TRABAJO PRACTICO NoSQL

Laboratorio 2

Grupo: 6

Fecha de entrega: 25/11/2024

Integrantes: Francisco Campora, Mora Otegui, Bruno Donato, Joaquin Serra

Consigna

1. Proponer y describir un caso real que requiera modelarse y diseñar una base de datos que cumpla con las necesidades de almacenamiento de información.
2. Modelar los objetos/entidades/tablas/relaciones y demás componentes requeridos o mencionados para el caso.
3. Seleccionar el paradigma de base de datos que crean más conveniente para llevar a cabo la implementación de la solución.
4. Justificar su elección y realizar un beanchmarking (comparación) contra alguno de los demás paradigmas vistos en clase, resaltando las ventajas y desventajas del paradigma elegido por ustedes frente a ese otro.
5. Entregar scripting ejemplificando operaciones sobre dicha base de datos (no es necesario codificar toda la operatoria del caso de uso seleccionado):
    - **a.** Configuración de ambiente de trabajo (base de datos).
    - **b.** Creación de objetos.
    - **c.** Carga de registros.
    - **d.** Modificación y consulta según casuísticas del caso desarrollado

1. Caso

El caso se centra en una tienda de ropa que busca optimizar la gestión y análisis de su información mediante una base de datos. Por el momento la tienda vende algunos productos como camisetas, pantalones, accesorios y buzos, pero tienen pensado agregar nuevos artículos. Los artículos están organizados en categorías (casual, deportiva), estilos (moderno, clásico, overzise), marca, y además dependiendo de la temporada algunos productos pueden estar en promoción.

La tienda cuenta con usuarios, clientes que realizan compras, quienes pueden utilizar distintos métodos de pago, como efectivo o tarjeta. Estas transacciones son gestionadas por empleados. Los productos se compran a distintos proveedores, registrando su precio y la cantidad.

2. Modelado de nodos y relaciones para el caso

NODOS con sus posibles atributos

producto ({nombre}, {precio}, {marca}, {estilo}, {descuento}, {stock}, {color})
cliente ({nombre}, {apellido}, {telefono}, {productos_comprados})
categoría ({nombre}, {temporada})
proveedor ({nombre}, {teléfono}, {direccion}, {ciudad})
empleado ({nombre},{apellido}, {nro_empleado})
promocion ({nombre}, {descuento}, {fecha_inicio}, {fecha_fin})

RELACIONES en general con sus posibles atributos

cliente - [:COMPRO{fecha}, {metodo_pago}] -> producto
producto - [:PERTENECE_A] -> categoría
proveedor - [:PROVEE{precio},{cantidad}] -> producto
empleado - [:ATIENDE{fecha}] -> cliente
promoción - [:APLICA_A] -> producto

3. Elección paradigma de BBDD

El sistema propuesto para la tienda de ropa requiere modelar relaciones complejas entre nodos como clientes, productos, categorías, estilos o ventas. Este tipo de modelo es naturalmente representado como un grafo, se utilizará la herramienta neo4J para levantar esta base de datos basada en grafos que representan las entidades en forma de nodos. Además cada nodo y relación puede tener sus propios atributos cuantitativos y cualitativos. Debido a la flexibilidad en el esquema se podrán agregar productos, clientes y lo que sea necesario, ya que la base puede adaptarse fácilmente a los cambios sin alterar la estructura. Gracias a sus herramientas gráficas resulta intuitivo y facilita la comprensión de las relaciones.

4. Comparación con otras bases NoSQL.

Comparando los tipos de bases de datos NoSQL con Neo4J podemos deducir que ésta es la mejor opción, ya que las relaciones son críticas y el diseño debe ser flexible. En contraposición encontramos bases como Cassandra que está optimizada para datos distribuidos y operaciones de escritura/lectura masivas, pero no para relaciones complejas, ya que las relaciones tendrían que gestionar manualmente en múltiples tablas, lo que resulta en redundancia de datos; MongoDB es una base de datos NoSQL orientada a documentos, aunque permite modelar relaciones, estas son más adecuadas para jerarquías simples y no para consultas relacionales complejas.

Neo4J utiliza el lenguaje cypher, que permite que las consultas relacionadas con conexiones y relaciones entre nodos sean altamente optimizadas y más fáciles de escribir; en cambio, otras bases de datos como Mongo o Cassandra, no tienen optimizada la búsqueda de relaciones y las consultas pueden llegar a ser lentas y complejas.

Con respecto a la escalabilidad del sistema, MongoDB y Cassandra son diseñadas para una enorme cantidad de datos distribuidos; Neo4J tal vez no es tan

escalable, pero al ser una tienda de ropa jamás deberá procesar millones de transacciones diarias.

