Práctica 3
Base de Datos Neo4j

Facultad De Ingeniería, Universidad De Cuenca
BIG DATA
Freddy L. Abad L.
freddy.abadl@ucuenca.edu.ec

***Resumen*- El incremento exponencial de generación de datos a presentado múltiples problemas a nivel de bases de datos relacionales. Estos problemas se ven desbordados en cuanto a tiempos de respuesta, organización de datos, entre otros. Una de las posibles soluciones existentes son las bases de datos NoSQL, esta tiene diversos enfoques, dependiendo el dominio del problema a resolver. Las Bases de Datos NoSQL de Grafos tienen diversas ventajas y son usadas en múltiples campos, este articulo detalla sobre estas bases de datos, enfocando su desarrollo con Neo4J.**
***Palabras Clave*- NoSQL, Neo4j, Grafos, Angular.**

I. INTRODUCCIÓN

Las bases de datos NoSQL plantean modelos de datos de esquemas flexibles que se adaptan al requerimiento de las aplicaciones actuales[1]. Estas no son tabulares y almacenan datos de manera diferente a las tablas relacionales [2] [3].

Objetivos

a. Objetivo General

- Implementar una aplicación de una base de datos NoSQL basado en grafos

b. Objetivo Especifico

- Analizar las funcionalidades que otorga una base de datos dirigido por grafos
- Implementar una aplicación usando Neo4J

II. MARCO TEORICO

Esta sección desarrolla los tópicos tratados en la metodología de la práctica de bases de datos dirigidos por grafos.

Base de Datos: Las bases de datos se puede entender como un conjunto de información relacionada organizada de manera grupal o estructurada[4]. Estos tienen la finalidad de ser recolectados y explotados por los sistemas de información de una empresa[2]. Las bases de datos pueden ser relacionales o no relacionales, determinadas por su dominio de uso y la forma en que estructural los datos. *Ver imagen 1*

Imagen 1: Concepto de base de datos como repositorio de información

Base de Datos NoSQL: El entorno de desarrollo de este artículo se torna alrededor de las bases no relacionales no relacionales o NoSQL. Estas no son tabulares y almacenan datos de manera diferente a las tablas relacionales, Las bases de datos NoSQL vienen en una variedad de tipos según su modelo de datos[2]. Estas bases de datos proporcionan esquemas flexibles y escalan fácilmente con grandes cantidades de datos y grandes cargas de usuarios. Las bases NoSQL se pueden clasificar según su tipología como: bases de datos documentales, clave-valor, dirigido por ancha y dirigidos por grafos [2].

Grafo: Los grafos "son una composición de un conjunto de objetos conocidos como nodos que se relacionan con otros nodos a través de un conjunto de conexiones conocidas como aristas"[5]. Estas composiciones permiten representar diversas situaciones complejas. Estas características hacen que su implementación sea amplia en estudios de ciencias exactas, ciencias sociales y en aplicaciones informáticas [5] . Los grafos pueden ser dirigidos, no dirigidos y etiquetados. (Ver imagen 2)

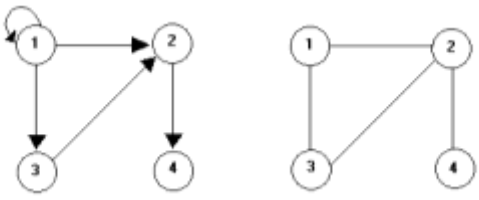

Imagen 2: Grafo dirigido y no dirigido

Base de Datos dirigido por Grafo: Estas son bases de datos NoSQL, caracterizándose por organizar la información de forma que se crean nodos de grafos, a diferencia de una relacional que

crea registros de una tabla. Representan la información en vértices y aristas cumpliendo con la Teoría de Grafos. Un grafo estará compuesto por dos elementos: los nodos y las relaciones. Un nodo representa una entidad, donde se almacenan piezas de datos o atributos de tipo clave-valor, mientras que las relaciones representan cómo se conectan y se asocian dos nodos [6]. Las BDOG dentro del teorema de CAP se pueden englobar dentro de CA (Consistencia y Disponibilidad) o PA (Tolerancia a particionamiento, Disponibilidad) [6]. Entre sus ventajas, incluyen no inferir conexiones de datos usando claves externas o procesamiento fuera de banda, como MapReduce. Su concepción detallo en ser significativamente más simple y expresivo que las bases de datos relacionales. Además de ser flexibles, agiles y de rápido performance. Están diseñadas teniendo en cuenta la integridad transaccional y la disponibilidad operativa [7].

