Lucas Hillert Jiménez
1ºASIR “A”
Gestión de Bases de Datos

Proyecto Final Primer Trimestre GBD

ÍNDICE

-Introducción

En este proyecto vamos a realizar una base de datos sobre cámaras de fotografía creando una colección llamada “camaras” donde introduciremos 15 documentos sobre cámaras con diferentes características.Seguidamente realizaremos consultas para buscar información específica sobre las cámaras.

También mostraremos la estructura de nuestra base de datos, explicaremos las consultas con los operadores que hemos visto en clase y además alguna aportación personal con cosas que no se han enseñado en clase.

-Estructura de la base de datos

```
db.camaras.insertMany(
    [
    {   _id:1,
        modelo:{
            nombre:"Alpha 7 III",
            marca:"Sony",
            megapixeles:37.8,
            estabilizador:true
        },
        objetivo:"Gran angular",
        conexion:{
            bluetooth:true,
            wifi:true
        },
        memoriascompatibles:["MS Duo","MS Pro-HG Duo","SD","SDHC","SDXC"],
        fpsgrabacion:[120,60,30],
        dimensiones:{
            altura:80.7,
            ancho:145.9,
            peso:"885g"},
        cyp:[
            {color:"negro",precio:3000},
            {color:"blanco",precio:3050},
            {color:"gris",precio:3100}
        ],
        fecha:{
            fabricacion:new Date("2021-09-10"),
            lanzamiento:new Date("2021-11-13")
        },
        garantia: 2
```

Esta es la estructura de mi base de datos sobre cámaras de fotografía en la cual he introducido los siguientes campos:

-Campo _id que es numérico que identifica a cada documento de las diferentes cámaras.
-Campo modelo que es un tipo documento que incluye a su vez dos campos cadena referidos al nombre y marca,uno numérico de los megapíxeles y uno booleano(de true o false) sobre si viene con estabilizador o no.
-Campo objetivo que es un tipo cadena que se refiere al objetivo que viene con la cámara.
-Campo conexión que también es un tipo documento que a su vez contiene dos campos booleanos uno sobre si tiene wifi y otro sobre el bluetooth.
-Campo memorias compatibles que es un tipo array de cadenas que contiene todas las memorias compatibles.
-Campo fps de grabación que es un campo tipo array numérico que muestra que capacidades de grabación tiene.
-Campo dimensiones que es un tipo documento que a su vez contiene dos campos decimales referidos a la altura y anchura y un campo cadena que contiene el peso en gramos.
-Campo odc(opciones de compra) que es un tipo array de documentos en el cual cada documento a su vez contiene un campo cadena sobre el color y otro numérico sobre el precio.
-Campo fecha que es un tipo documento que contiene dos campos tipo fecha con las respectivas fechas de fabricación y lanzamiento.
-Campo garantía que es un tipo numérico que contiene información sobre cuántos años de garantía tiene

-Explicación de consultas con los operadores vistos en clase

Operadores:

$eq nos muestra los documentos que tienen un valor igual al especificado.
ej :

```
db.camaras.find( {"modelo.estabilizador": {$eq:true}} )
```

Esta consulta nos muestra todas las cámaras con el valor true en el campo modelo.estabilizador.
$gt nos muestra los documentos que tienen un valor mayor al especificado.
ej:

```
db.camaras.find( {fpsgrabacion: {$gt:30} })
```

Esta consulta nos muestra todas las cámaras que tienen un valor mayor a 30 en el campo fpsgrabación.
&gte nos muestra los documentos que tienen un valor mayor o igual al especificado.
ej:

```
db.camaras.find({"modelo.megapixeles": {$gte:28}})
```

Esta consulta nos muestra todas las cámaras que tienen un valor mayor o igual a 28 en el campo modelo.megapixeles.
$lt nos muestra los documentos que tienen un valor menor al especificado.
ej:

```
db.camaras.find({"dimensiones.ancho":{$lt: 140}})
```

