Reporte de Laboratorio Nro. 4

Danny Bedoya[L00379402], Ronald Villavicencio[L00385751], Cesar Zambrano[L00383003]

Universidad de las Fuerzas Armadas
djbedoya1@espe.edu.ec
rsvillavicencio@espe.edu.ec
cgzambrano4@espe.edu.ec

Tema: CRUD en MongoDB

Resumen

Se realizarán las diferencies operaciones básicas CRUD haciendo uso dl Shell de MongoDB. Se creará un documento de una colección de ejemplo con el fin de aprender los comandos básicos de MongoDB, y obtener conocimientos acerca de las bases de datos no relaciones. Se creará, leerá, actualizará y eliminará documentos de una colección de base de datos de MongoDB y también se eliminará la base de datos creadas.

1. Introducción

En el presente informe de laboratorio se plasmará la actividad propuesta en clases, para ello realizaremos las operaciones básicas de CRUD mediante la herramienta de MongoDB. Se realizará la creación, lectura, actualización y eliminación de documentos de una colección.

Como bien sabemos MongoDB es un sistema de base de datos NoSQL, este sistema está orientado a documentos de código abierto. Almacena los datos en estructura de datos BSON el cual es similar a JSON a diferencia que maneja un esquema dinámico. Como se mencionó MongoDB está orientado a documentos de código abierto, por lo cual se encuentra disponible en los sistemas operativos Windows, GNU/Linux, OS X y Solaris.

El principal objetivo de este laboratorio es aprender a realizar las principales operaciones básicas de CRUD desde el Shell de MongoDB. Con el fin de realizar una buena practica y obtener el conocimiento básico de base de datos no relacionales y adentrarse en la utilización de comandos básicos de MongoDB.

2. Método

2.1. Paso 1: Creación de base de datos

Para iniciar con esta practica CRUD en MongoDB crearemos una base de datos, para ello se usa el comando "use" como en la Figura 1. Para comprobar que se creo de manera correcta la base de datos nos indicara el mensaje "switched to dbNewPersons", lo cual nos indica que se creo la base de datos con el nombre establecido. Se ejecutará el comando "show dbs" para enlistar las bases de datos que tenemos creadas.

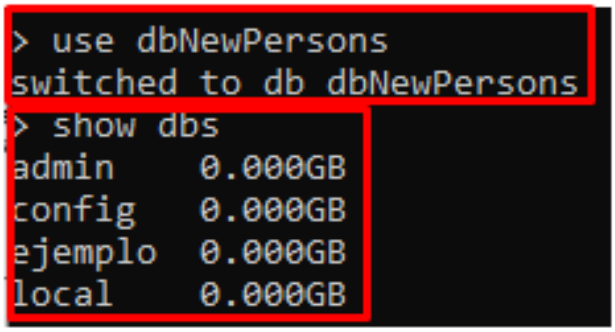

Figura 1: Creación de base de datos.

2.2. Paso 2: Agregar documentos a una colección

Como podemos observar al ejecutar el comando "show dbs" no se encuentra enlistada la base de datos que acabamos de crear, esto se debe a que no hay datos agregados a la misma. Se creará una colección de nombres y se ingresará un documento a esa misma colección para ello se usará el comando "insert". Ahora ya podemos volver a ejecutar el comando "show dbs" y ahora veremos que la base de datos "dbNewPersons" ya se encuentra enlisatada.

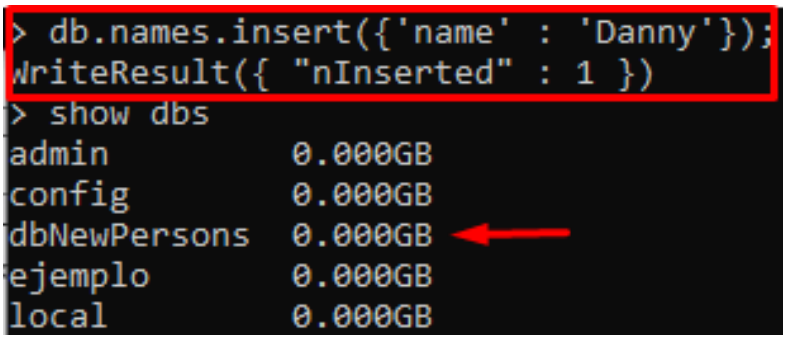

Figura 2: Agregar documentos a una colección.

