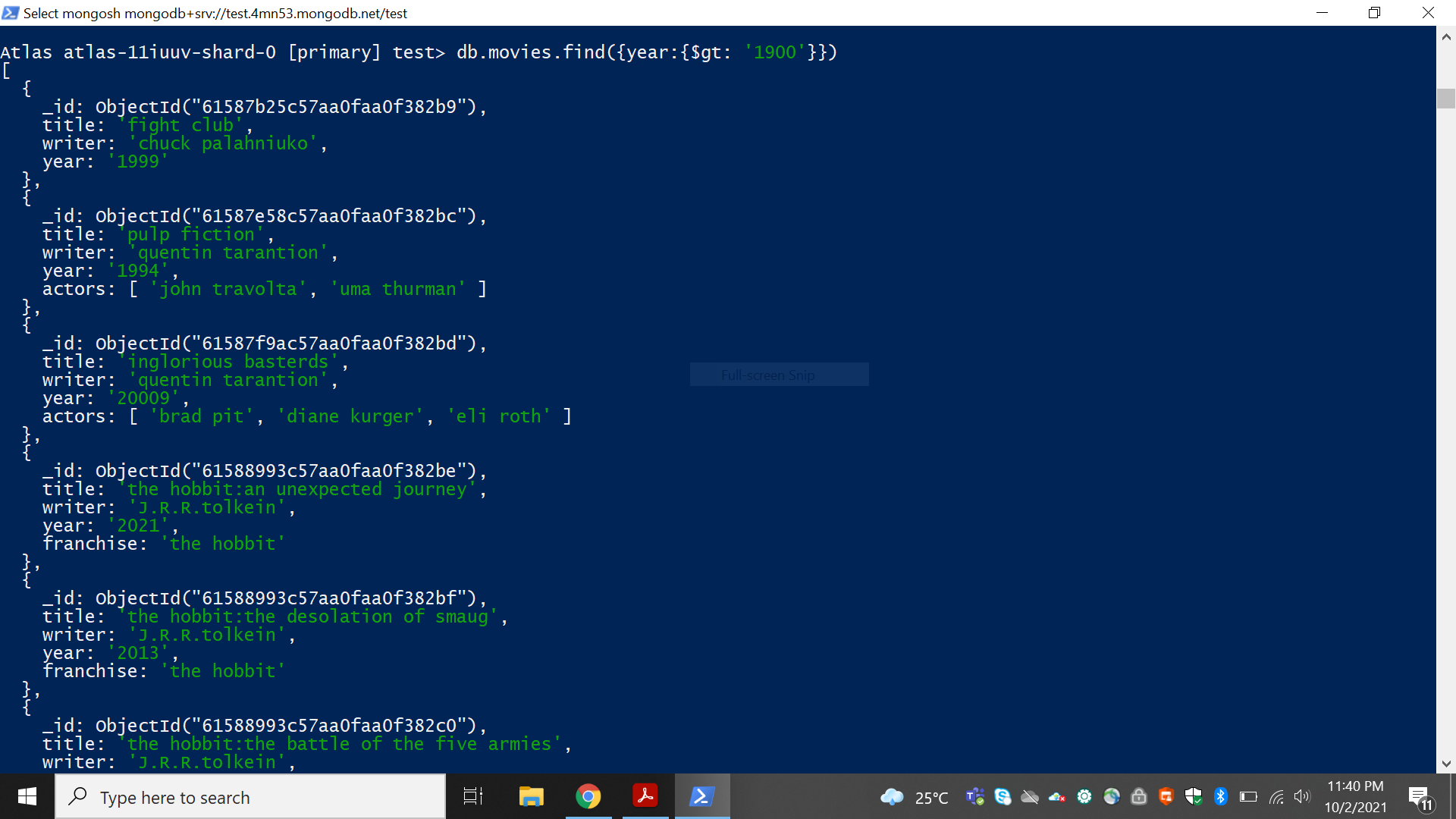


3.get all documents where actors include "Brad Pitt"