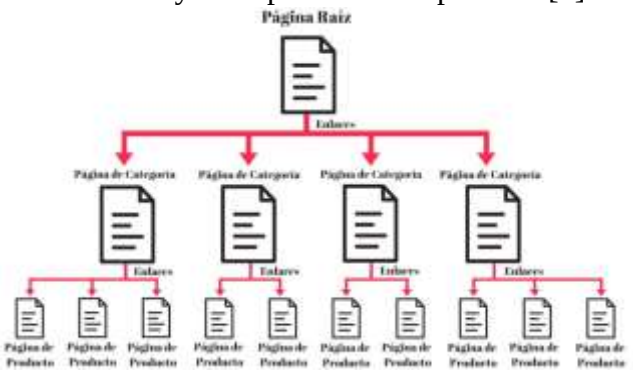

Imagen 3: Organización de una base de datos orientada a grafos

SPA: Las SPA o Single Page Aplicación son una tipología de aplicación web donde todas las pantallas las muestra en la misma página, sin recargar el navegador, es decir, un sitio donde existe un único punto de entrada. El uso de una SPA se realiza debido a que se mantiene el enfoque de varias vistas y una sola página. Incrementa su tiempo de carga, usa arquitecturas modernas y su mantenibilidad es exponencialmente mas sencilla que una aplicación web de muchas páginas, pocas vistas [8]. El framework de frontend AngularJS, utiliza este enfoque en su implementación desde la versión 1, hasta la actual (v12).

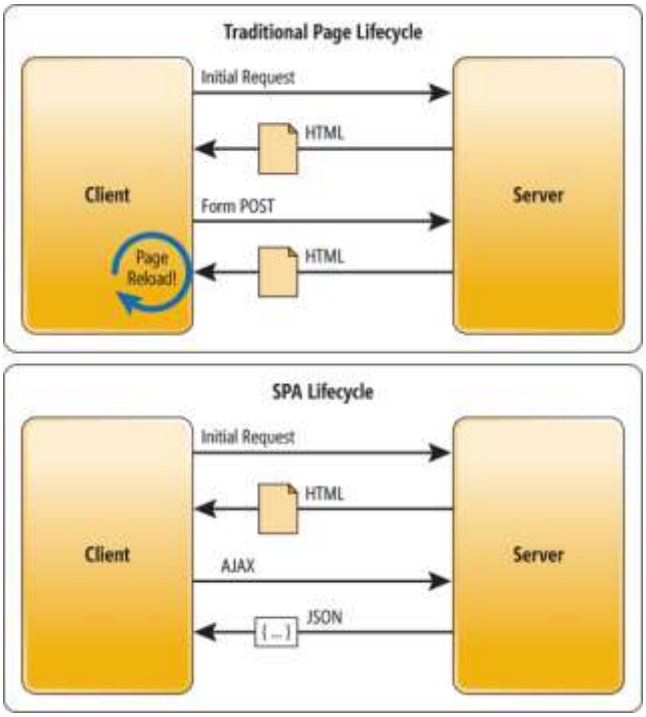

Imagen 4: Arquitectura tradicional vs Arquitectura SPA.

III. METODOLOGÍA

El código fuente de esta práctica se encuentra en el siguiente link [https://bit.ly/34Fmaj7]. Esta práctica contempla una aplicación en Angular 10 con almacenamiento de datos en Neo4j.

A. Proyecto Angular

El proceso de creación de un proyecto en angular se inicia con la sentencia de la Imagen 5 que se ingresa en el terminal.

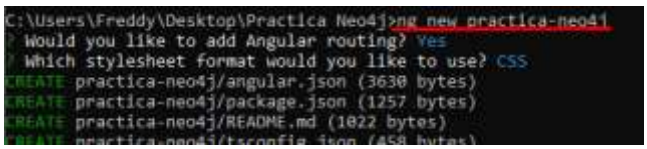

Imagen 5: Creación del proyecto de angular

El desarrollo de tal aplicación web, se realizó mediante el gestor de interfaz gráfica (ver imagen 6) La vista se puede visualizar en la pantalla de la imagen 6.

