

TALLER-2 INTRODUCCION A MONGODB

DANIEL ESTEBAN RODRIGUEZ VELASCO

PORTAFOLIO:

https://drive.google.com/drive/folders/1rT2zLh7Z-EWqcbi2Pw-dMozb9m2plEcs?usp=share_link

GITHUB:

https://github.com/darocode/introduction_MongoDb

INTRODUCCION A BASE DE DATOS NO RELACIONALES

1. Consulte y realice un cuadro comparativo sobre los tipos de dato que se pueden usar dentro de Mongo DB, indicando el nombre, su descripción o uso y un ejemplo de representación de este.

MongoDb -Tipo de datos		
Nombre	Descripción	Ejemplo
String	Permite almacenar cadenas de caracteres y el valor del field va entre corchetes	"Name": "Daniel"
Date	Se usa para valores tipo fechas	"Fecha": date("2022-01-01") "Fecha": ISODate("2022-01-01T21:49:096.01")
Int/numéricos	Permite almacenar valores numéricos tipo int y tipo double según sea el caso se debe especificar con numberlog	"Numero": 123
Array	Almacena otros field con tipos de datos diferentes según sea necesario	"datos": [{"name": "", "lastname": ""}]
Boolean	Datos de tipo booleanos que almacenan true o false	"Publicado": Boolean { "publicado": true }
ObjectId	Es el tipo de dato que almacena la primary key de un documento	"_id": 1

2. Consulte el nombre, la definición y uso de cada uno de los operadores propuestos para trabajo con consultas en Mongo DB

OPERADOR	NOMBRE	DEFINICION
\$eq	Equal(igual que)	Una consulta de un field debe coincidir con un valor dado
\$gt	Greather than(mayor que)	Consulta un field y si su valor es mayor al de la consulta devuelve los documentos que coincidan
\$gte	Greather than or equal(mayor o igual que)	Consulta un field y si su valor es mayor o igual al de la consulta devuelve los documentos que coincidan
\$lt	Lower than(menor que)	Consulta un field y si su valor es menor al de la consulta devuelve los documentos que coincidan
\$lte	Lower than or equal(menor o igual que)	Consulta un field y si su valor es menor o igual al de la consulta devuelve los documentos que coincidan
\$ne	Not equal(distinto de)	Según los valores de los field devuelve todos los documentos que no tengan el valor dado en la consulta
\$in	In (dentro de)	Según el valor de los field devuelve unos los documentos entre rangos
\$nin	Nin (no dentro de)	Según el valor de los field devuelve unos los documentos que no esté entre rangos de la consulta
\$or	Operación or(encontrar uno u otro)	Encuentra los documentos que coincidan con uno u otro valor del field
\$and	Operación and(encontrar valores)	Deben encontrar los valores de la consulta y si no coinciden devuelve error
\$not	Operación Not(no traer valores igual que)	No trae los documentos que tengan un valor de un field según la consulta

\$nor	Operacion or	Trae los documentos, dejando fuera los que tengan valores de los field asignados
\$exists	Exist(valor encontrado)	Es una consulta boolean true y puede usar otros operadores.
\$type	Type(date)	Con este operador consulta el valor de los field de los documentos
\$mod	Mod(encontrar datos de un documento)	Este devuelve los documentos con los valores dados de las posiciones del documento
\$regex	Regex(consulta)	Es una búsqueda de patrones dentro de un documento
\$elemMatch	elementMatch(consults)	Selecciona los field usando varias consultas y devuelve la colección que coincida
\$size	Size(consulta)	Devuelve los documentos con el tamaño de fielda especificado

EXPLICACION

1. Crear la base de datos

Usaremos la instrucción “ use ” y el nombre de nuestra base de datos para crearla y al mismo tiempo usarla

```
admin> use videojuegos
switched to db videojuegos
videojuegos>
```

2. Crear las colecciones

Posteriormente crearemos las colecciones de nuestra base de datos. Entonces usaremos el comando **db.createCollection(“NOMBRE DE LA COLECCION”)** y luego con el comando **show collections** verificamos que si se hayan creado

```
videojuegos> db.createCollection('FPS')
{ ok: 1 }
videojuegos> db.createCollection('deportes')
{ ok: 1 }
videojuegos> db.createCollection('Terror')
{ ok: 1 }
videojuegos> show collections
deportes
FPS
Terror
videojuegos>
```

3. Insertar documentos con sus fields y sus valores

Se puede insertar documentos de diferentes formas, se puede usar **insertOne** para agregar un solo registro o **insertMany** este último nos permitirá insertar varios documentos a la vez.

