Assignment - 1

1.

```
db.movies.insert([
        {
                title : "Fight Club",
                writer : "Chuck Palahniuk",
                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"
        }
])


2.db.movies.find().pretty()

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

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

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

6. db.movies.find({"year":{$gte:1990 ,$lte:1999}}).pretty()

7. db.movies.find({$or:[{year:{$lte:2000}}, {year:{$gte:2010}}]}).pretty()


8.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."}})

9. 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."}})

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

11. db.movies.find({synopsis: /.Bilbo./}).pretty()

12.db.movies.find({synopsis: /.Gandalf./}).pretty()

13. db.movies.find({$and: [{synopsis: /.Bilbo./}, {synopsis:
/.^Gandalf./}]}).pretty()

14. db.movies.find({$and: [{synopsis: /Bilbo/}, {synopsis: {$not:
/Gandalf/}}]}).pretty()

15. db.movies.find({$or: [{synopsis: /dwarves/}, {synopsis:
/hobbit/}]}).pretty()
```

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

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

18. db.users.find({}).pretty()

19. db.posts.find({}).pretty()

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

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

22. db.comments.find({}).pretty()

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

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

25. db.comments.find({post: db.posts.findOne({title: "Reports a bug in your
code"})._id})


Assignment - 2
---------------
Atlanta Population
-------------------
1 db.zipcodes.find({$and:[{city:"ATLANTA"},{state:"GA"}]})
2 db.zipcodes.aggregate([{$match: {city:"ATLANTA",state:"GA"}}])
3 db.zipcodes.aggregate([{$group: {_id: {city:"$city"},count:{"$sum":1}}},
{$match: {"_id.city":"ATLANTA"}}])
4 db.zipcodes.aggregate([{$match: { city:"ATLANTA" }}, {$group: { _id:{city:
"$city"},TotalPop:{$sum:"$pop"}}}])
Populations By State
1 db.zipcodes.aggregate([{$group: {_id:{State:
"$state"},TotalPop:{$sum:"$pop"}}}])
2 db.zipcodes.aggregate([{$group: {_id:{State: "$state"},TotPop:{$sum:"$pop"}}},
{$sort: { TotPop: -1}}])
3 db.zipcodes.aggregate([{$group: {_id:{State: "$state"},TotPop:{$sum:"$pop"}}},
{$sort: { TotPop: -1}},{$limit:3}]

Populations by City
------------------
1 db.zipcodes.aggregate([{$group: {_id:{City:
"$city",State:"$state"},TotPop:{$sum:"$pop"}}}])
2 db.zipcodes.aggregate([{$group: {_id:{City:
"$city",State:"$state"},TotPop:{$sum:"$pop"}}},{$sort:{ TotPop: -1}])
3 db.zipcodes.aggregate([{$group: {_id:{City:
"$city",State:"$state"},TotPop:{$sum:"$pop"}}},{$sort:{ TotPop: -1},{$limit:3}])
4 db.zipcodes.aggregate([{$group: {_id:{City:
"$city",State:"$state"},TotPop:{$sum:"$pop"}}},
        {$match: {"_id.State":"TX"}}, {$sort: {TotPop:-1}}, {$limit: 3}])

Bonus
```

```
1 db.zipcodes.aggregate([{$group:{_id:{State:"$state"},AvgPop:{$avg:"$pop"}}}])
2  db.zipcodes.aggregate([{$group:
{_id:{State:"$state",City:"$city"},AvgPop:{$avg:"$pop"}}},
        {$sort: { AvgPop:-1}}, {$limit: 3}])
```

Assignment - 3

```
1.db.addresses.find().pretty()


2. db.addresses.find({},{"restaurant_id" : 1,"name":1,"borough":1,"cuisine"
:1}).pretty();


3. db.addresses.find({},{"restaurant_id" : 1,"name":1,"borough":1,"cuisine"
:1,"_id":0}).pretty()

4.db.addresses.find({},{"restaurant_id" :
1,"name":1,"borough":1,"address.zipcode" :1,"_id":0}).pretty()