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

5.get all movies released in the 90s

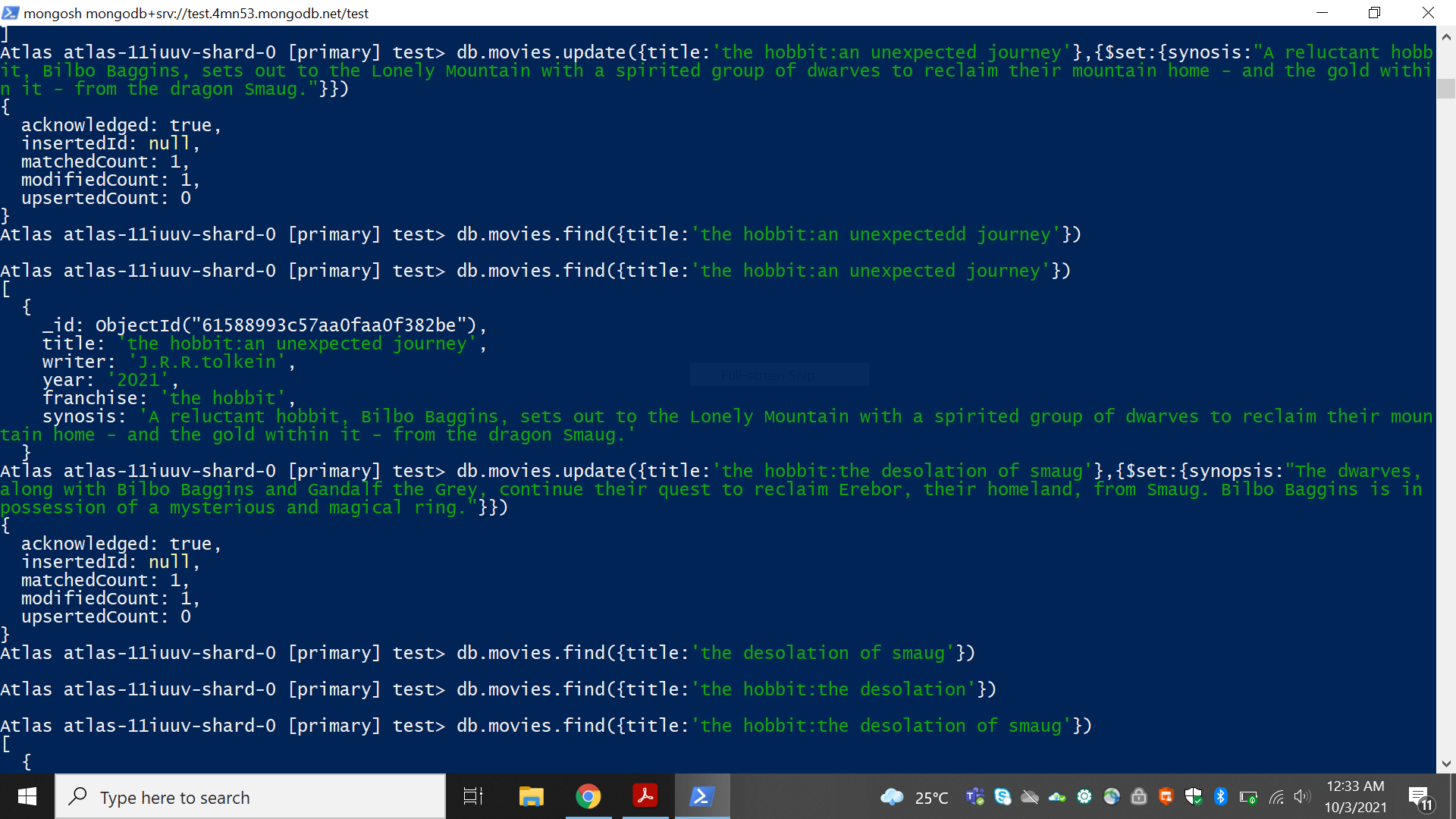
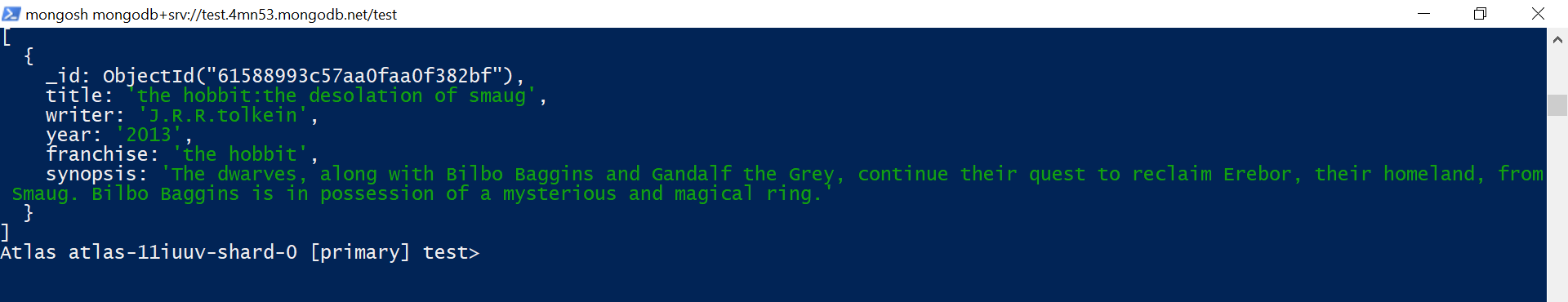
6.get all movies released before the year 2000 or after 2010



