Proyecto 1ª evaluación

MONGODB

Antonio José Méndez Ramírez
IES PUNTA DEL VERDE | 1°DAM

Contenido

1.) Introducción

En este proyecto he utilizado una base de datos de una empresa que se dedica a vender exclusivamente portátiles.

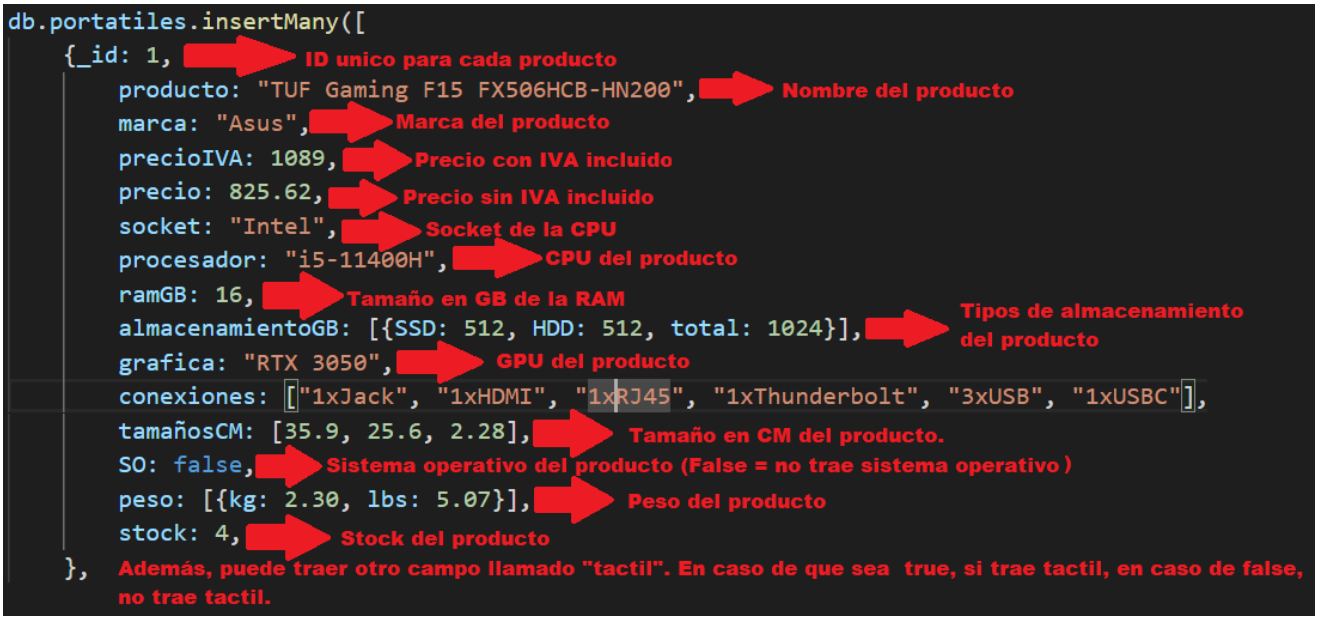
En las inserciones he usado varios tipos de datos, tales como: Enteros, cadenas alfanuméricas, booleanos, fechas, arrays de documentos, arrrays de enteros, arrays de cadenas alfanuméricas, arrays dentro de arrays, etc…

En las propias consultas a la base de datos he usado varios operadores como $and, $or, $nor, $not, $in, etc…