Las bases de datos tienen un peso de 0.000GB debido a que los datos ingresados son muy pequeños y por eso no se visualizan.

Agregaremos mas documentos para poder continuar con los demás procesos CRUD.

```
> db.names.insert ({'name': 'Danny'});
WriteResult({ "nInserted" : 1 })
> db.names.insert ({'name': 'Yande'});
WriteResult({ "nInserted" : 1 })
> db.names.insert ({'name': 'Steven'});
WriteResult({ "nInserted" : 1 })
> db.names.insert ({'name': 'Jean'});
WriteResult({ "nInserted" : 1 })
> db.names.insert ({'name': 'Manuel'});
```

Figura 3: Agregar documentos a una colección.

2.3. Paso 3: Iterar los documentos de una colección

Para poder ver todos los documentos de una colección, se lo realiza mediante un proceso de dos pasos. Primero vamos a obtener una referencia a la colección, para ello usaremos el método next(). Se usará un ciclo while y dentro del mismo insertaremos el método hasNext(), esto hará que se obtenga el documento "siguiente" dentro de la colección. Mediante un printJson se imprimirá el valor retornado. Este proceso se puede observar en la Figura 4.

```
> db.names.find();
{ "_id" : ObjectId("61a6ea4b8a249360d9e62031"), "name" : "Danny" }
{ "_id" : ObjectId("61a6eb3a8a249360d9e62032"), "name" : "Juan" }
{ "_id" : ObjectId("61a6eb3a8a249360d9e62033"), "name" : "David" }
{ "_id" : ObjectId("61a6eb3a8a249360d9e62034"), "name" : "Damian" }
{ "_id" : ObjectId("61a6eb3a8a249360d9e62035"), "name" : "Jose" }
{ "_id" : ObjectId("61a6eb3a8a249360d9e62036"), "name" : "Adrian" }
{ "_id" : ObjectId("61a6eb3a8a249360d9e62037"), "name" : "Steven" }
{ "_id" : ObjectId("61a6eb3a8a249360d9e62038"), "name" : "Jose" }
{ "_id" : ObjectId("61a6eb3a8a249360d9e62039"), "name" : "Pedro" }
{ "_id" : ObjectId("61a6eb3a8a249360d9e6203a"), "name" : "Diego" }
{ "_id" : ObjectId("61a6eb3a8a249360d9e6203b"), "name" : "Pepe" }
{ "_id" : ObjectId("61a6eb3a8a249360d9e6203c"), "name" : "Juan" }
{ "_id" : ObjectId("61a6eb3a8a249360d9e6203d"), "name" : "Catherin" }
{ "_id" : ObjectId("61a6eb3a8a249360d9e6203e"), "name" : "Karen" }
```

Figura 4: Lectura de documentos de una colección.

El valor obtenido en la propiedad "id" de cada documento serán diferentes, esto debido a que estos son únicos, como se ve en la Figura 5.

