1. Model the following Department system as a document database.

Consider a set of students, course and marks. A student can register for more than one course.
2. Assume appropriate attributes and collections as per the query requirements.
3. Insert at least 10 documents in each collection.
4. Answer the following Queries a. Count the number of students having more than 80 percentage
b. List the name and age of the oldest 5 students with marks less than 40
c. Use a cursor to display names of students whose percentage is greater than 70.
d. Find all female students which either live in Pune and Mumbai or got percentage less than 50

```
db.student.insert({name:"Abhi",course:[{coursename:"bcs"},{
        coursename:"bvoc"}],marks:80,age:21,gender:"male",city:"pune"})



        db.student.insert({name:"mukesh",course:[{coursename:"bcs
        "},{coursename:"bvoc"}],marks:60,age:22,gender:"male",city:
        "pune"})


        db.student.insert({name:"manisha",course:[{coursename:"m
        cs"},{coursename:"bvoc"}],marks:90,age:22,gender:"female",
        city:"mumbai"})


        db.student.insert({name:"manasi",course:[{coursename:"mcs
        Mukesh A. Patil Page 2
        "},{coursename:"bvoc"}],marks:92,age:22,gender:"female",ci
        ty:"latur"})


        db.student.insert({name:"apurva",course:[{coursename:"mcs
        "},{coursename:"bvoc"}],marks:37,age:22,gender:"female",ci
        ty:"sasvad"})


        db.student.insert({name:"arati",course:[{coursename:"mcs"},
        {coursename:"bvoc"}],marks:32,age:22,gender:"female",city:
        "bekarai"})
```

****

```
        4)=>
        a) > db.student.count({marks:{$gt:80}})
        b) > db.student.find({marks:{$lt:40}})
        c) > var my=db.student.find({marks:{$gt:70}});
```

```
> while(my.hasNext()){print(tojson(my.next()));}
d)>db.student.find({gender:"female",$or:[{city:"pune"},{city:"
mumbai"},{marks:{$lt:50}}]})
```

1. Model the following sales system as a document database.

Consider a set of products, customers, orders and invoices. An invoice is generated when an order is processed.
2. Assume appropriate attributes and collections as per the query requirements.
3. Insert at least 10 documents in each collection.
4. Answer the following Queries. a. List all products in the inventory.
b. List the details of orders with a value >10000.
c. List all the orders which has not been processed (invoice not generated).
d. List all the orders along with their invoice for “Mr. Arun Kumar”.

```
db.product.insert({name:"robot",price:12000})
 db.product.insert({name:"toycar",price:2000})
 db.product.insert({name:"cricketset",price:9000})
 db.product.insert({name:"studymaterial",price:19000})

db.custome.insert({cname:"mukesh",modelname:"samsungj6",amount:20000})


db.custome.insert({cname:"abhijeet",modelname:"samsungj6",amount:20060})

 db.custome.insert({cname:"manasi",modelname:"iphone7+",amount:30060})

db.custome.insert({cname:"manisha",modelname:"iphone7+",amount:30070})

 db.custome.insert({cname:"dipak",modelname:"iphone7+",amount:30800})


db.shopping.insert({brandname:"samsung",rate:6,model:[{mname:"s40",ram:"3GB",r
om:"32GB",rate:4},
{mname:"j6",ram:"4GB",rom:"32GB",rate:7},{mname:"j7",ram:"6GB",rom:"64GB",
rate:6}]})


db.shopping.insert({brandname:"vivo",rate:8,model:[{mname:"Y55",ram:"3GB",rom:
"32GB",rate:6},{mname:"Ys5",ram:"4
GB",rom:"32GB",rate:4},{mname:"YYY",ram:"6GB",rom:"64GB",rate:6}]})


**************************************************
a) >db.shopping.find({"model.ram":"3GB","model.rom":"32GB"})
```

b) > db.custome.find({modelname:"samsung j6"})
c) > db.shopping.aggregate([{"$sort":{"rate":-1}},{"$limit":1},{$group:{_id:"$brandname"}}])d) > db.custome.find().sort({ "cname": 1 })
kumar",product:{productName:"toycar",price:20000},order_
date:"12/2/2019",status:"processed",Totalbill:2039,invoice:{i
nvoiceNO:67564,bill:2039,date:"17/2/2019"}})

db.order.insert({orderno:3737,custName:"arun
kumar",product:{productName:"robot",price:12000},order_d
ate:"11/3/2019",status:"processed",Totalbill:12800,invoice:{i
nvoiceNO:67574,bill:12039,date:"17/3/2019"}})

> db.order.insert({orderno:3738,custName:"arun
kumar",product:{productName:"cricketset",price:9000},order
_date:"15/5/2019",status:"in process",Totalbill:9050})

db.order.insert({orderno:3739,custName:"mukesh
patil",product:{productName:"studentmaterial",price:19000}
,order_date:"15/8/2019",status:"in process",Totalbill:19080})

****
4)
a)> db.product.find().pretty()
b) > db.order.find({Totalbill:{$lt:10000}})
c) > db.order.find({status:"in process"})
d) >db.order.find({custName:"arunkumar",status:"processed"})

Model the following books system as a document database.