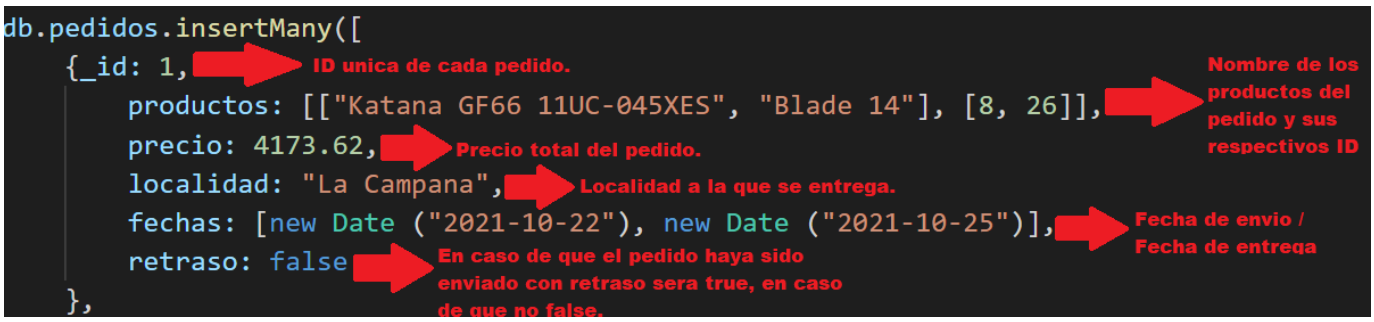
2.)Inserciones

Hay 2 colecciones en este proyecto:

- La colección portátiles:



- La colección pedidos:



3.) Operadores de comparación usados.

3.1) $eq:

Compara documentos donde el valor es igual al indicado:

```
{<campo>: { $eq: <valor> } }
```

3.2) $gt:

Compara aquellos documentos donde el valor es mayor al indicado:

```
{<campo>: { $gt: <valor> } }
```

3.3) $gte:

Compara aquellos documentos donde el valor es mayor o igual al indicado:

```
{<campo>: { $gte: <valor> } }
```

3.4) $lt:

Compara aquellos documentos donde el valor es menor al indicado:

```
{<campo>: { $lt: <valor> } }
```

3.5) $lte:

Compara aquellos documentos donde el valor es menor o igual al indicado:

```
{<campo>: { $lte: <valor> } }
```

3.6) $in:

Selecciona los documentos donde el valor de un campo especificado es igual a los valores especificados dentro de una Array:

```
{ <campo>: { $in: [<valor1>, <valor2>, ... <valorN> ] } }
```

3.7) $ne:

Selecciona los documentos que son la negación del valor indicado dentro del campo especificado.

```
{<campo>: { $ne: <valor> } }
```

3.8) $nin:

Selecciona los documentos donde el valor de un campo especificado es la negación a los valores especificados dentro de una Array:

```
{ <campo>: { $nin: [<valor1>, <valor2>, ... <valorN> ] } }
```

4.) Operadores lógicos usados.

4.1) $and:

Realiza una lógica donde ambos valores indicados tienen que ser cumplidos para que la propia condición AND se cumpla, y selecciona todos los documentos que cumplen todas las expresiones.

```
{ $and: [ { <expresion1> }, { <expresion2> }, …,{ <expresionN> } ] }
```

4.2) $or:

Realiza una lógica donde uno de los dos valores indicados tienen que ser cumplidos para que la propia condición OR se cumpla, y selecciona todos los documentos que cumplen todas las expresiones.

```
{ $or: [ { <expresion1> }, { <expresion2> }, …,{ <expresionN> } ] }
```

4.3) $not:

Realiza una lógica donde uno de los dos valores indicados NO tienen que ser cumplidos para que la propia condición NOT se cumpla, y selecciona todos los documentos que cumplen todas las expresiones.

```
{ $not: [ { <expresion1> }, { <expresion2> }, …,{ <expresionN> } ] }
```

4.4) $nor:

Realiza una lógica donde uno de los dos valores indicados tienen que ser cumplidos para que la propia condición NOR se cumpla, y selecciona todos los documentos que cumplen todas las expresiones.

```
{ $nor: [ { <expresion1> }, { <expresion2> }, …,{ <expresionN> } ] }
```

5.) Operadores de elemento.

5.1) $exist:

Busca un campo dentro de un documento: en caso de que sea true y el campo exista devolverá el campo indicado, en caso de que sea false no hace nada.

```
{<campo>: { $exist: <valor booleano> } }
```

6.) Operadores de evaluación.

6.1) $regex:

Proporciona opciones de expresiones regulares para campos alfanuméricos para patrones coincidentes.

```
{<campo>: { $regex: /<valor>/<opción> } }
```

7.) Operadores de Arrays.