2.4. Paso 4: Actualizar documentos de una colección

Para actualizar cualquier documento haremos uso de la propiedad "id", que ya vimos que cada documento tiene un id único. El proceso de actualización se lo lleva a cabo mediante el método

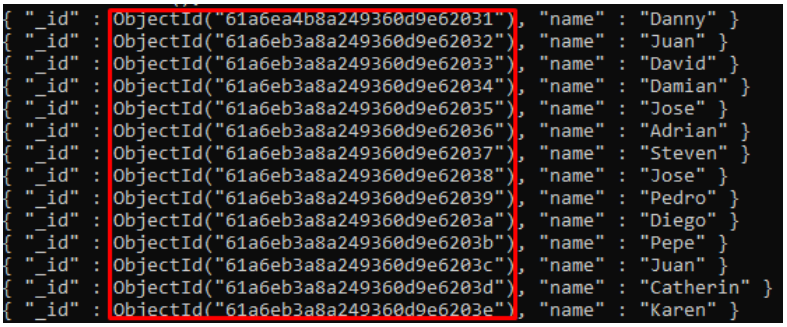

Figura 5: Ejecución comando show dbs.

"save". Pasaremos el "id" del documento que queremos actualizar. En la Figura 6 podemos observar cómo se realiza la actualización.

Figura 6: Actualización de documentos de una colección.

Ahora al imprimir los documentos de la colección podemos ver como se realizo de manera correcta la actualización del documento modificado.

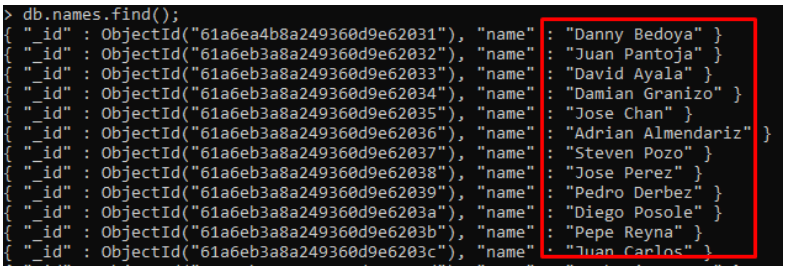

Figura 7: Comprobación del documento actualizado.

2.5. Paso 5: Eliminar documentos de una colección y base de datos

Para eliminar un procedimiento de una colección, se lo realiza mediante el método “remove”. Para ello también se hace uso de la propiedad “id” especificando el documento que se quiere eliminar

```
> db.names.remove({  "_id" : ObjectId("61a6eb3a8a249360d9e62032")});
WriteResult({ "nRemoved" : 1 })
> db.names.remove({  "_id" : ObjectId("61a6eb3a8a249360d9e62033")});
WriteResult({ "nRemoved" : 1 })
> db.names.remove({  "_id" : ObjectId("61a6eb3a8a249360d9e62034")});
WriteResult({ "nRemoved" : 1 })
> db.names.remove({  "_id" : ObjectId("61a6eb3a8a249360d9e62035")});
WriteResult({ "nRemoved" : 1 })
> db.names.remove({  "_id" : ObjectId("61a6eb3a8a249360d9e62036")});
WriteResult({ "nRemoved" : 1 })
> db.names.remove({  "_id" : ObjectId("61a6eb3a8a249360d9e62037")});
WriteResult({ "nRemoved" : 1 })
> db.names.remove({  "_id" : ObjectId("61a6eb3a8a249360d9e62038")});
WriteResult({ "nRemoved" : 1 })
> db.names.remove({  "_id" : ObjectId("61a6eb3a8a249360d9e62039")});
WriteResult({ "nRemoved" : 1 })
```

Figura 8: Eliminación del documento.

Finalmente volvemos a imprimir los documentos de la colección, para poder observar como el dato seleccionado ha sido removido.

```
> db.names.find();
{ "_id" : ObjectId("61a6eb3a8a249360d9e62045"), "name" : "Adrian" }
{ "_id" : ObjectId("61a6eb3a8a249360d9e62046"), "name" : "Fredy" }
{ "_id" : ObjectId("61a6eb3a8a249360d9e62047"), "name" : "Fernando" }
{ "_id" : ObjectId("61a6eb3a8a249360d9e62048"), "name" : "JAvier" }
{ "_id" : ObjectId("61a6eb3a8a249360d9e62049"), "name" : "Pedro" }
{ "_id" : ObjectId("61a6eb518a249360d9e6204a"), "name" : "Daniel" }
{ "_id" : ObjectId("61a6ebbc8a249360d9e6204b"), "name" : "Yande" }
{ "_id" : ObjectId("61a6ebbc8a249360d9e6204c"), "name" : "Steven" }
{ "_id" : ObjectId("61a6ebbc8a249360d9e6204d"), "name" : "Jean" }
```

Figura 9: Comprobación de documentos de la colección.