Consider a set of books and publishers A publisher can publish more than one book.
2. Assume appropriate attributes and collections as per the query requirements.
3. Insert at least 10 documents in each collection.
4. Answer the following Queries. a. List all Publishers which are located in Mumbai
b. List the details of books with a cost >1000.
c. List all the book which are written by “RaghuRamkrishnan” and published in 2017
List all the books published by “O Reilly” and are written either in English or Marathi

db.book.insert({BName:"shyamchiaai",cost:700,author:"sane guruji",published:2007})

db.book.insert({BName:"TwoSaints",cost:1700,author:"raguramkrishna",published:2017})

db.book.insert({BName:"ramkrushnaparamhans",cost:800,author:"raguramkrishna",publishe
d:2017})

db.book.insert({BName:"DMS",cost:300,author:"raguramkrishna",published:2005})

db.publisher.insert({pname:"OReilly",language:"English",books:[{BName:"ramkrushna
paramhans"},{BName:"Two Saints"}],city:"mumbai"})

db.publisher.insert({pname:"vision",language:"English",books:[{BName:"DMS"}],city:"pun
e"})

db.publisher.insert({pname:"OReilly",language:"marathi",books:[{BName:"shyamchiaai"}],c
ity:"mumbai"})

****

a) db.publisher.find({city:"mumbai"})
b) db.book.find({cost:{$lt:1000}})
c) db.book.find({author:"raguramkrishna",published:2017})
d) db.publisher.find({pname:"OReilly",$or:[{language:"English"},{language:"marathi"}]})

Model the following blog database with the following requirements: Every post has the unique title, description and url, Every post can have one or more tags, Every post has the name of its publisher and total number of likes, Every post has comments given by users along with their name, message, data-time and likes. On each post, there can be zero or more comments.
2. Assume appropriate attributes and collections as per the query requirements.
3. Insert at least 10 documents in each collection. [
4. Answer the following Queries a. List all the blogs which are tagged as food blogs
b. List all the blogs that are posted by “Amit”
c. List all the blogs that are tagged a “travel blogs” and were created before 2018 and are have comments written by “Sagar” and commented as “like”
d. List all the blogs that have comments which are posted before August 2019 or are not liked by the user posting the comment

db.post.insert({title:"online",url:"www.abc.com",tag:["food","travel"],pname:"mukesh",
pdate:new Date("2019-03-12"),like:89,user:[{name:"abhi",comment:"good",
message:"do best", cdate:new Date("2020-03-12"),like:1}]})

db.post.insert({title:"wetpet",url:"www.wetpet.com",tag:["food","travel"],
pname:"Amit",pdate:new Date("2018-03-
12"),like:82,user:[{name:"abhi",comment:"good",message:"dobest",time:"4pm",like:1},
{name:"mukesh",comment:"best",message:"success", cdate:new Dat("2008-11-
12"),like:2}]})

db.post.insert({title:"wetpet",url:"www.wetpet.com",tag:["food","travel","magic"],
pname:"abhijeet",pdate:newDate("2017-03-12"),like:182,
user:[{name:"sagar",comment:"like",message:"dobest",time:"4pm",like:1},
{name:"mukesh",comment:"best",message:"success", cdate:new Date("2019-03-12"),like:2}]})

db.post.insert({title:"nonveg",url:"www.non.com",tag:["food","travel","chiken"],pname:"Amit",pdate:new Date("2019-07-12"),
like:82,user:[{name:"manisha",comment:"good",message:"dobest",time:"4pm",like:0},
{name:"manasi",comment:"best",message:"success", cdate:new Date("2018-03-12"),like:0}]})

****

a) >db.post.find({tag:"food"})
b) >db.post.find({pname:"Amit"})
c) > db.post.find({tag:"travel",pdate:{"$lte":new Date("2018-03-11")},"user.name":"sagar","user.comment":"like"})
d) > db.post.find({$or:[{"user.cdate":{$lte:new Date("2019-08-07")}},{"user.like":0}]})

Model the following Tours information as a document database.