7.1) $all:

Este operador selecciona los documentos donde el valor de un campo es un array que contiene todos los elementos especificados.

```
{ <campo>: { $all: [ <value1> , <value2> ... ] }
}
```

7.2) $elemMatch:

Este operador compara documentos que contienen un campo de array con al menos un elemento que coincide con todos los criterios de consulta especificados.

```
{ <campo>: { $elemMatch: [ <comparador1> ,
<comparador2> ... ] } }
```

7.3) $size:

Este operador devuelve documentos que coinciden con un campo de arrays con el tamaño que le pasemos.

```
{ <campo>: { $size: <valor numérico> } }
```

8.) Aportación personal.

8.1) .? en regex:

```
db.portatiles.find({
    producto: {$regex:/.?G733QR-K4008T/i}
```

8.2) Aggregate:

Procesan varios documentos y devuelven resultados procesados. Se pueden utilizar para:

- Agrupar valores de varios documentos diferentes.
- Realizar operaciones para devolver un resultado único.
- Analizar cambios a la larga.

8.3) $group:

Agrupa los documentos de entrada por la expresión indicada mediante la "_id" y para cada agrupación distinta, genera un documento.

```
db.portatiles.aggregate([
    {$group:
        {_id: "$marca",
        total:
            {$avg: "$precio"}
        }
    }
])
```

8.4) $sum:

Calcula y devuelve la suma de valores numéricos.

```
{ $sum: <expression> }
```

8.5) $multiply:

Calcula y devuelve la multiplicación de valores numéricos.

```
{ $multiply: <expression> }
```

8.6) $subtract:

Calcula y devuelve la resta de valores numéricos.

```
{ $subtract: [ <expression1>, <expression2> ] }
```

8.7) $avg:

Calcula y devuelve el promedio de valores numéricos.

```
{ $avg: [ <expression1>, <expression2> ] }
```

8.8) $project:

Pasa los documentos con los campos solicitados a la siguiente

etapa del proceso. Loscampos especificados pueden ser campos existentes de los documentos de entrada o campos recién calculados.

```
{ $project: { <especificaciones> } }
```

8.9) $size:

Hace coincidir cualquier array con el número de elementos especificados por el argumento.

```
{ <campo>: { $size: <valor numérico> } }
```

8.10) $match:

Hace coincidir cualquier array con el número de elementos especificados por el argumento.

```
{ $match: { <query> } }
```

8.10) $max:

Devuelve el valor máximo.

```
{ $max: <expression> }
```

8.11) $min:

Devuelve el valor minimo.

```
{ $min: <expression> }
```

8.12) update:

Sirve para añadir campos y valores a un documento ya existente.

8.13) $set:

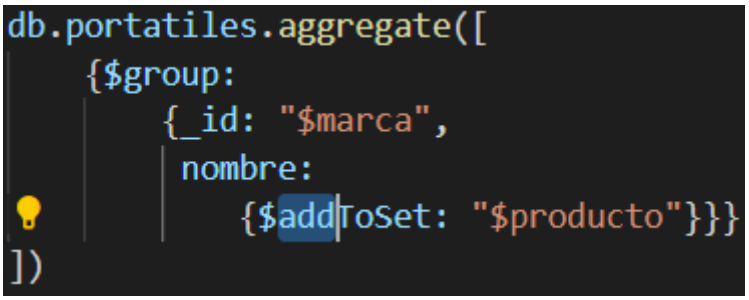
Se combina con el update para añadir campos sin sobreescribir nada a un documento ya existente.

```
db.portatiles.updateOne({_id: 22}, {$set: {"tactil": true}})
```

8.14) $addToSet:

Se puede usar tanto en el agreggate para agrupar campos como en el update con la misma función que el $set.

```
db.portatiles.updateOne(
    {_id: 22},
    {
        $addToSet: {tactil: true}
    }
)
```


db.portatiles.aggregate([
{$group:
{_id: "$marca",
nombre:
{$addToSet: "$producto"}}}
])