Imagen 6: Instalación de Bootstrap

Imagen 7: Vista del SPA

B. Instalación de Neo4j

El proceso de instalación en Windows, es como la mayoría de gestores de bases de datos. Inicialmente se descarga el instalador, se procede con todos los pasos

de setup, configuración de contraseñas. El proceso de instalación concluye cuando se visualiza la imagen 8. En la Imagen 9 se visualiza la pantalla para la creación de una base de datos, la imagen 10 refiere a la configuración de base de datos. Finalmente, en la imagen 11 se visualiza la inicialización del servidor de base de datos, en la imagen 12 la Pantalla de Inicio de la Base de Datos creada y en la imagen 13 una inicialización de la base de datos con data de prueba.

Imagen 8: Proceso finalización de instalación del gestor.

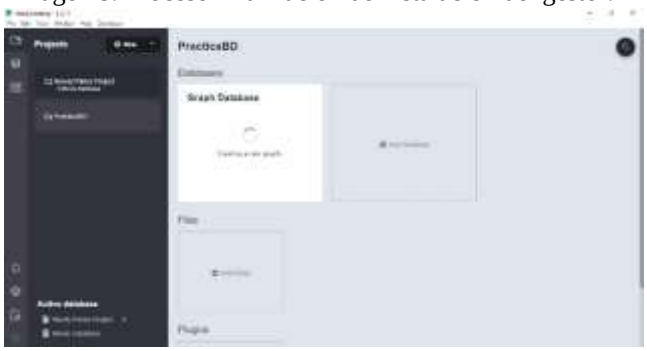

Imagen 9: Creación de una base de datos

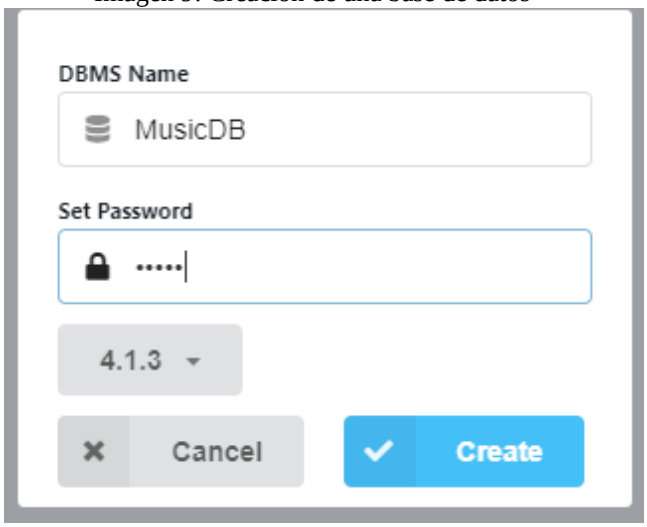

Imagen 10: Configuración de base de datos

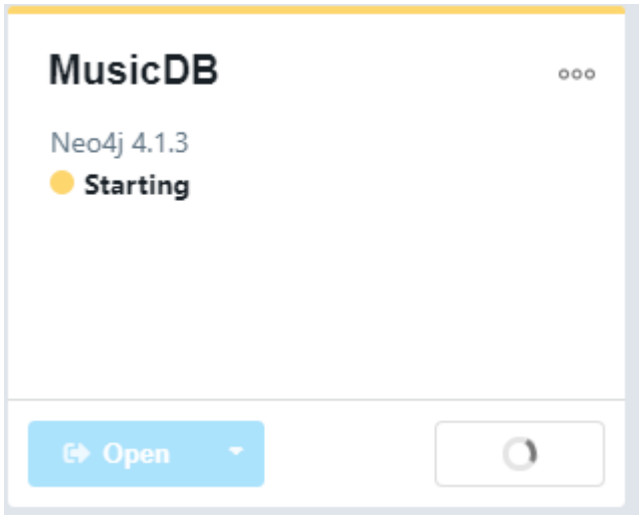

Imagen 11: Inicialización del servidor de base de datos.

