DIGITALHOUSE

Revisión semana 7

Certified Tech Developer

The Ultimate Degree

Índice

01 Compass

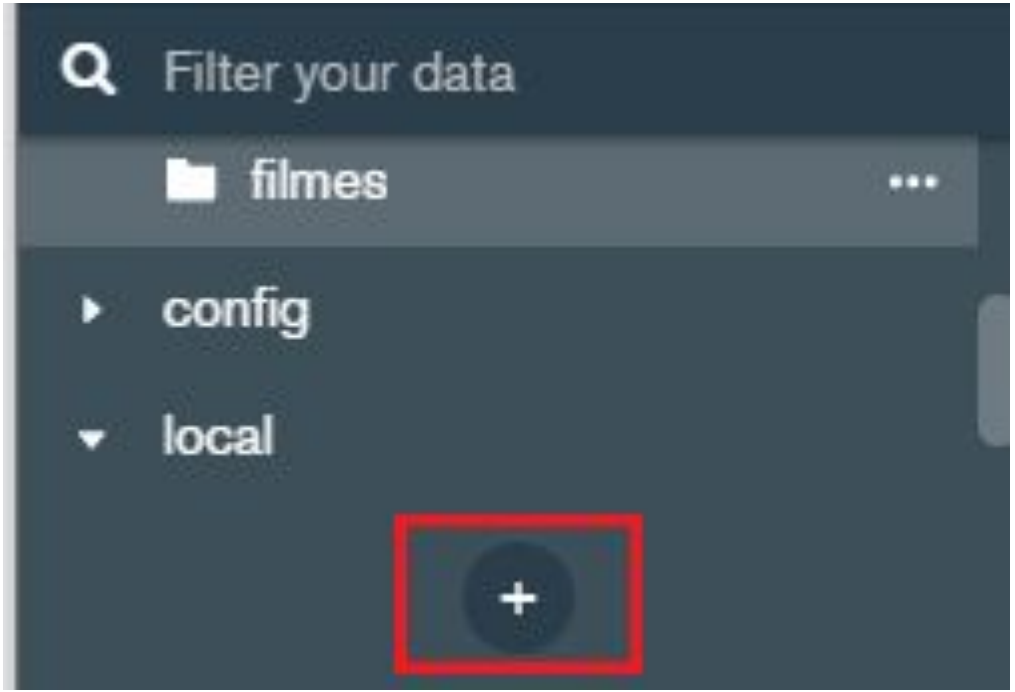
Comandos CRUD

- **db.collection.updateMany()** - Actualiza varios documentos.
- **db.collection.deleteOne()** - Excluye un único documento.
- d**b.collection.deleteMany** - Excluye varios documentos.
- **db.collection. find()** - Recupera los documentos de una colección.
- **db.collection.insertOne()** - Para insertar un único documento.
- **db.collection.insertMany()** - Para insertar varios documentos.
- **db.collection.find()** - Semejante a Select. Recupera documentos de una colección.
- **db.collection.updateOne()** - Actualiza un único documento.

Compass

Compass es la GUI para MongoDB. Compass, intuitivo y flexible, está diseñado para explorar y manipular fácilmente una base de datos. Proporciona vistas detalladas del esquema, métricas de rendimiento en tiempo real, capacidades de consulta sofisticadas y más.