Reference:

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

# Update Documents

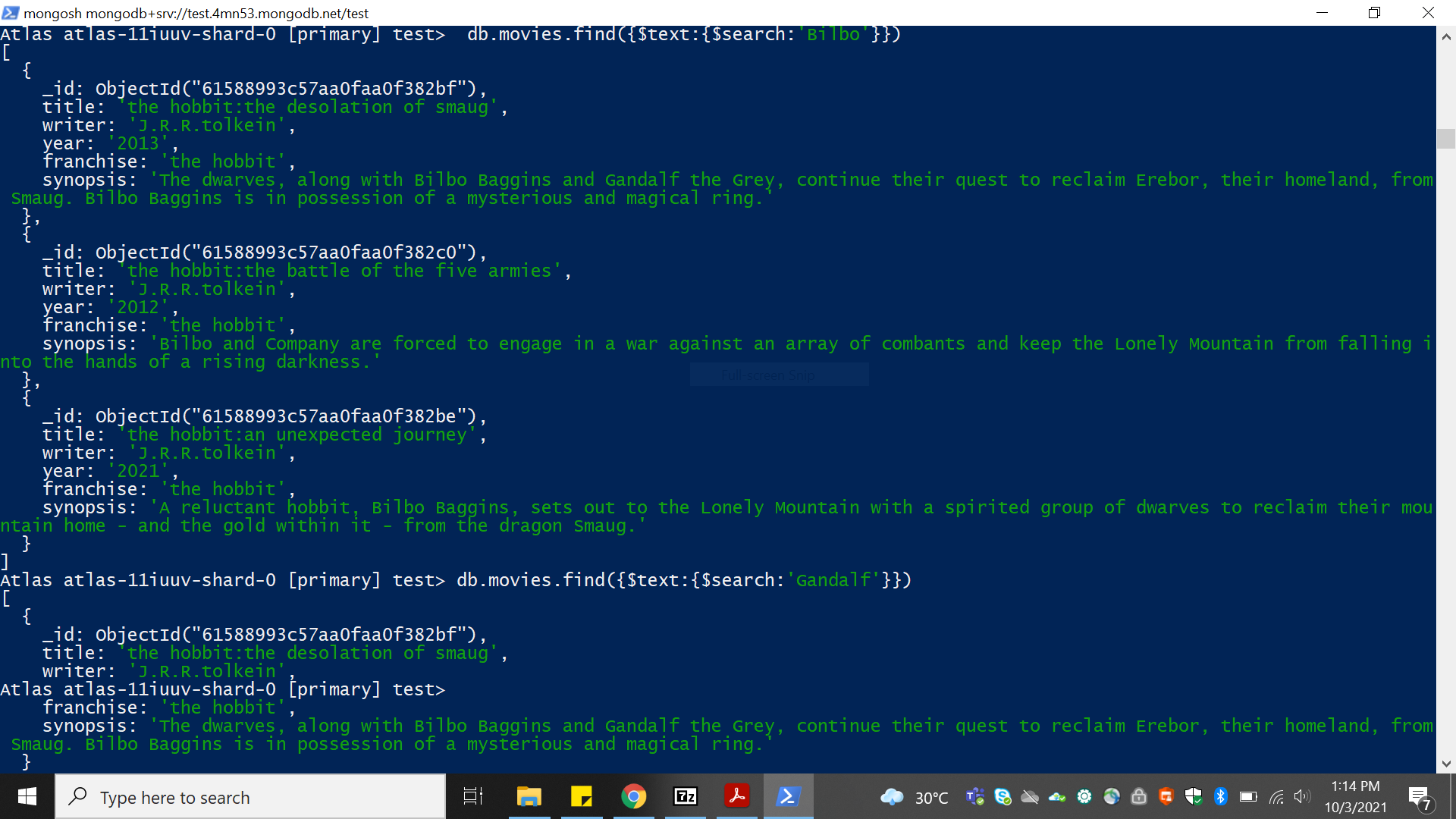
1. add a synopsis to "The Hobbit: An Unexpected Journey" : "A reluctant hobbit, Bilbo Baggins, sets out to the Lonely Mountain with a spirited group of dwarves to reclaim their mountain home - and the gold within it - from the dragon Smaug."
2. add a synopsis to "The Hobbit: The Desolation of Smaug" : "The dwarves, along with Bilbo Baggins and Gandalf the Grey, continue their quest to reclaim Erebor, their homeland, from Smaug. Bilbo 
3. Baggins is in possession of a mysterious and magical ring."
4. add an actor named "Samuel L. Jackson" to the movie "Pulp Fiction"

