Ejercicio práctico de MongoDB

Para resolver este ejercicio se utilizará la consola o ***mongo shell*** de MongoDB.

Debe conectarse al servidor, y crear una base de datos llamada `mongo_ejercicio`

Documentar todas sus consultas/instrucciones en un archivo de Word para usar como referencia.

R/ use mongo_ejercicio

Documentos a Insertar

Insertar los siguientes documentos en una colección llamada `películas`

R/ `db.createCollection('peliculas')`

```
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

R/ `db.peliculas.insertMany([{title: 'Fight Club', writer: 'Chuck Palahniuk', year: 1999, actors: ['Brad Pitt', 'Edwar Norton']},{title: 'Pulp Fiction', writer: 'Quentin Tarantino', year: 1994, actors: ['John Travolta', 'Uma Thurman']},{title: 'Inglorius Basterds', writer: 'Quentin Tarantino', year: 2009, actors: ['Brad Pitt', 'Diane Kruger', 'Eli Roth']}, {title: 'The Hobbit: An Unexpected Journey', writer: 'J. R. R. Tolkien', year: 2012, franchise: 'The Hobbit'}, {title: 'The Hobbit: The Desolation of Smaug', writer: 'J. R. R. Tolkien', year: 2013, franchise: 'The Hobbit'}, {title: 'The Hobbit: The Battle of the Five Armies', writer: 'J. R. R. Tolkien', 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 Hernan's Big Adventure"}, {title: 'Avatar'}])`

Consultas / Buscar documentos

Realizar las siguientes consultas en la colección `peliculas`:

1. Obtener todos los documentos
   R/ db.peliculas.find()

2. Obtener documentos con `writer` igual a "Quentin Tarantino"
   R/ db.peliculas.find({writer: 'Quentin Tarantino'})

3. Obtener documentos con `actors` que incluyan a "Brad Pitt"
   R/ db.peliculas.find({actors: 'Brad Pitt'})

4. Obtener documentos con `franchise` igual a "The Hobbit"
   R/ db.peliculas.find({franchise: 'The Hobbit'})

5. Obtener todas las películas de los 90s.
   R/ db.peliculas.find({year: { $gte: 1990, $lt: 2000 } })

6. Obtener las películas estrenadas entre el año 2000 y 2010.
   R/ db.peliculas.find({year: { $gte: 2000, $lte: 2010 } })

7. Obtener todos los documentos, mostrar sólo el título de la película, ordenar por título ascendentemente y limitar el resultado a sólo 4 resultados, saltando los 2 primeros.
   R/ db.peliculas.find({} ,{ title: 1, _id: 0 }).sort({title:1}).limit(4).skip(2)

Actualizar Documentos

1. Agregar ***sinopsis*** a "The Hobbit: An Unexpected Journey" : "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."
   R/ db.peliculas.updateOne({title: 'The Hobbit: An Unexpected Journey'}, {$set: {sinopsis: 'A reclutant 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.'}})

2. Agregar ***sinopsis*** a "The Hobbit: The Desolation of Smaug" : "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."
   R/ db.peliculas.updateOne({title: 'The Hobbit: The Desolation of Smaug'}, {$set: {sinopsis: '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.'}})

3. Agregar una actor llamado "Samuel L. Jackson" a la película "Pulp Fiction"
   R/ db.peliculas.updateOne({title: 'Pulp Fiction'}, {$set: {'actors.2': 'Samuel L. Jackson'}})

Búsqueda por Texto

1. Encontrar las películas que en la sinopsis contengan la palabra "Bilbo" (mostrar sólo el título y año)
   R/ db.peliculas.find({ sinopsis: { $regex: 'Bilbo' } },
   {title:1,year:1,_id:0})

2. Encontrar las películas que en la sinopsis contengan la palabra "Gandalf"
   R/ db.peliculas.find({ sinopsis: { $regex: 'Gandalf' } })

3. Encontrar las películas que en la sinopsis contengan la palabra "Bilbo" y no la palabra "Gandalf"
   R/ db.peliculas.find({$and: [{sinopsis: {$regex: 'Bilbo'}}, {sinopsis: {$not: /Gandalf/}}]})

4. Encontrar las películas que en la sinopsis contengan la palabra "dwarves" ó "hobbit"
   R/ db.peliculas.find({$or: [{sinopsis: {$regex: 'dwarves'}}, {sinopsis: {$regex: 'hobbit'}}]})

5. Encontrar las películas que en la sinopsis contengan la palabra "gold" y "dragon" (ordenar por año de forma descendiente)
   R/ db.peliculas.find({$and: [{sinopsis: {$regex: 'gold'}}, {sinopsis: {$regex: 'dragon'}}]}).sort({year:-1})

Eliminar Documentos

1. Eliminar la película "Pee Wee Herman's Big Adventure"
   R/ db.peliculas.deleteOne({title: "Pee Wee Hernan's Big Adventure"})

2. Eliminar la película "Avatar"
   R/ db.peliculas.deleteOne({title: "Avatar"})

Eliminar Colecciones

1. Escribe la sintaxis para eliminar una colección “productos”
   R/ db.productos.drop()

Cuestionario

1. Señala cuál de las siguientes consultas en MongoDB devuelve el nombre, DNI y _id de los documentos de la colección “usuarios” con edad mayor o igual a 18:

a) `db.usuarios.find({edad: {gte: 18}},{dni:1, nombre:1, _id:0})`
b) `db.usuarios.find({edad: {gt: 18}},{dni:1, nombre:1})`
c) `db.usuarios.find({edad: {gte: 18}},{dni:1, nombre:1})`
d) `db.usuarios.find({edad: {gte: 18}},{dni:0, nombre:0, _id:0})`

2. La siguiente operación sobre una colección en MongoDB devuelve:

```
db.socios.find({nombre: 'Alberto'},{nombre: 1, apellidos: 1,
dni: 1}).sort({nombre: -1})
```

a) El nombre, apellidos y DNI de los socios con nombre ‘Alberto’ en orden ascendente según el nombre.
b) El nombre, apellidos y DNI de los socios con nombre ‘Alberto’ en orden descendente según el nombre.
c) El nombre, apellidos, DNI y _id de los socios con nombre ‘Alberto’ en orden ascendente según el nombre.
d) El nombre, apellidos, DNI y _id de los socios con nombre ‘Alberto’ en orden descendente según el nombre.

3. En los documentos de una colección en MongoDB, el usuario no puede dar valor al campo _id:

a) Verdadero
b) Falso

4. Explique qué es una base de datos y cuál es la diferencia entre SQL y NoSQL. A su criterio ¿cuál es mejor?

R/ Una base de datos es un lugar o herramienta donde se almacenan y se guardan datos para posteriormente ser consultados. La principal diferencia entre SQL y NoSQL es que la primera utiliza una estructura como tablas, con filas y columnas, y lleva un orden específico, mientras que NoSQL no las utiliza. A mi criterio es mucho más práctico NoSQL por la facilidad que aporta al momento de diligenciar una información y el criterio de búsqueda.

Una base de datos es un sistema organizado que permite almacenar, gestionar y consultar grandes volúmenes de información de manera eficiente.

Las bases de datos SQL (relacionales) utilizan un esquema estructurado basado en tablas con filas y columnas, y son ideales cuando se requiere integridad y relaciones entre los datos. En cambio, las bases de datos NoSQL (no relacionales) permiten estructuras más flexibles, como documentos JSON, grafos o pares clave-valor, lo que las hace más adecuadas para datos no estructurados o en constante cambio.

A mi criterio, NoSQL puede ser más práctico en proyectos donde la estructura de los datos varía frecuentemente, se necesita escalabilidad horizontal o se manejan grandes volúmenes de datos sin relaciones complejas. Sin embargo, la mejor opción depende del tipo de proyecto.