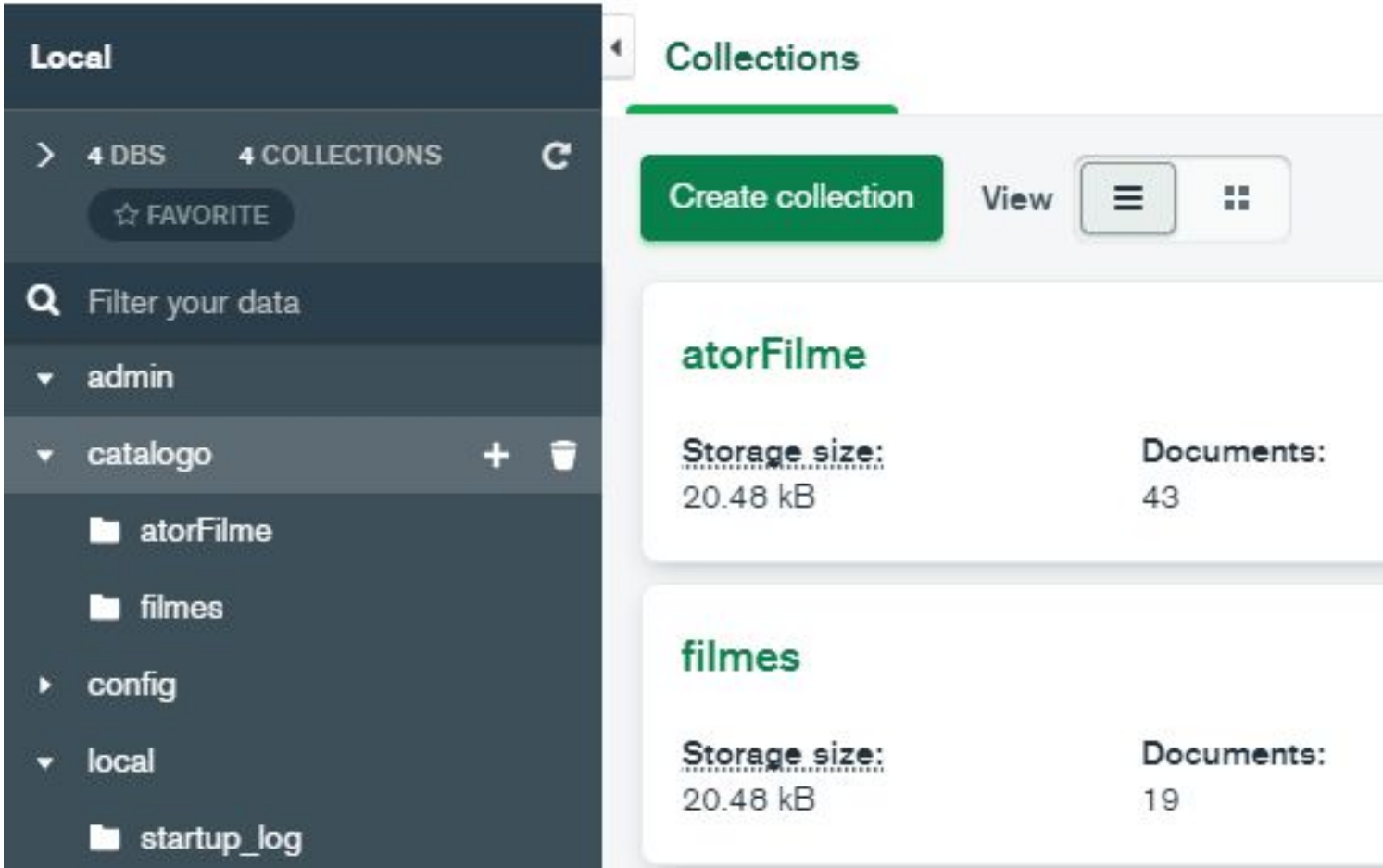
Hagan clic en el signo + para crear una nueva base de datos.



Compass

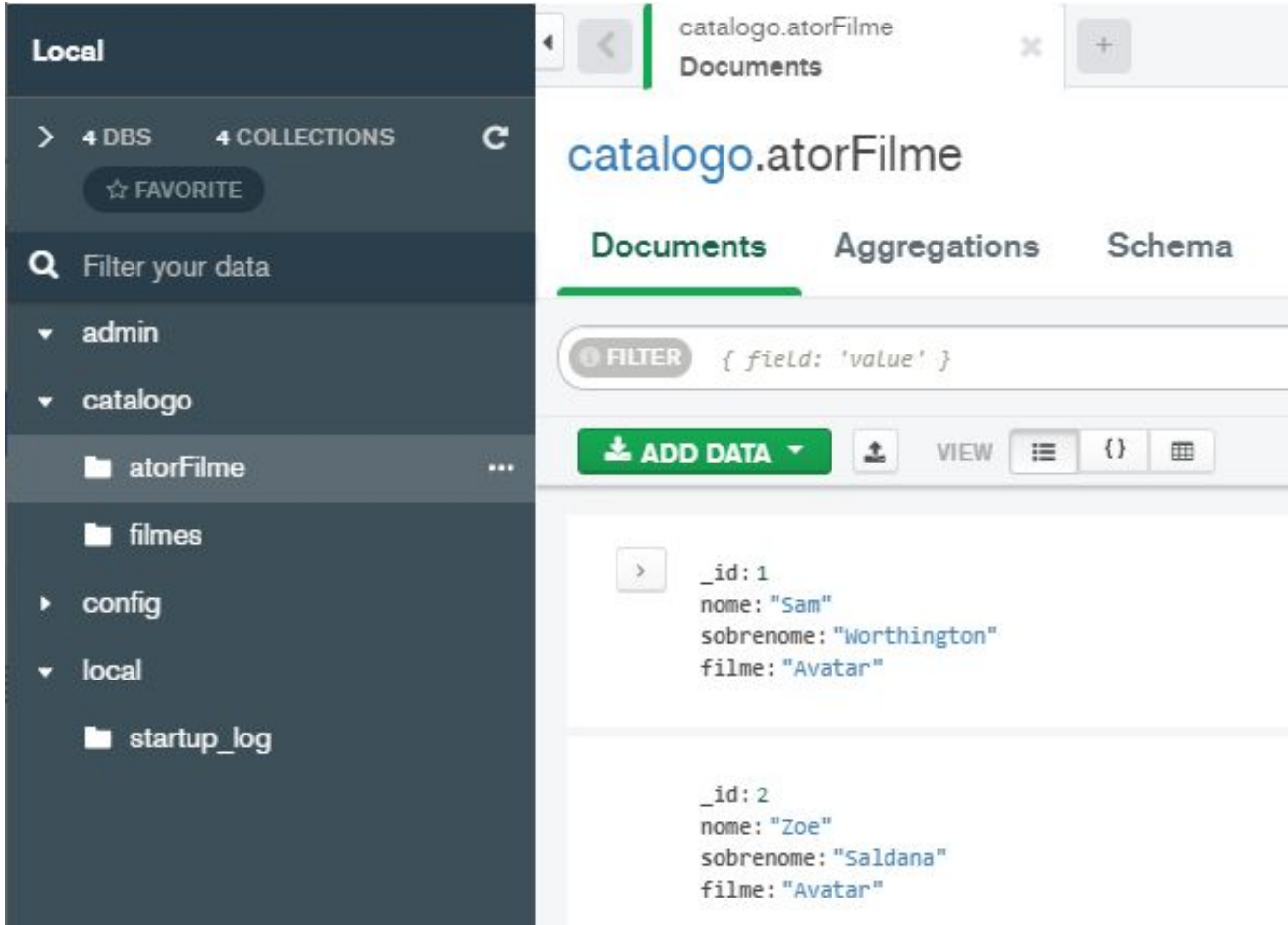
Al crear la base de datos, también se puede crear la colección.