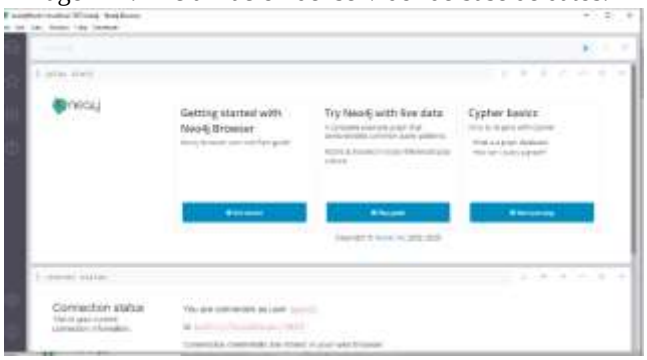

Imagen 12: Pantalla de Inicio de la Base de Datos creada

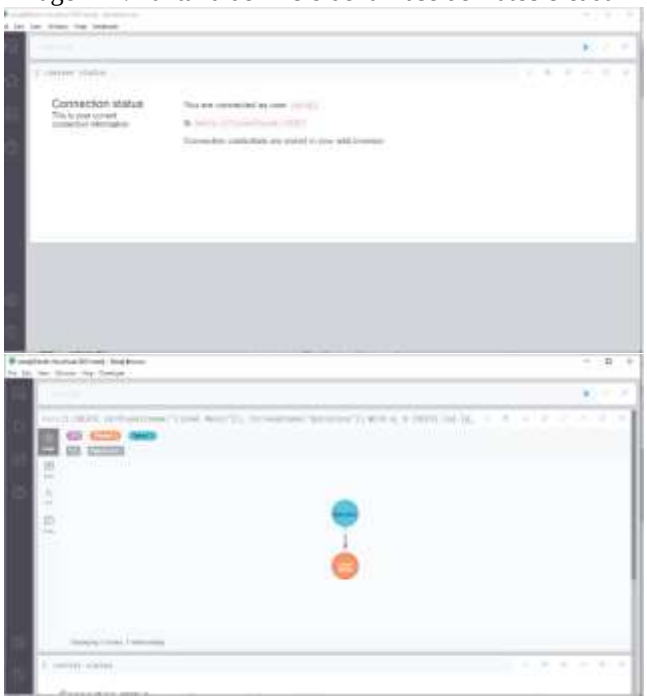

Imagen 13: Inicialización de la base de datos con data de prueba

C. Setup Neo4j Angular

La conexión entre el aplicativo web y la base de datos se hace mediante un driver, disponible en [9]. La sección importante de conexión se detalla en la imagen

Imagen 14: Instalacion de la librería disponible en [9]

```
import { AngularNeo4jService } from './angular-neo4j.service';
...
constructor(private neo4j: AngularNeo4jService) {}
```

Imagen 15: Importacion de librerias desde el componente de la vista.

```
const url = 'bolt://localhost:7687';
const username = [redacted];
const password = [redacted];
const encrypted = true;

this.neo4j
    .connect(
      url,
      username,
      password,
      encrypted
    )
    .then(driver => {
      if (driver) {
        console.log(`Successfully connected to ${url}`);
      }
    });
```

Imagen 16: Conexión con la base de datos a través del driver [9]

```
const query = 'MATCH (n:USER {name: {name}}) RETURN n';
const params = { name: 'bob' };

this.neo4j.run(query, params).then(res => {
    console.log(res);
  });
```

Imagen 17: Sentencia ejemplo usada en Neo4J.

D. Sentencias CRUD

Las sentencias desarrolladas son la creación, la consulta y la eliminación de los registros.

Creación de nodo y relación

La creación implica la implementación de un nodo y la relación con otro, la cual se detalla en la imagen 18. Los datos insertados se detallan en la imagen 19. La consulta grafica mediante el gestor de la base de datos se puede visualizar en la imagen 20.

```
public crearUsuario(usuario1: string, usuario2: string) {
  var query =
    ' CREATE ' +
    '   (' + usuario1 + ':Person{name:' + usuario1 + '}), ' +
    '   (' + usuario2 + ':Person{name:' + usuario2 + '}) ' +
    ' WITH m, b ' +
    ' CREATE (' + usuario1 + ')-[:FriendsWith]->(' + usuario2 + ') ';
  this.neo4j.run(query).then(res => {
    console.log(res);
  });
}
```