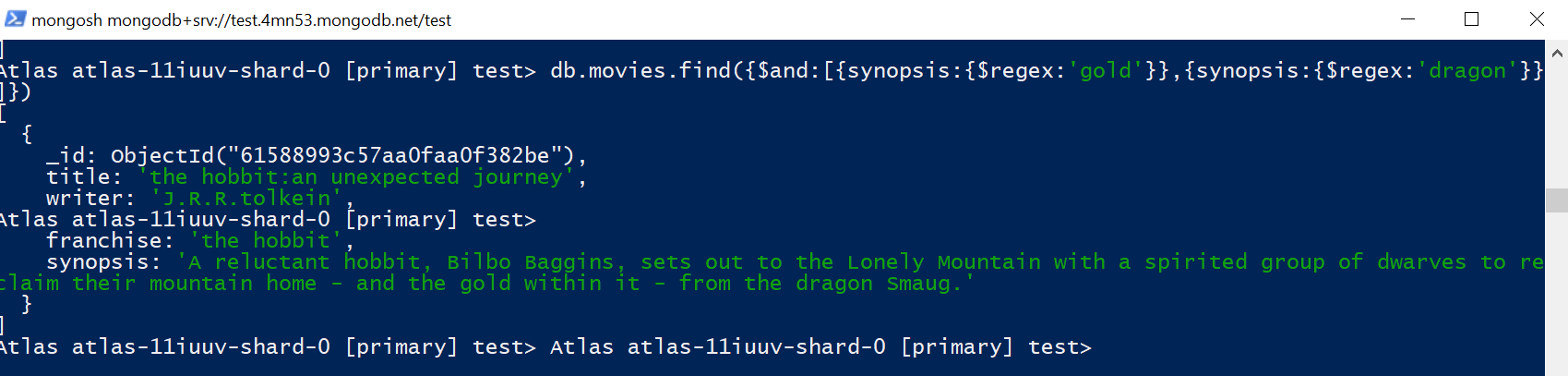
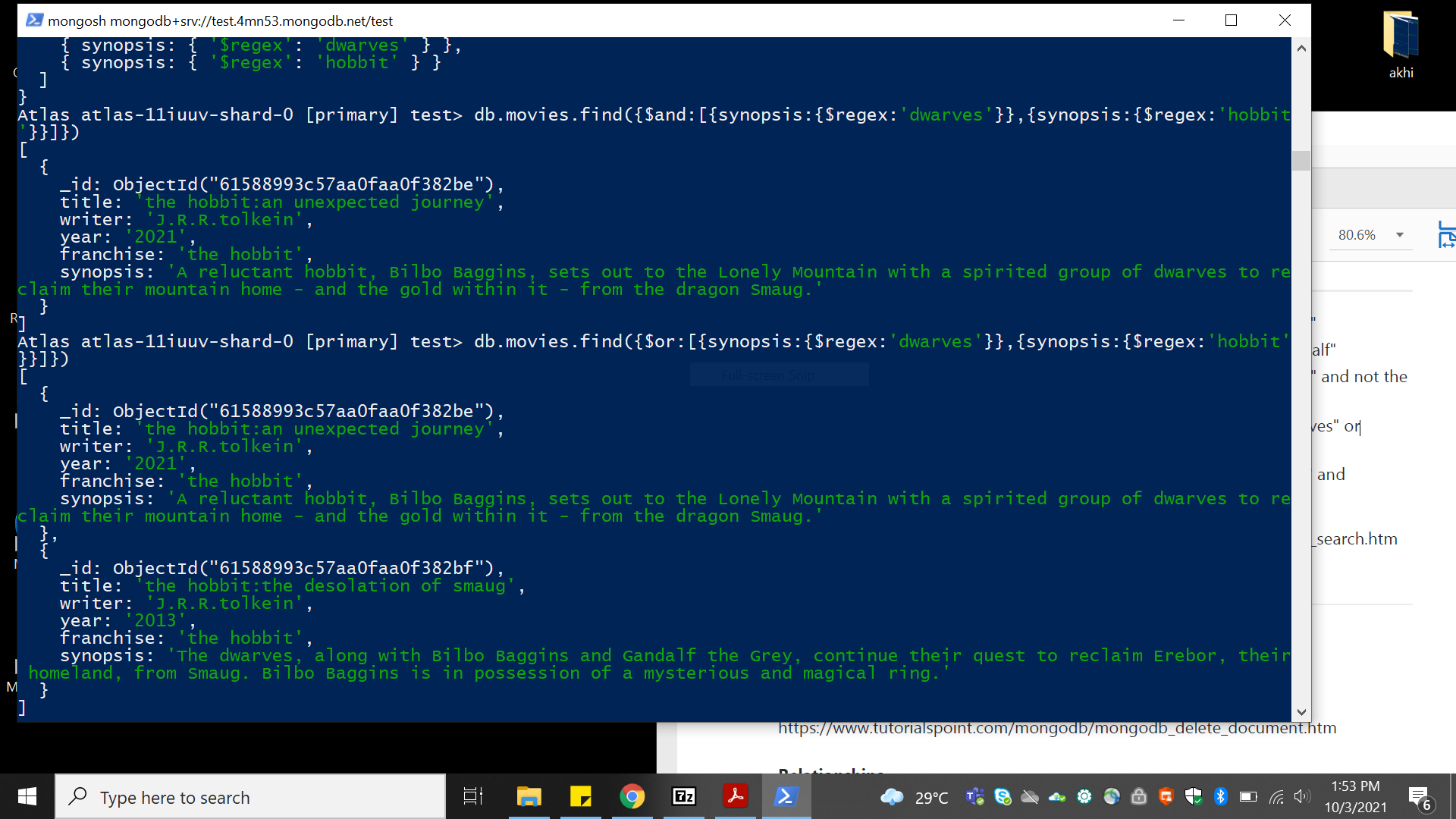
