Base de datos con MongoDB

Actores

Iván Gasent Zarza 1 DAM

Índice de contenido

Estructura de la base de datos

Las entradas de la base de datos tienen la siguiente estructura:

```
{
nombre: "Carrie Fisher",
nacimiento: new Date("1956-10-21"),
muerte: new Date("2016-12-27"),
pais: "Estados Unidos",
familia: {
        estado_civil: "divorciada",
        hijos: 1
        },
peliculas: [
        {
                titulo: "Star Wars: Episodio IV - Una nueva esperanza",
                estreno: new Date(1977 - 05 - 25),
                recaudacion: 775398007.57
        },
        .
        .
        .
        {
                titulo: "Rogue One: Una historia de Star Wars",
                estreno: new Date(2019 - 12 - 19),
                recaudacion: 1074144248.85
        },
    ],
premiado: true
}
```

Nombre es un alfanumérico que indica el nombre del actor.
Nacimiento es una fecha que indica la fecha de nacimiento.
Muerte es una fecha que indica la fecha de muerte del actor, si el actor está vivo no existirá este campo.
Pais es un alfanumérico que indica el país donde nació el actor.
Familia es un documento con los campos estado_civil e hijos, que indican el estado civil como alfanumérico y el número de hijos que tiene como un número entero. Si no tiene hijos, el valor será 0.
Peliculas indica las películas en las que ha trabajado, siendo este un array de documentos con los campos titulo, estreno y recaudacion. Estos campos indican el titulo de la película como

alfanumérico, la fecha de estreno como fecha y el dinero en dólares que ha recaudado como un número decimal.
Premiado es un booleano indica si el actor ha recibido algún premio.

Operadores usados para resolver enunciados

Para resolver los enunciados que aparecen en el archivo enunciadosActores.js, he usado los siguientes operadores y comandos como parámetros para los comandos find():

db.collection.distinct(): muestra todos los valores de un campo dado.

$all: selecciona los documentos donde se cumplan todos los criterios que tenga en un array.

$elemMatch: selecciona los documentos donde al menos un valor de un campo array coincida con todos los criterios especificados.

$exists: selecciona los documentos donde el campo elegido existe si se da el valor true o no exista si se da el valor false.

$and: selecciona los documentos que cumplan todos los criterios especificados.

$or: selecciona los documentos que cumplan alguno de los criterios especificados.

$in: selecciona los documentos donde se coincida con alguno de los valores de un array.

$nor: selecciona los documentos que no cumplan ninguno de los criterios especificados.

$not: selecciona los documentos que no cumplan el criterio especificado.

$nin: selecciona los documentos donde no se coincida con ninguno de los valores de un array.

$gt: selecciona los documentos donde el valor sea mayor que el especificado.

$gte: selecciona los documentos donde el valor sea mayor o igual que el especificado.

$lt: selecciona los documentos donde el valor sea menor que el especificado.

$lte: selecciona los documentos donde el valor sea menor o igual que el especificado.

$regex: selecciona los valores donde la cadena incluya la expresión regular.

Aporte personal

Como aporte personal, he usado las operaciones de actualización de entradas para poder cambiar la información de los documentos existentes. El comando usado tiene la siguiente estructura:

```
db.collection.updateMany(
    <filtrado>,
    <actualización>,
    {
        upsert: <boolean>,
        writeConcern: <document>,
        collation: <document>,
        arrayFilters: [ <filterdocument1>, ... ],
```

```
        hint: <document|string>
    }
)
```

Los valores dentro de los corchetes son parámetros opcionales y solo se usrá arrayFilters en esta base datos. arrayFilters permite seleccionar valores de un array si cumplen ciertas condiciones.

El parámetro <filtrado> funciona igual que las condiciones para el comando find pero <actualización> necesita el uso de operadores de actualización. Los operadores de actualización usados han sido:

$set: Establece el campo seleccionado al valor dado. Si no existe ese campo en el documento se crea.

$push: Añade un elemento a un array.

$inc: Aumenta un número en el valor dado.

$[<identificador>]: Funciona como un marcador de posición para los elementos que coincidan con arrayFilter.

$mul: Multiplica el valor por un número dado.

Referencias

https://docs.mongodb.com/manual/reference/method/db.collection.distinct/

https://docs.mongodb.com/manual/reference/operator/query/all/

https://docs.mongodb.com/manual/reference/operator/query/elemMatch/#mongodb-query-op.-elemMatch

https://docs.mongodb.com/manual/reference/operator/query/exists/

https://docs.mongodb.com/manual/reference/operator/query/nor/

https://docs.mongodb.com/manual/reference/operator/query/not/

https://docs.mongodb.com/manual/reference/operator/query-comparison/#std-label-query-selectors-comparison

https://docs.mongodb.com/manual/reference/operator/query/regex/

https://docs.mongodb.com/manual/tutorial/update-documents/

https://docs.mongodb.com/manual/reference/operator/update/#std-label-update-operators