Laboratorio No.1. Bases de Datos 2. NoSQL con MongoDB

Guía No. 1.

Bases de datos No Relacionales
No SQL

Docente:

Julian René Muñoz Burbano

Universidad Autónoma de Occidente

2025

Objetivo del taller

El objetivo de este taller es comprender los conceptos fundamentales de las bases de datos NoSQL, específicamente MongoDB. A través de ejercicios prácticos, los estudiantes aprenderán a:

- Crear una base de datos en MongoDB.
- Insertar documentos en una colección.
- Consultar datos en una colección.
- Actualizar registros en MongoDB.
- Eliminar documentos de una colección.
- Entender la diferencia entre bases de datos SQL y NoSQL.

Parte 1: Introducción a MongoDB

MongoDB es una base de datos NoSQL orientada a documentos que almacena información en formato JSON (BSON). A diferencia de las bases de datos relacionales (SQL), MongoDB no requiere esquemas fijos y permite un almacenamiento flexible y escalable.

Conceptos Clave

1. **Base de datos**: Conjunto de colecciones dentro de MongoDB.
2. **Colección**: Similar a una tabla en SQL, pero sin esquema fijo.
3. **Documento**: Registros en MongoDB, representados en formato JSON.
4. **Campo**: Similar a una columna en SQL, representa un atributo dentro de un documento.
5. **BSON**: Formato binario de JSON optimizado para almacenamiento en MongoDB.

Parte 2: Instalación y Configuración de MongoDB

Para ejecutar este taller, necesitas tener instalado MongoDB en tu sistema. Puedes seguir estos pasos:

1. **Descargar MongoDB Community Edition** desde https://www.mongodb.com/try/download/community.
2. **Instalar MongoDB** siguiendo las instrucciones para tu sistema operativo.
3. **Iniciar el servidor de MongoDB** ejecutando en la terminal:

   ```
   mongod
   ```

4. **Abrir la consola de MongoDB** con:

   ```
   mongo
   ```

Parte 3: Taller Práctico

Ejercicio 1: Creación de una Base de Datos y Colección

1. Abre la terminal y accede a la consola de MongoDB escribiendo:

```
mongo
```

2. Crea una base de datos llamada `taller_mongo`:

```
use taller_mongo
```

3. Verifica que estás en la base de datos correcta:

```
db
```

4. Crea una colección llamada `estudiantes` e inserta un documento:

```
db.estudiantes.insertOne({
    "nombre": "Juan Pérez",
    "edad": 21,
    "carrera": "Ingeniería de Sistemas",
    "ciudad": "Bogotá"
})
```

5. Verifica que el documento fue insertado:

```
db.estudiantes.find().pretty()
```

Ejercicio 2: Inserción de Múltiples Documentos

1. Inserta varios documentos en la colección `estudiantes`:

```
db.estudiantes.insertMany([
    {
        "nombre": "María Gómez",
        "edad": 22,
        "carrera": "Administración",
        "ciudad": "Medellín"
    },
    {
        "nombre": "Carlos Rodríguez",
        "edad": 24,
        "carrera": "Ingeniería de Software",
        "ciudad": "Cali"
    },
    {
        "nombre": "Ana Fernández",
        "edad": 23,
        "carrera": "Ciencias de la Computación",
        "ciudad": "Cartagena"
    }
])
```

2. Muestra todos los documentos:

```
db.estudiantes.find().pretty()
```

Ejercicios adicionales

1. Inserta un nuevo estudiante con los siguientes datos:

```
{
    "nombre": "Luis Torres",
    "edad": 26,
    "carrera": "Ingeniería Electrónica",
    "ciudad": "Pasto"
}
```

2. Consulta todos los estudiantes menores de 25 años.
3. Encuentra los estudiantes que estudian Ingeniería de Sistemas o Ingeniería de Software.
4. Actualiza la carrera de "María Gómez" a "Ingeniería Industrial".
5. Elimina todos los estudiantes que viven en "Cali".
6. Inserta un conjunto de documentos que representen profesores con los siguientes campos: nombre, edad, asignatura y experiencia en años.
7. Crea una colección llamada `cursos` e inserta varios documentos con el nombre del curso, la duración en semanas y el número de estudiantes inscritos.
8. Actualiza la información de un curso para aumentar su duración en 2 semanas.
9. Crea una consulta para encontrar todos los cursos con más de 30 estudiantes inscritos.
10. Elimina todos los documentos de la colección `estudiantes`.