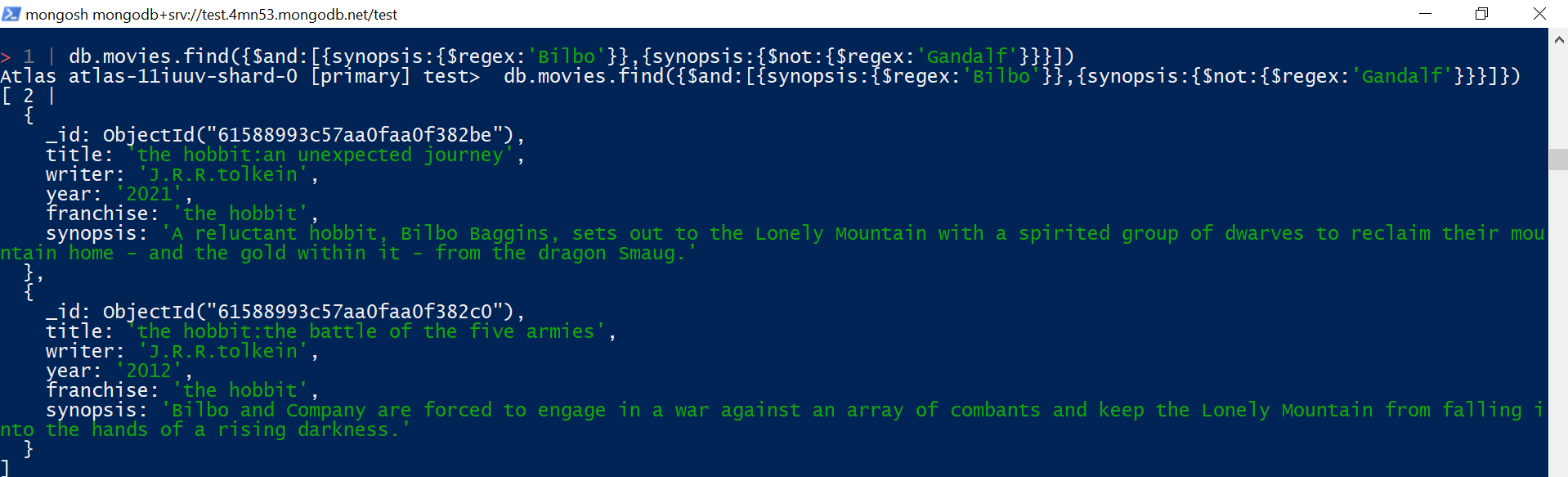
Reference:

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

# Text Search

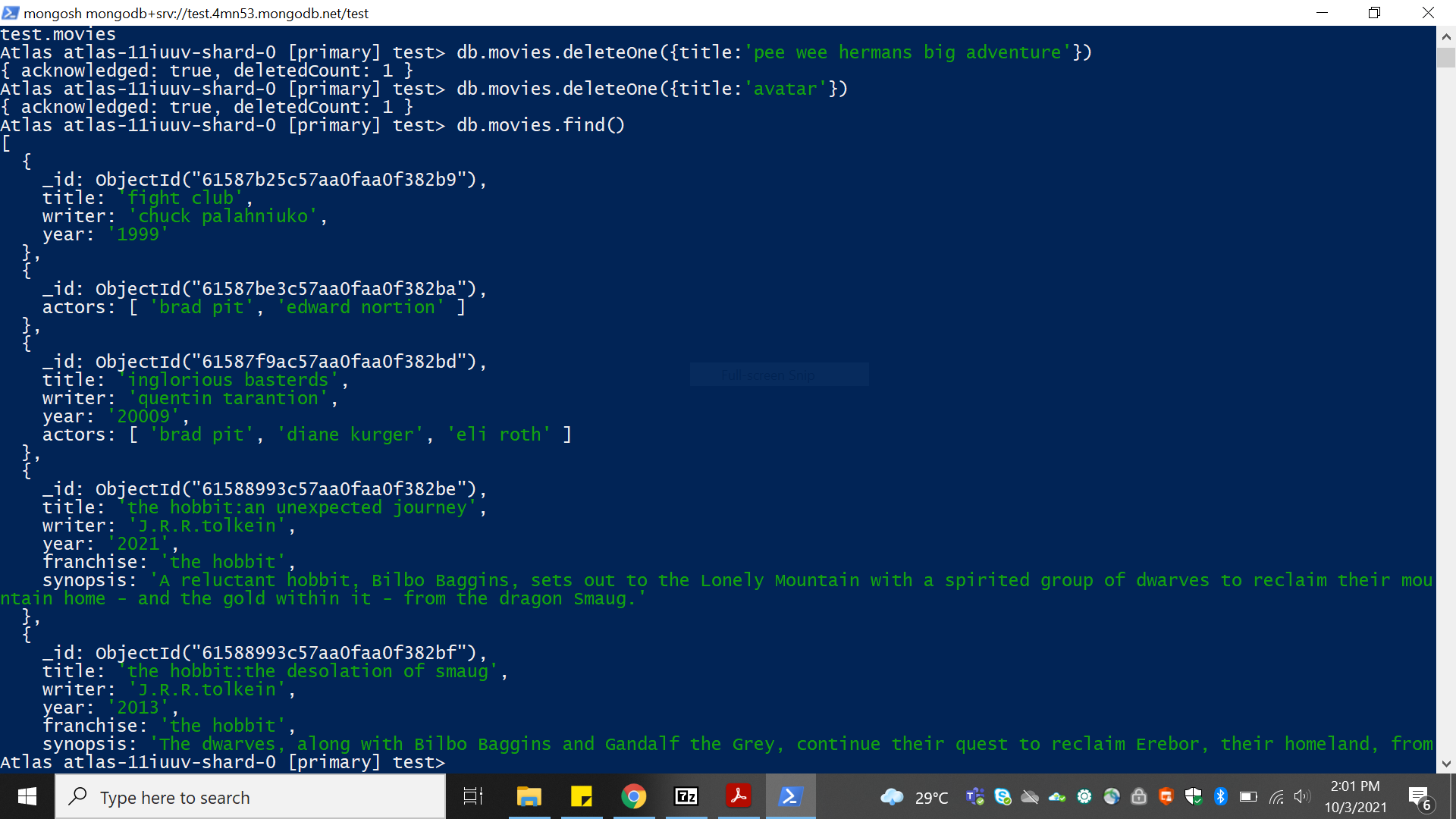
1. find all movies that have a synopsis that contains the word "Bilbo"
2. find all movies that have a synopsis that contains the word "Gandalf"
3. find all movies that have a synopsis that contains the word "Bilbo" and not the word "Gandalf"
4. find all movies that have a synopsis that contains the word "dwarves" or "hobbit"
5. find all movies that have a synopsis that contains the word "gold" and "dragon"