Esta consulta nos muestra todas las cámaras que tienen un valor menor a 140 en el campo dimensiones.ancho.
$lte nos muestra los documentos que tienen un valor menor o igual al especificado.
ej:

```
db.camaras.find({"odc.precio": {$lte:1300}})
```

Esta consulta nos muestra todas las cámaras que tienen un valor menor o igual a 1300 en el campo odc.precio.
$ne nos muestra los documentos que no tienen un valor igual al especificado(efecto contrario al operador **$eq**).
ej:

```
db.camaras.find( {"conexion.bluetooth":{$ne: true}})
```

Esta consulta nos muestra todas las cámaras que no tienen el valor true en el campo conexión.bluetooth.
$in nos muestra los documentos que tengan cualquiera de los valores especificados.
ej:

```
db.camaras.find({objetivo: {$in: ["Normal","Teleobjetivo"]}})
```

Esta consulta nos muestra todas las cámaras que tengan el valor Normal o Teleobjetivo en el campo objetivo.
&or nos muestra los documentos que cumplan cualquiera de las condiciones que especifiquemos.
ej:

```
db.camaras.find({$or: [{objetivo: "Ojo de pez"},{"odc.color":"azul"}] })
```

Esta consulta nos muestra todas las cámaras que cumplen las condiciones de el campo objetivo con el valor Ojo de pez o del campo odc.color con el valor azul.
$nin nos muestra los documentos que no tengan ninguno de los valores especificados.
ej:

```
db.camaras.find( {"modelo.marca": {$nin: ["Sony","Canon"]}} )
```

Esta consulta nos muestra todas las cámaras que no tienen el valor Sony ni el valor Canon en el campo modelo.marca.
$nor nos muestra los documentos que no cumplan ninguna de las condiciones que especifiquemos.

```
db.camaras.find({$nor:[{objetivo:"Gran angular"},{garantia:2}] })
```

Esta consulta nos muestra todas las cámaras que no cumplen ningúna de las condiciones del campo objetivo con el valor Gran angular o del campo garantía con el valor 2.
$exists nos muestra los documentos en los que existe un campo especificado
ej:

```
db.camaras.find( {garantia: {$exists:true}} )
```

En esta consulta nos muestra todas las cámaras en las que el campo garantía existe.

$regex nos muestra los documentos que tienen el valor de una expresión regular especificada.
ej:

```
db.camaras.find( {"modelo.nombre": {$regex:/^e/i}} )
```

Esta consulta nos muestra todas las cámaras que en el campo modelo.nombre tienen un valor que coincida con la expresión regular que hemos puesto.
&all nos muestra los valores especificados para un tipo array.
ej:

```
db.camaras.find({memoriascompatibles: {$all: ["SDHC","SD"] } })
```

Esta consulta nos muestra todas las cámaras que en el campo memoriascompatibles tipo array tiene los valores “SDHC”,”SD”.
$elemMatch nos muestra los documentos que al menos en un valor de un campo tipo array cumple las condiciones especificadas.
ej:

```
db.camaras.find({"odc": {$elemMatch: {color:{$eq:"blanco"},precio: {$lt:1000} } } })
```

$size nos muestra los documentos que en un campo tipo array tiene un tamaño con un valor especificado
ej:

```
db.camaras.find( {fpsgrabacion: {$size: 2}})
```

Esta consulta nos muestra todas las cámaras que en el campo fpsgrabación tipo array tiene un tamaño de dos valores.
$and nos muestra los documentos que cumplen todas las condiciones especificadas.
ej:

```
db.camaras.find( {$and: [ {"odc.precio":
{$lte:1300}},{"modelo.estabilizador": {$eq:true}},{"conexion.wifi":
{$eq:true}},{"conexion.bluetooth": {$eq:true}} ] } )
```

Esta consulta nos muestra todas las cámaras que cumplen las siguientes condiciones:
-que el valor del campo odc.precio sea menor o igual a 1300
-que el valor del campo modelo.estabilizador sea igual a true
-que el valor del campo conexion.wifi sea igual a true
-que el valor del campo conexion.bluetooth sea igual a true
Método **.count()**

```
db.camaras.find( {garantia: {$exists:true} }).count()
```