También se puede seleccionar una base de datos y hacer clic en el botón Create collection para crear una colección.



Seleccionen una colección para ver o insertar documentos.

Para insertar documentos, haciendo clic en el botón ADD DATA, podemos elegir entre importar un archivo csv o Json o Insertar documento en modo Json o editor campo por campo.



DBII.contato

DOCUMENTS 6 | STORAGE SIZE 20.5KB | AVG. SIZE 121B | INDEXES 1 | TOTAL SIZE 36.9KB | AVG. SIZE 36.9KB

Documents | Aggregations | Schema | Explain Plan | Indexes | Validation

COLLATION | Untitled | SAVE | SAMPLE MODE | AUTO PREVIEW

6 Documents in the Collection | Preview of Documents in the Collection

La guía Aggregations permite crear pipelines, o sea, etapas para nuestra consulta.

Estos pipelines pueden ser exportados a JAVA, C#, Node.js o Python, para insertarlos en nuestro proyecto, reduciendo el tiempo de desarrollo.

```
_id: 1
nome: "Ana Luisa"
idade: 22
cidade: "Campo Grande"
estado: "MS"
telefone: "(67) 3032-2233"
```

Select... | A sample of the aggregated results from this stage will be shown below

DBII.contato

DOCUMENTS 6 | STORAGE SIZE 20.5KB | AVG. SIZE 121B | INDEXES 1 | TOTAL SIZE 36.9KB | AVG. SIZE 36.9KB

Documents | Aggregations | Schema | Explain Plan | Indexes | Validation

COLLATION | Untitled | SAVE | SAMPLE MODE | AUTO PREVIEW

6 Documents in the Collection

Preview of Documents in the Collection

También podemos guardar el pipeline creado para su uso posterior o crear una view, que podemos consultar siempre que sea necesario.

Select an operator to construct expressions used in the aggregation pipeline stages. Learn more

```
_id: 1
nome: "Ana Luisa"
idade: 22
cidade: "Campo Grande"
estado: "MS"
telefone: "(67) 3032-2233"
```

Select...

A sample of the aggregated results from this stage will be shown below

Guías de Compass

- **Schema** - Proporciona una descripción general del tipo de datos y la forma de los campos en una colección específica.
- **Validation** - Su función es definir reglas de validación para su schema.
- **Explain Plain** (**Planes de Ejecución**) - Muestra el comportamiento de una consulta. Estos datos son importantes para que identifiquemos problemas de performance o uso de índices.
- **Indexes** - Muestra los índices que se crearon en una colección y cuáles se usaron. Esta es información importante ya que puede sugerir cambios en su query o índice.

02 Queries Complejas

Operadores y funciones de MongoDB para crear consultas complejas

$exists - Comprueba si existe o no un determinado campo en la colección.
Sintaxis:
db.collection.find({ campo: { $exists: true, $nin: ["parametro1", "parametro2"] } })

Like - Utilizamos / / - similar al comodín % de MySQL.
Sintaxis: db.collection.find({campo: /^a/})

$group - Similar a Group By en MySQL, agrupa los datos correspondientes a un parámetro.
Sintaxis : db.collection.aggregate([{$group: {_id: "$campo",Total: {$sum: 1}}}])

$in - Similar a MySQL Lista de documentos que coinciden con algún parámetro definido en $in.
Sintaxis: db.collection.find({ campo: {$in: ["parametro1", "parametro2"] } })

Operadores y funciones de MongoDB para crear consultas complejas

$nin - Similar a not in de MySQL
Enumera los documentos que no coinciden con ningún parámetro definido en el $nin.