A tour will consider the source and destination. Destination may be all around the world. The tours are planned using different tourism industries. The industries provide the complete information before selecting a particular package. Customers select different packages according to their requirements and can rate/review the tourism industry.
2. Assume appropriate attributes and collections as per the query requirements.
3. Insert at least 10 documents in each collection.
4. Answer the following Queries. a. List the details of packages provided by “Veena World”
b. List the highest rated tourism industry.
c. List all the details of expenses made by John on his first 3 trips. Also display the total expenses.
d. List the names of the customers who went on a tour to Shillong.
db.tourism .insert({name:"veenaword",rate:9,package:[{pname:"shillong",cost:10000},{pname:"gujart",cost:7000},{pname:"karnataka",cost:6000}]})

db.tourism
.insert({name:"rohit",rate:7,package:[{pname:"shillong",cost:10000},{pname:"rujan",cost:7000}]})

db.tour.insert({sourc:"john",destination:"shillong",toerisumName:"veena word",tourisumrate:8000,expense:20000,year:2018,customer:[{cname:"mukesh",city:"pune"},

{cname:"abhijeetsangita",city:"baramati"},{cname:"manisha",city:"15no"},{cname:"manasi"
,city:"latur"}]})

db.tour.insert({sourc:"john",destination:"karnataka",toerisumName:"veena
word",tourisumrate:80090,expense:20900,year:2017,customer:[{cname:"mukesh",city:"pune
"},{cname:"abhijeetsangita",city:"baramati"},
{cname:"manisha",city:"15no"},{cname:"manasi",city:"latur"}]})

db.tour.insert({sourc:"john",destination:"rajasthan",toerisumName:"rohit",tourisumrate:6000,
expense:30400,year:2019,
customer:[{cname:"mukesh",city:"pune"},{cname:"abhijeetsangita",city:"baramati"},{cname
:"manisha",city:"15no"},{cname:"manasi",city:"latur"}]})

db.tour.insert({sourc:"john",destination:"taj",toerisumName:"rohit",tourisumrate:60090,expe
nse:10400,year:2016,
customer:[{cname:"mukesh",city:"pune"},{cname:"abhijeetsangita",city:"baramati"},{cname
:"manisha",city:"15no"},{cname:"manasi",city:"latur"}]})

****
a) >db.tourism .find({name:"veena word"}).pretty()
b) >db.tourism .find({}).sort({"rate":-1}).limit(1)
c)
>db.tour.aggregate([{"$sort":{"year":1}},{"$limit":3},{$group:{_id:null,"count":{"$sum":"$
expense"}}}])
d) > db.tour.find({destination:"shillong"})

Model the following scientist information as a document database.

The document keeps information about the scientist who has contributed in various fields like Artificial intelligence, Fortran etc. The scientist may have contributed in more than one filed. The scientist may have received more than one awards for his contribution in various fields.
2. Assume appropriate attributes and collections as per the query requirements.
3. Insert at least 10 documents in each collection.
4. Answer the following Queries. a. List names of all scientists whose last name starts with a N
b. List all scientist who were born after 1/1/1950 and are still alive
c. For each year list the identifiers of scientists thatreceived an award in that year
d. List all scientists who have received “Turing Machine Award” before 1980 and has made contributed in 4 fields
db.scien.insert({fname:"mukesh",lname:"navse",BOD:new
Date("1952-04-18"),DOD:"still
alive",field:["tcs","java","c","sql"],award:[{name:"turing
machine",year:1976},{name:"robotic",year:1998},{name:"cod
e talent",year:1995}]})

db.scien.insert({fname:"abhi",lname:"nalave",BOD:new

Date("1972-04-18"),DOD:"still
alive",field:["tcs","java","sql"],award:[{name:"code
master",year:1976},{name:"robot",year:1998},{name:"puzzle
talent",year:1995}]})

db.scien.insert({fname:"manisha",lname:"hipparkar",BOD:ne
w Date("1942-04-18"),DOD:new Date("2009-08-
06"),field:["tcs","java"],award:[{name:"topper",year:1976},{n
ame:"puraskar",year:1998},{name:"puzzle
talent",year:1995}]})

****

a) > db.scien.find({ lname: { $regex: /n/ } })

b) > db.scien.find({BOD:{"$gt":new Date("1950-03-11")},DOD:"still alive"})

c) db.scien.aggregate([{$group:{_id:{year:"$award.year",Name:"$award.name"}}}])

d) > db.scien.find({"award.name":"turingmachine","award.year":{$lt:1980},field:{$size:4}})

Model the following inventory information as a document database.