Solución y Explicación de los Ejercicios Adicionales

1. **Inserta un nuevo estudiante con los siguientes datos:**

```
db.estudiantes.insertOne({
    "nombre": "Luis Torres",
    "edad": 26,
    "carrera": "Ingeniería Electrónica",
    "ciudad": "Pasto"
})
```

Explicación: Este comando inserta un documento con los datos proporcionados en la colección `estudiantes`.

2. **Consulta todos los estudiantes menores de 25 años.**

```
db.estudiantes.find({ "edad": { "$lt": 25 } }).pretty()
```

Explicación: Se usa `$lt` (less than) para filtrar estudiantes cuya edad es menor de 25.

3. **Encuentra los estudiantes que estudian Ingeniería de Sistemas o Ingeniería de Software.**

```
db.estudiantes.find({ "carrera": { "$in": ["Ingeniería de Sistemas", "Ingeniería
```

Explicación: El operador `$in` permite buscar coincidencias dentro de una lista de valores.

4. **Actualiza la carrera de "María Gómez" a "Ingeniería Industrial".**

```
db.estudiantes.updateOne(
    { "nombre": "María Gómez" },
    { "$set": { "carrera": "Ingeniería Industrial" } }
)
```

Explicación: Se usa `$set` para modificar solo el campo de carrera sin afectar los demás campos.

5. **Elimina todos los estudiantes que viven en "Cali".**

```
db.estudiantes.deleteMany({ "ciudad": "Cali" })
```

Explicación: `deleteMany` elimina todos los documentos que cumplen la condición.

6. **Inserta un conjunto de documentos que representen profesores.**

```
db.profesores.insertMany([
    { "nombre": "Dr. José Rojas", "edad": 45, "asignatura": "Bases de Datos", "e
    { "nombre": "Martha Silva", "edad": 38, "asignatura": "Programación", "exper
])
```

Explicación: `insertMany` permite agregar varios documentos a la colección `profesores`.

7. **Crea una colección `cursos` e inserta documentos.**

```
db.cursos.insertMany([
    { "nombre": "MongoDB Básico", "duracion": 6, "estudiantes_inscritos": 40 },
    { "nombre": "Desarrollo Web", "duracion": 8, "estudiantes_inscritos": 35 }
])
```

Explicación: Se crea la colección `cursos` y se insertan múltiples registros.

8. **Actualiza la información de un curso para aumentar su duración en 2 semanas.**

```
db.cursos.updateOne(
    { "nombre": "MongoDB Básico" },
    { "$inc": { "duracion": 2 } }
)
```

Explicación: `$inc` incrementa el valor del campo `duracion` en 2 semanas.

9. **Crea una consulta para encontrar todos los cursos con más de 30 estudiantes inscritos.**

```
db.cursos.find({ "estudiantes_inscritos": { "$gt": 30 } }).pretty()
```

Explicación: `$gt` (greater than) filtra cursos con más de 30 estudiantes.

10. **Elimina todos los documentos de la colección `estudiantes`.**

```
db.estudiantes.deleteMany({})
```

Explicación: Un objeto vacío `{}` en `deleteMany` elimina todos los documentos de la colección.

Con estos ejercicios, los estudiantes pueden reforzar su conocimiento en inserciones, consultas, actualizaciones y eliminación de datos en MongoDB.