Se usan paréntesis y después dentro de estos va la estructura de los documentos y sus fields y se separan con “ , ”.

```

videojuegos> db.FPS.insertMany(
... [
... {
...   _id:1,
...   "nombre":"Call of Duty",
...   "fecLanzamiento":"29/10/2003",
...   "numSecuelas":45,
...   "online":"si"
... },{
...   _id:2,
...   "nombre":"Battlefield",
...   "fecLanzamiento":"20/09/2002",
...   "numSecuelas":21,
...   "online":"si"
... },{
...   _id:3,
...   "nombre":"Valorant",
...   "fecLanzamiento": "02/06/2020",
...   "numSecuelas":1,
...   "online":"si"
... },{
...   _id:4,
...   "nombre":"Far cry",
...   "fecLanzamiento":"23/03/2004",
...   "numSecuelas":16,
...   "online":"no"
... },{
...   _id:5,
...   "nombre":"Halo",
...   "fecLanzamiento":"15/11/2001",
...   "numSecuelas":15,
...   "online":"si"
... },{
...   _id:6,

```

4. Consultas

Usaremos los diferentes métodos e instrucciones para consultar, estos se encuentran en la documentacion de MONGODB.

- Utilizamos \$gt(mayor que) para traer los documentos que sus fields tengan un valor mayor al dado

```
videojuegos> db.FPS.find({ numSecuelas: {$gt:20}})
[
  {
    _id: 1,
    nombre: 'Call of Duty',
    feclanzamiento: '29/10/2003',
    numSecuelas: 45,
    online: 'si'
  },
  {
    _id: 2,
    nombre: 'Battlefield',
    feclanzamiento: '20/09/2002',
    numSecuelas: 21,
    online: 'si'
  }
]
```

- Pero también podemos utilizar las diferentes instrucciones como:
 - \$eq(igual que)
 - \$ne(distinto a)
 - \$lt(menor que)
 - \$lte(menor o igual que)
 - \$nin(valores no dentro del rango)
 - \$in(dentro de)

CONSULTAS

gt->mayor que.

```
db.FPS.find({ numSecuelas: {$gt:20}})
```

eq-> Igual que.

```
db.FPS.find({ online:{$eq:"no"}})
```

ne-> Distinto de.

```
db.FPS.find({ online:{$ne:"no"}})
```

lt-> Menor que.

```
db.Terror.find({ numSecuelas:{$lt:10}})
```

lte-> Menor o igual que.

```
db.Terror.find({ numSecuelas:{$lte: 2}})
```

nin ->No este entre.

```
db.deportes.find({ numSecuelas:{$nin:[2,10]}})
```

in-> Dentro de.

```
db.deportes.find({ numSecuelas:{$in:[3,10,20]}})
```

or-> Un valor u otro

```
db.FPS.find({ $or:[{ nombre:{$eq:"Call of Duty"}},{ online:{$eq:"si"}}]})
```

5. Actualizaciones

También pueden variar, por ejemplo para poder hacer varias al tiempo utilizamos **updateMany** donde dentro de este método primero ponemos el valor que queremos actualizar y después separado por una coma ira el valor al que queremos cambiar

```

videojuegos> db.FPS.updateMany({ online:{$eq:"no"}},{ $set:{ online:"si"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 2,
  modifiedCount: 2,
  upsertedCount: 0
}
videojuegos> db.FPS.updateMany({ numSecuelas:{$eq:1}},{ $set:{ numSecuelas:3}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 3,
  modifiedCount: 3,
  upsertedCount: 0
}
videojuegos>

```

También podemos actualizar una sola actualización con **updateOne**, tiene la misma estructura que el anterior

```

videojuegos> db.deportes.updateOne({ nombre:"Pro Evolution Soccer"},{ $set:{ nombre:"PES"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
videojuegos>

```

La tercera forma es en caso de no encontrar el documento que quieres actualizar hará un insert y al final agregará un field upsert con su valor en true

```

videojuegos> db.FPS.updateOne({_id:{$eq:11}},{ $set:{ nombre:"Spec Ops"}},{ $set:{ "fecLanzamiento": "23/04/2014"}},{ $set:{ "numSecuelas":3}},{ $set:{ "online":"no"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 0,
  modifiedCount: 0,
  upsertedCount: 0
}
videojuegos>

```

6. Eliminar

Si queremos eliminar algún documento, tenemos tres formas.

- Para eliminar uno a la vez

```
videojuegos> db.FPS.findOneAndDelete({"nombre":"Valorant"})
{
  _id: 3,
  nombre: 'Valorant',
  fechaLanzamiento: '02/06/2020',
  numSecuelas: 3,
  online: 'si'
}
videojuegos>
```

- Para eliminar varios

```
videojuegos> db.Terror.deleteMany({"online":{"$eq":"no"}})
{ acknowledged: true, deletedCount: 4 }
videojuegos>
```

- Para encontrar y eliminar

```
videojuegos> db.deportes.findOneAndDelete({"fechaLanzamiento":"23/03/2002"})
{
  _id: 4,
  nombre: 'WWE 2k',
  fechaLanzamiento: '23/03/2002',
  numSecuelas: 22,
  online: 'si'
}
videojuegos>
```