The inventory keeps track of various items. The items are tagged in various categories. Items may be kept in various warehouses and each warehouse keeps track of the quantity of the item.
2. Assume appropriate attributes and collections as per the query requirements
3. Insert at least 10 documents in each collection.
4. Answer the following Queries. a. List all items from the inventory where the status equals "D" and qty is greater than 30
b. List all items which have 3 tags
c. List all items having status equal to “A” or having quantity less than 30 and height of the product should be greater than 10
d. Find all warehouse that keeps item “Planner” and having instock quantity less than 20

db.item.insert({itemName:"planner",tag:["wash","food","vehicle"],status:"A",height:5,width:
9,
instack:15,warehouse:[{location:"pune",quntity:36},{location:"mumbai",quntity:67}]})

db.item.insert({itemName:"toycar",tag:["food","vehicle"],status:"D",height:5,width:9,instack
:15,
warehouse:[{location:"pune",quntity:36},{location:"mumbai",quntity:67}]})

db.item.insert({itemName:"roboticcar",tag:["food","vehicle"],status:"A",height:9,width:9,instack:5,
warehouse:[{location:"pune",quntity:26},{location:"mumbai",quntity:17}]})

db.item.insert({itemName:"bag",tag:["food","vehicle","school","travel"],status:"c",height:19,width:39,instack:75,
warehouse:[{location:"surat",quntity:26},{location:"lanavala",quntity:17}]})

****

a) > db.item.find({status:"D","warehouse.quntity":{$gt:30}})
b) > db.item.find({"tag":{$size:3}})
c) >db.item.find({$or:[{status:"A"},{"warehouse.quntity":{$lt:30},height:{$gt:10}}]})
d) > db.item.find({itemName:"planner",instack:{$lt:20}})

Model the following transaction information as a document database.

The transaction keep track of items purchased by a customer and the way in which the payment was done – Cash, Credit Card or Debit Card
2. Assume appropriate attributes and collections as per the query requirements.
3. Insert at least 10 documents in each collection.
4. Answer the following Queries. a. Find all transactions which were made by the user “John”
b. Find all the transactions which were made using debit card
c. Find transaction id and total amount of purchase made using a credit card
d. Find the total payment for each payment type

db.transaction.insert({itemName:"toy",customerName:"john",paymentmode:"debitcard",payment:8000})

db.transaction.insert({itemName:"car",customerName:"john",paymentmode:"creditcard",payment:4000})

db.transaction.insert({itemName:"bag",customerName:"mukesh",paymentmode:"cash",payment:5000})

db.transaction.insert({itemName:"airlineticket",customerName:"rohit",paymentmode:"cash",payment:50090})

db.transaction.insert({itemName:"mango",customerName:"abhijeet",paymentmode:"creditcard",payment:8000})

db.transaction.insert({itemName:"bus",customerName:"manasi",paymentmode:"debitcard",payment:7000})

****

a) > db.transaction.find({customerName:"john"})

b) > db.transaction.find({paymentmode:"debitcard"})

c)
>db.transaction.aggregate([{$match:{"paymentmode":"creditcard"}},{$group:{_id:null,"count":{"$sum":"$payment"}}}])

d)
>db.transaction.aggregate([{$group:{_id:"$paymentmode","count":{"$sum":"$payment"}}}])

Model the following Online Mobile Shopping information as a document database. Consider online mobile shopping where the customer can get different models from different brands. Customers can rate the brands and the models individually.
2. Assume appropriate attributes and collections as per the query requirements
3. Insert at least 10 documents in each collection.
4. Answer the following Queries. a. List the mobiles having RAM and ROM as 3GB and 32GB.
b. List the customers who bought Samsung J6.
c. List the names of the distinct brands available. Also display the name of the brand with highest rating.
d. List all the customers in ascending order who bought iPhone 7plus.

db.custome.insert({cname:"mukesh",modelname:"samsungj6",amount:20000})

db.custome.insert({cname:"abhijeet",modelname:"samsungj6",amount:20060})

db.custome.insert({cname:"manasi",modelname:"iphone7+",amount:30060})

db.custome.insert({cname:"manisha",modelname:"iphone7+",amount:30070})

db.custome.insert({cname:"dipak",modelname:"iphone7+",amount:30800})

db.shopping.insert({brandname:"samsung",rate:6,model:[{mname:"s40",ram:"3GB",rom:"32GB",rate:4},
{mname:"j6",ram:"4GB",rom:"32GB",rate:7},{mname:"j7",ram:"6GB",rom:"64GB",rate:6}]})

db.shopping.insert({brandname:"vivo",rate:8,model:[{mname:"Y55",ram:"3GB",rom:"32GB",rate:6},{mname:"Ys5",ram:"4
GB",rom:"32GB",rate:4},{mname:"YYY",ram:"6GB",rom:"64GB",rate:6}]})

****

a) >db.shopping.find({"model.ram":"3GB","model.rom":"32GB"})

b) > db.custome.find({modelname:"samsung j6"})

c) > db.shopping.aggregate([{"$sort":{"rate":-1}},{"$limit":1},{$group:{_id:"$brandname"}}])d) > db.custome.find().sort({ "cname": 1 })