Esto es un ejemplo de una consulta que muestra el número de cámaras en las que se cumples que el campo garantía existe.Cuando queramos saber el número de cámaras que hay en una colección con unas condiciones ya establecidas usamos el método **.count()** después de usar el método **.find()**

1)Consultas usando varios operadores

```
db.camaras.find( {$and: [ {"odc.precio":
{$lte:1300}},{"modelo.estabilizador": {$eq:true}},{"conexion.wifi":
{$eq:true}},{"conexion.bluetooth":
{$eq:true}},{"fecha.fabricacion":{$gt: new Date("2021-01-01")}} ] } )
```

En esta consulta el objetivo es encontrar las cámaras con un precio menor o igual a 1300 euros, que venga con estabilizador, conexión wifi y bluetooth, y además que su fecha de fabricación sea a partir de este año.
Primeramente usaremos el operador **$and** para que se cumplan todos las condiciones que especificamos en la consulta. A continuación para buscar las que tienen un precio menor igual ponemos el operador **$lte** y seguidamente usamos el operador **$eq**
para que nos busque específicamente las cámaras que tienen el campo true en estabilizador conexión wifi y bluetooth. Por último ponemos el operador **$gt** para que nos busque una fecha a partir de este año.

2)Consultas usando varios operadores

```
db.camaras.find( {$and: [ {"modelo.marca": {$nin:
["Sony","Canon"]}},{"odc.color":{$eq:"azul"}},{fpsgrabacion: {$gt:30} }
] } )
```

En esta consulta nos va a buscar las cámaras que no sean de la marca Sony ni la Canon,que tengan la opción de compra de color azul y además graben a más de 30 fps.
Primero usamos el operador **$and** para que cumpla todos los requisitos que buscamos .En segundo lugar usamos el operador **$nin** para que no nos busque las cámaras de la marca Sony ni Canon, seguidamente usamos el operador **$eq** para especificar que queremos el campo color con el valor azul, y por último usamos el operador **$gt** para buscar todas las cámaras que tengan más de 30 fps de grabación.

3)Consultas usando varios operadores

```
db.camaras.find( {$and: [ {"dimensiones.peso":
{$gt:"500g"}},{"modelo.megapixeles": {$gte:28}},{objetivo: {$in:
["Normal","Teleobjetivo"]}},{garantia: {$exists:true}} ] } )
```

Esta consulta nos busca las cámaras que tienen peso mayor a 500 gramos, 28 megapíxeles o más, objetivo Normal o Teleobjetivo y además tienen garantía.
Al igual que con todos las consultas complejas iniciamos con un operador **$and** para que nos cumpla todas las condiciones que ponemos después con los distintos operadores.
A continuación con el operador **$gt** buscamos las cámaras que en el campo peso dentro del campo dimensiones tienen un valor mayor a 500 gramos,seguidamente con el operador **$gte** buscamos que tengan en el campo megapíxeles un valor igual o mayor a 28, luego con el operador **$in** buscamos las que tengan objetivo Normal o Teleobjetivo y por último el operador **$exists** para que exista el campo garantía.

4)Consultas usando varios operadores

```
db.camaras.find( {$and: [ {"odc": {$elemMatch:
{color:{$eq:"blanco"},precio: {$lt:1000} } } },{$or: [{fpsgrabacion:
{gte:60}},{odc:{$size: 3}}] },{"fecha.lanzamiento":{$gt: new
Date("2021-05-01")}} ] } )
```

En esta consulta estamos buscando las cámaras que tengan al menos una opción de compra con el color blanco y precio menor a 1000 euros, que tenga 60 fps o más de grabación, y que tenga al menos 3 opciones de compra con su respectivo precio y color, además de que su fecha de lanzamiento sea a partir de el (2021-05-01).
Como en todas las consultas complejas usamos el **$and** para empezar y que nos aplique todas las condiciones puestas a la búsqueda. Seguidamente hemos usado el operador **$elemMatch** para indicar que un valor de un array corresponda a las condiciones especificadas color y precio, en segundo lugar hemos usado un operador **$or** para que nos busque o que tenga fps de grabación mayor o igual a 60 mediante **$gte** o que el campo odc que es un array tenga 3 valores de opciones de compra y por último que tenga una fecha de lanzamiento mayor a la especificada con el operador **$gt**.