**Delete Documents**

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



# Relationships

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

|  |
| --- |
| db.users.insertMany([  {  Username:”goodguygreg”,  Firstname:”good guy”  Last\_name:”greg”  },  {  Username:’scumbagsteve’,  Fullname:{  Firsrt:”scumbag”  Body:’steve’  },  }  ]); |

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

|  |
| --- |
| b.posts.insertMany([  {  Username:’goodguygreg’  Title:’passes out at party’  Body:’wakesup early and cleans house’  },  {  Username:’goodguygreg’  Title:’steals your identity’  Body:’rise your credit score’  },  {  Username:’goodguygreg’  Title:’report a bug in your code’  Body:’sends you a pull request’  };  {  Username:’sccumbagdsteve’  Title:’borrows something’  Body;’sells it’  },  {  Username:’scumbagsteve’  Title:’borrows everything’  Body:’the end’  },  {  Username:’scumbagsteve’  Title:’forks you repo on git hub’  Body:’sets to private’  }  ]); |

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

Db.comments.insertMany([

{

Username:’goodguygreg’,

Comment:’hope you got a good deal’,

Post:Objectid(‘5f44d3a148197d7749864def’)

},

{

Username:’goodguygreg’

Comment:”don’t violate the licensing agreement!”,

Post:Objectid(‘5f44d3a148197d7749864def’)

},

{

Username:’goodguygreg’

Comment:”don’t violate the licensing agreement!”,

Post:Objectid(‘5f44d3a148197d7749864def’

)

},

{

Username:’scumbagsteve’

Comment:”it still isn’t clean”,

Post:Objectid(‘5f44d3a148197d7749864def’) },

{

Username:’scumbagsteve’

Comment:”denied your PR cause I found a hack”,

Post:Objectid(‘5f44d3a148197d7749864def’)

}

]);

# Querying related collections

1. find all users

db.user.find().pretty();

1. find all posts

db.posts.find().pretty();

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

db.posts.find({username:’goodguygreg’}).pretty();

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

db.posts.find({username:’scumbagsteve’}).pretty();

1. find all comments

db.comments.find().pretty();

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

db.comments.find({username:’goodguygreg’}).prertty();

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

db.comments.find({username:’scumbagsteve’}).pretty();

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

db.posts.aggregate([

{

$match:{title:’reports a bug in your code’}

},

{

$lookup:{

From:’comments’

localField:’\_id’

foreigenField:’posts’,

as:’comments’

}

}

]).pretty();

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