Exercise 12 – Nested Documents and Aggregate Pipelines

In the exercises we will be using the more advanced queries like searching nested documents, searching arrays, and writing aggregate pipelines.

Download the file mongobookstore.zip from D2L. Create a new database called “Exercise11” and create collections for the data from the zip file and upload the data.

The data inside these JSON files should be the same as what was created in the previous labs. But if you have modified any of the data from the previous lab, this will put everything back to the same starting values.

# Part A – With Instructor

## Question 1

Write a single query that lists just the titles of the books that were written by the author ‘Elmore Leonard’.

db.book.find({"author.name":"Elmore Leonard"}, {"title":1, "\_id":0})

## Question 2

Write a single query that lists the title and prices of the books from the publisher “Jove Books” that fall in the category of “Graphic Design”.

db.book.find({$and: [{"publisher.name":"Jove Books"}, {"category":"Graphic Design"}]}, {"title":1, "price":1, "\_id":0})

## Question 3

Find out how many shopping carts contain the book “West of Dodge”. You can use two queries for this:

1. Use a query to find the ID number of the book.
2. Use a query to count how many times the ID number shows up (do not worry about the quantity in the shopping cart). You will want to use the elemMatch operator as shown in the slides.

db.book.find({"title":"West of Dodge"}, {"\_id":1})

db.shopping\_cart.find({'items':{$elemMatch:{'item':ObjectId('64a986022e39eb32f0217111')}}}).count()

## Question 4

Consider the following SQL query:

select category, count(\*) from book

group by category

having count(\*) >= 70 and count(\*) <= 72;

This query would list which categories of books have between 70 and 72 books in the collection.

Write a MongoDB aggregate pipeline that produces the same result.

db.book.aggregate([

{

$group: {\_id:"category", count:{$sum:1}}

},

{

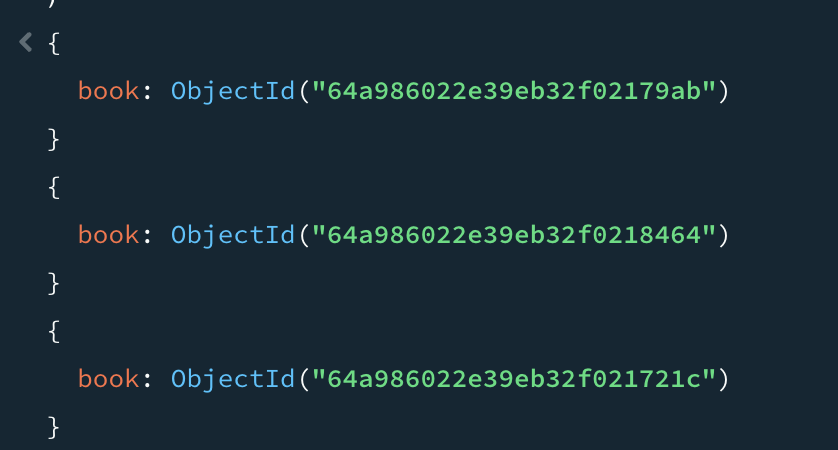
$match: { $and : [ {count:{$gte:70}}, {count:{$lte:72}} ] }

}

])

## Question 5

Write a query that will list the IDs of all the books that the customer ‘Dani Mose’ has put into their shopping carts. You may write two queries, the first to find the customer ID of Dani Mose then an aggregate pipeline to list the book IDs. The output should look like this:



db.shopping\_cart.aggregate([

{

$match:{"customer":ObjectId('64a982c13188fc5c9613cda1')}

},

{

$unwind:"$items"

},

{

$group: {\_id:"$items.item"}

},

{

$project : {\_id:0, book:"$\_id"}

}

])