5.db.addresses.find({"borough": "Bronx"}).limit(5).pretty()

6.db.addresses.find({"borough": "Bronx"});

7.db.addresses.find({"borough": "Bronx"}).skip(5).limit(5);

8.  db.addresses.find({grades : { $elemMatch:{"score":{$gt : 90}}}});

9.db.addresses.find({grades : { $elemMatch:{"score":{$gt : 80 , $lt :100}}}});

10. db.addresses.find({"address.coord" : {$lt : -95.754168}});

11.db.addresses.find({$and:[{"cuisine" : {$ne :"American "}},{"grades.score" :
{$gt : 70}},{"address.coord" : {$lt : -65.754168}}]}).pretty()

12.db.addresses.find({"cuisine" : {$ne : "American "},"grades.score" :{$gt:
70},"address.coord" : {$lt : -65.754168}}).pretty()

13.db.addresses.find( {"cuisine" : {$ne : "American "},"grades.grade"
:"A","borough": {$ne : "Brooklyn"}}).sort({"cuisine":-1}).pretty()

14.db.addresses.find({name: /^Wil/},{"restaurant_id" :
1,"name":1,"borough":1,"cuisine" :1}).pretty();

15.db.addresses.find({name: /ces$/},{"restaurant_id" :
1,"name":1,"borough":1,"cuisine" :1}).pretty();

16.db.addresses.find({"name": /.*Reg.*/},{"restaurant_id" :
1,"name":1,"borough":1,"cuisine" :1}).pretty();
```

```
17.db.addresses.find({ "borough": "Bronx" , $or : [{ "cuisine" : "American " },{
"cuisine" : "Chinese" }] }).pretty();

18.db.addresses.find({"borough" :{$in :["Staten
Island","Queens","Bronx","Brooklyn"]}},{"restaurant_id" :
1,"name":1,"borough":1,"cuisine" :1}).pretty();

19.db.addresses.find({"borough" :{$nin :["Staten
Island","Queens","Bronx","Brooklyn"]}},{"restaurant_id" :
1,"name":1,"borough":1,"cuisine" :1}).pretty();

20.db.addresses.find({"grades.score" : { $not: {$gt : 10}}},{"restaurant_id" :
1,"name":1,"borough":1,"cuisine" :1}).pretty();

21.db.addresses.find({$or: [{name: /^Wil/}, {"$and": [{"cuisine" : {$ne
:"American "}},{"cuisine" : {$ne :"Chinees"}}]}]},{"restaurant_id" :
1,"name":1,"borough":1,"cuisine" :1}).pretty();

22.db.addresses.find( {"grades.date": ISODate("2014-08-11T00:00:00Z"),
"grades.grade":"A" ,"grades.score" : 11},{"restaurant_id" :
1,"name":1,"grades":1}).pretty();

23.db.addresses.find({ "grades.1.date":
ISODate("2014-08-11T00:00:00Z"),"grades.1.grade":"A" , "grades.1.score" :
9},{"restaurant_id" : 1,"name":1,"grades":1}).pretty();

24.db.addresses.find({"address.coord.1": {$gt : 42, $lte : 52}},{"restaurant_id"
: 1,"name":1,"address":1,"coord":1}).pretty();

25.db.addresses.find().sort({"name":1}).pretty();

26.db.addresses.find().sort({"name":-1}).pretty();

27.db.addresses.find().sort({"cuisine":1,"borough" : -1,}),pretty();

28.db.addresses.find({"address.street" :{ $exists : true }}).pretty();

29. db.addresses.find({"address.coord" :{$type : 1}}).pretty();

30.db.addresses.find({"grades.score" :{$mod : [7,0]}},{"restaurant_id" :
1,"name":1,"grades":1}).pretty();

31.db.addresses.find({ name :{ $regex : "mon.*", $options: "i"
}},{"name":1,"borough":1,"address.coord":1,"cuisine" :1}).pretty();

32.db.addresses.find({ name :{ $regex : /^Mad/i,
}},{"name":1,"borough":1,"address.coord":1,"cuisine" :1}).pretty();
```