Para eliminar una base de datos se usa el comando ”dropDatabase”.

```
> db.dropDatabase();
{ "ok" : 1 }
```

Figura 10: Comprobación de documentos de la colección.

3. Results and Analysis

Para poder conocer la herramienta de MongoDB y determinar si es una buena opción para trabajar con bases de datos NoSQL. Realizaremos una comparación de Mongo y Cassandra mediante una tabla que nos ofrece cada una de estas herramientas. El Cuadro 1 muestra diferencias de sus características.

Cuadro 1: MongoDB y Cassandra.

MongoDB	Cassandra
Realiza copias de seguridad continua con coherencia entre clústeres y recuperación puntual.	No realiza copias de seguridad continua con coherencia entre clústeres y recuperación puntual.
Cuenta con cifrado de nivel de campo del lado del cliente.	Requiere un código de aplicación significativo, soporte de tipo de datos limitado, sin cumplimiento, sin consultas.
Tiene búsqueda de texto completo incorporada impulsada por Lucene.	No contiene búsqueda de texto completo incorporada impulsada por Lucene.

4. Discusión

Se aprendió el manejo de CRUD en MongoDB. CRUD hace referencia a las operaciones básicas, estas son la creación, lectura, actualización y eliminación de datos. En el caso de MongoDB se denominan documentos.

Cuando se habla de herramientas de bases de datos NoSQL tenemos como opción MongoDB y Cassandra. MongoDB contiene un nodo maestro que dirige múltiples nodos esclavos, por lo tanto, si el nodo maestro deja de funcionar, uno de los esclavos asume la función. Cassandra tiene varios nodos maestros, todo esto dentro de un clúster.

Se podría decir que MongoDB tiene una gran ventaja ante Cassandra debido a que admite muchos mas lenguajes de programación. Para realizar las operaciones básicas de CRUD en MongoDB se lo puede realizar desde el Shell mediante sus comandos básicos. Para insertar un documento se usa el método insert, para leer los documentos el método find, para actualizar el método save y para eliminar el método remove.

MongoDB tiene una amplia contra Couchbase y CouchbaseDB., entre las cuales destaca que MongoDB puede realizar transacciones ACID de varios documentos, mientras que Couchbase y CouchbaseDB.es muy limitado para este proceso. También MongoDB realiza copia de seguridad continua con coherencia entre clústeres y recuperación puntual, mientras que Couchbase y CouchbaseDB no.

Un atributo puede ser eliminado o a su vez modificado, para la mejor adaptación de la base de datos. La razón por la que un índice no cabe en la RAM puede ser por la cantidad de procesos que este realizando la misma simultáneamente.

5. Conclusión

MongoDB es una opción muy viable para trabajar en bases de datos NoSQL, gracias a sus múltiples características y a que es de código abierto, lo cual permite el fácil acceso a la herramienta. Se puede decir que los procedimientos almacenados CRUD son muy óptimos para realizar las operaciones de creación, lectura, actualización y eliminación de datos. Realiza estos procesos de manera óptima y reduce el tráfico de rey entre el cliente y el servidor.

Referencias

[1] https://conpilar.es/como-crear-y-eliminar-bases-de-datos-en-mongodb-desde-cli/. Accessed: 2021-12-1.

[2] Ángel Robledano. Qué es MongoDB, 08 2021.

[3] Fernando Tablado. Base de datos no relacional. ¿Qué es? Características y ejemplos, 09 2020.