Imagen 18: Creación de nodo y relación

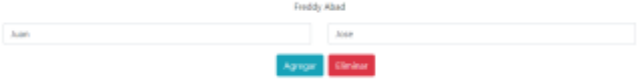

Imagen 19: Datos insertados

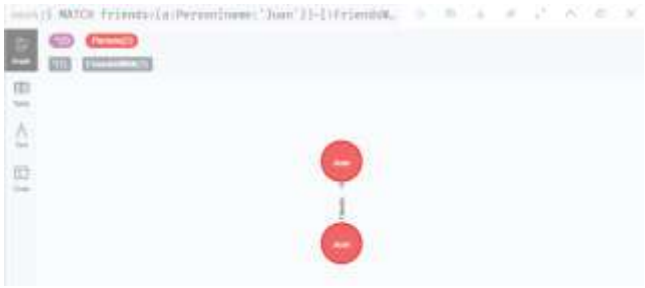

Imagen 20: Consulta en el gestor de la base de datos

Consulta de nodo y relación

La consulta implica la creación de un nodo y la relación con otro, la cual se detalla en la imagen 21.

```
public editarUsuario(usuario1: string, usuario2: string) {
  var query = 'MATCH (n:Person {' + usuario1 + ':' + usuario2 + '}) RETURN n'
  this.neo4j.run(query).then(res => {
    console.log(res);
  });
}
```

Imagen 21: Consulta de nodo y relación

Eliminación de nodo y relación

La eliminación implica el borrado del nodo y las relaciones con otros, la cual se detalla en la imagen 22.

```
public eliminarUsuario(usuario1: string) {
  var query =
    ' MATCH ('+usuario1+':Person)-[rel:IS_RELATED_TO]->(a)' +
    +'DELETE rel; ';
  this.neo4j.run(query).then(res => {
    console.log(res);
  });
}
```

Imagen 22: Eliminación de nodo y relaciones asociadas

IV. CONCLUSIONES

En esta práctica se pudo identificar los distritos Query permitidos en Neo4j para un CRUD, su implementación como representación de una base de datos orientada a grafos, permitió identificar las ventajas de su uso. El no contar con un esquema definido, tal como lo hacen las bases de datos relacionales, permite implementar en problemas complejos, gracias a su flexibilidad. Uno de los puntos que a futuro se debería estudia es el uso en bases de datos pesadas y recurrentes, desde el punto de vista del desarrollador, así se podrá obtener un criterio mas formado respecto a estas bases de datos NoSQL.

V. BIBLIOGRAFÍA

[1] GrapherEverywhere, “Bases de Datos NoSQL | Qué son, marcas, tipos y ventajas,” 2019. https://www.grapheverywhere.com/bases-de-datos-nosql-marcas-tipos-ventajas/ (accessed Dec. 23, 2020).

[2] M. Editors, “What is NoSQL? NoSQL Databases Explained | MongoDB,” 2020. https://www.mongodb.com/nosql-explained

(accessed Dec. 23, 2020).

[3] A. Editors, “Bases de datos no relacionales | Bases de datos de gráficos | AWS,” 2019. https://aws.amazon.com/es/nosql/ (accessed Dec. 23, 2020).

[4] P. V. Damian, “¿Qué son las bases de datos?,” 2007. http://www.maestrosdelweb.com/que-son-las-bases-de-datos/ (accessed Dec. 23, 2020).

[5] GrapherEverywhere, “Qué son los grafos,” 2019. https://www.grapheverywhere.com/que-son-los-grafos/ (accessed Dec. 23, 2020).

[6] GrapherEverywhere, “Base de datos orientada a Grafos. ¿Qué son y para qué se usan? - Sngular,” 2016. https://www.sngular.com/es/que-son-bases-datos-a-grafos/ (accessed Dec. 23, 2020).

[7] Neo4j Editors, “Why Graph Databases?,” 2020. https://neo4j.com/why-graph-databases/ (accessed Dec. 23, 2020).

[8] Guerrero Nataly, “¿Qué es una SPA? (Angular)),” 2019. https://www.programaenlinea.net/una-spa-angular/ (accessed Dec. 23, 2020).

[9] Salnikov Maxim, “angular-neo4j - npm.” https://www.npmjs.com/package/angular-neo4j (accessed Dec. 23, 2020).