Sintaxis: db.collection.find({ campo: {$nin: ["parametro1", "parametro2"] } })

$lookup - Función equivalente a left join.

$push - Agrega una matriz del atributo seleccionado.

Sintaxis:
db.collection.aggregate([{$group:{ _id:"$campo", alias: {$sum:1}, alias1: {$push: "$campo"} } }])
MongoDB ofrece muchos más operadores y funciones para facilitar la manipulación de datos.

$unwind - Transforma el array en objeto, facilitando la recuperación de los datos.

03 Índices

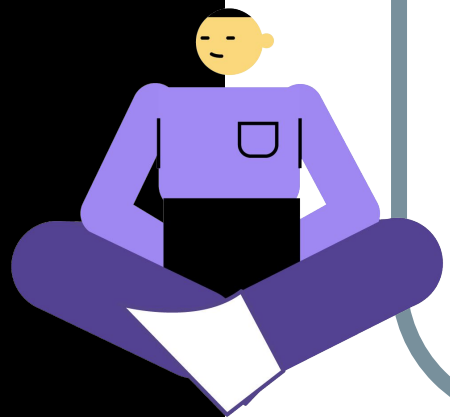
Los **índices** en **MongoDB** funcionan de manera similar a los índices en bases de datos relacionales.

Se generan en forma de **B-Tree**, es decir, los datos se almacenan en forma de árbol, pero manteniendo los nodos equilibrados.

Esto aumenta la velocidad en la búsqueda y también en la devolución de resultados ya ordenados.

Para mejorar la eficiencia de los índices, se recomienda que tengan una alta **cardinalidad**.

La cardinalidad es el número de elementos que no se repiten. Cuantos más valores únicos tenga el campo, mayor será la cardinalidad. Y más eficiente será el índice.



Tipos de índices

- **Índices simples o de campo único** - Estos índices se aplican a un solo campo en una colección.

  Sintaxis:
  db.collection.ensureIndex({ "_id": 1})

- **Índices compuestos** - En este caso el índice se generará en varios campos y podremos utilizarlos para consultar uno o más campos, sin tener que incluirlos todos.

  Sintaxis:
  db.collection.ensureIndex({ campo1: 1, campo2:-1 })eles.

Tipos de índices

- **Índices únicos -** Se crean para valores únicos. Agregamos el parámetro único al crearlos

  Sintaxis:
  db.collection.ensureIndex({ _id : 1 }, {"unique":true})

- **Índices dispersos** - Incluye todos los documentos. No es necesario que el campo exista en todos los documentos.

  Sintaxis: db.collection.ensureIndex({ "campo" : 1 }, {"sparse":true})

Tipos de índices

- **Índices de Texto** - Los índices de texto deben ser un string o una matriz de elementos de string.

  Sintaxis:
  db.collection.createIndex({campo: "text"})

- **Índice comodín** - Usando el especificador comodín ($**), puede crear múltiples campos de índice de texto. MongoDB indexa todos y cada uno de los campos que contienen datos de string en todos los documentos presentes en la colección dada.

  Sintaxis:
  db.collection.createIndex({ "$**": "text" } **)**

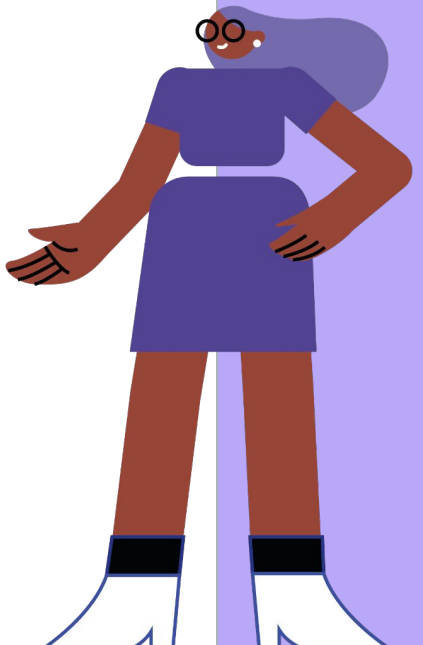
Conclusiones

¿Ven cuánto aprendimos esta semana?

Ahora tenemos mucho que estudiar y practicar.

No olviden anotar todas las dudas y discutir con compañeros y profesores, al fin y al cabo, el intercambio de información y experiencias también es conocimiento.

¡Buenos estudios!



DIGITALHOUSE

¡Muchas gracias!