5)Consultas usando varios operadores

```
db.camaras.find( {$and: [ {memoriascompatibles: {$all: ["SDHC","SD"] }
},{$nor:[{objetivo:{$eq:"Gran
angular"}},{"dimensiones.altura":{$lt:80}}] },{"modelo.nombre":
{$regex:/^eos/i}} ] } )
```

En esta consulta estamos buscando las cámaras que tengan las memorias compatibles SDHC y SD,que no tengan objetivo gran angular ni altura menor a 80 mm además de que el nombre del modelo empiece por la expresión eos sin tener en cuenta mayúsculas y minúsculas.
Al igual que en las demás comenzamos con el operador $and para que nos aplique todas las condiciones establecidas para la búsqueda. En primer lugar usamos el operador &all para buscar específicamente dentro del array memoriascompatibles los valores “SDHC” y “SD”,en segundo lugar usamos el operador $nor para que no nos cumpla ninguna de las condiciones puestas a continuación que son, que el campo objetivo sea igual al valor gran angular por medio del operador $eq ni que el campo dimesiones.altura sea menor que 80 por medio del operador $lt, y por último que por medio del operador $regex nos busque que el valor del campo modelo.nombre comience por las letras eos sin respetar mayúsculas o minúsculas.

-Explicación consultas nuevas

Método .sort() y .limit()

En este nuevo método especificamos el orden en el que la consulta devuelve documentos especificando un campo en concreto en este ejemplo vamos a usar el campo fecha de fabricación.
Para que nos ordene los documentos por el campo fechas tenemos que ejecutar dentro del método **sort()** el campo y darle un valor 1 para ordenarlo de manera ascendente(de menor a mayor) o -1 para que se ordenarlo de manera descendente(de mayor a menor),en este caso como es un tipo fechas pues nos ordenará numéricamente.

```
db.camaras.find().sort({"fecha.fabricacion": -1}).limit(1)
```

Después ponemos el **.limit()** para limitar la lista a lo que queramos si queremos que solo nos muestre el primer documento de la lista pondremos 1 y si queremos ver todos los documentos pues dejamos el espacio vacío de la siguiente manera:

```
db.camaras.find().sort({"fecha.fabricacion": -1}).limit()
```

De esta forma la consulta nos mostrará todos los documentos listados por el campo fecha.fabricación de mayor a menor.
También podemos ordenar un campo que sea de letras o letras y números usando el mismo método pero en este ejemplo vamos a ordenarlo de manera ascendente con el valor 1 por tanto nos mostrará en orden alfabético y numérico de menor a mayor la siguiente lista filtrada por el siguiente campo:

```
db.camaras.find().sort({"modelo.nombre": 1}).limit(4)
```

Como podemos ver hemos modificado el método .limit() a 4 para que nos muestre solo los 4 primeros de la lista ordenada por el campo modelo.nombre de forma alfabética y numérica ascendente.

Método .aggregate() y operadores $set $unset

En este nuevo método nos permite agregar a la colección estos dos operadores **$set** para agregar un campo con valores y luego el operador **$unset** para poder quitar un campo completo con todos sus valores.
Observamos cómo funciona en el siguiente ejemplo:

```
db.camaras.aggregate( [{ $set: { "odc.unidades": 3 }}] )
```

Esta consulta nos devolverá todos los documentos con un nuevo campo llamado “unidades” con valor 3 dentro del campo “odc” usando el operador **$set**.

```
db.camaras.aggregate( [{$unset:"odc.unidades"}] )
```

Esta otra consulta nos devolverá todos los documentos con el campo anteriormente creado “unidades” eliminado de todos los campos “odc” usando el operador **$unset**.