Neo4J tiene la particularidad de que su estructura es totalmente flexible, por lo que se pueden agregar, modificar o eliminar todos los nodos, relaciones, atributos de manera sencilla sin la necesidad de rediseñar toda la base.

5. Scripting

a. Se utiliza cualquier consola de Neo4J, la más accesible ya que no se debe iniciar sesion es: https://console.neo4j.org/

b. Creacion de objetos, relaciones y carga de registros

```
CREATE
(c1:Cliente {nombre: "Iker", apellido: "Romero", productos_comprados: 2}),
(c2:Cliente {nombre: "Pedro", apellido: "Sanchez", productos_comprados: 1}),
(c3:Cliente {nombre: "Carlos", apellido: "Larrea", productos_comprados: 4}),

(p1:Producto {nombre: "camiseta rayada", precio: 17000, marca: "Lacoste",
descuento: 0, stock: 10, color: "Celeste"}),
(p2:Producto {nombre: "Buzo Argentina", precio: 25000, marca: "Adidas",
estilo: "Oversize", descuento: 50, stock: 1, color: "Blanco"}),
(p3:Producto {nombre: "jean 502", precio: 150000, marca: "Levis", estilo:
"straight", descuento: 15, stock: 100, color: "negro"}),
(p4:Producto {nombre: "bermuda cargo", precio: 48500, marca: "Pampero",
estilo: "cargo", stock: 15, color: "gris"}),
(p5:Producto {nombre: "camisa cuadros", precio: 96000, marca: "Polo", estilo:
"entallada", stock: 25, color: "blanco"}),

(cat1:Categoria {nombre: "camiseta"}),
(cat2:Categoria {nombre: "buzo"}),
(cat3:Categoria {nombre: "pantalon"}),
```

```
(cat4:Categoria {nombre: "bermuda"}),
(cat5:Categoria {nombre: "camisa"}),

(prov1:Proveedor {nombre: "Polo", telefono: "2324-518196", direccion:
"Cordoba 5000", ciudad: "CABA"}),
(prov2:Proveedor {nombre: "Pampero", telefono: "2322-528295", direccion:
"Santa Fe 2000", ciudad: "CABA"}),
(prov3:Proveedor {nombre: "Levis", telefono: "2323-118899", direccion: "Mario
Bravo 2050", ciudad: "CABA"}),
(prov4:Proveedor {nombre: "Adidas", telefono: "011-4959-3400", direccion:
"Paraguay 4959", ciudad: "CABA"}),
(prov5:Proveedor {nombre: "Lacoste", telefono: "011-4559-0400", direccion:
"Honduras 4959", ciudad: "CABA"}),

(emp1:Empleado {nombre: "Joaquin", apellido: "Serra", nro_empleado: 10}),
(emp2:Empleado {nombre: "Francisco", apellido: "Campora", nro_empleado:
27}),
(emp3:Empleado {nombre: "Tomas", apellido: "Gomez", nro_empleado: 4}),

(promo1:Promocion {nombre: "Cyber Week", descuento: 25, fecha_inicio: "11-
11-2024", fecha_fin: "18-11-2024"}),
(promo2:Promocion {nombre: "2x1", descuento: 50, fecha_inicio: "21-11-
2024", fecha_fin: "25-12-2024"}),

(c1)-[:COMPRO {fecha: "23-11-2024", metodo_pago: "Mercado Pago"}]->(p3),
(c1)-[:COMPRO {fecha: "12-10-2020", metodo_pago: "Efectivo"}]->(p5),
(c2)-[:COMPRO {fecha: "15-08-2024", metodo_pago: "Mercado Pago"}]->(p4),
(c3)-[:COMPRO {fecha: "05-03-2024", metodo_pago: "Efectivo"}]->(p3),
(c3)-[:COMPRO {fecha: "05-03-2024", metodo_pago: "Efectivo"}]->(p1),
(c3)-[:COMPRO {fecha: "12-12-2023", metodo_pago: "Tarjeta"}]->(p3),
(c3)-[:COMPRO {fecha: "23-07-2023", metodo_pago: "Tarjeta"}]->(p5),
```

(p1)-[:PERTENECE_A]->(cat1),
(p2)-[:PERTENECE_A]->(cat2),
(p3)-[:PERTENECE_A]->(cat3),
(p4)-[:PERTENECE_A]->(cat4),
(p5)-[:PERTENECE_A]->(cat5),

(prov1)-[:PROVEE {precio_unidad: 33000, cantidad: 25}]->(p5),
(prov2)-[:PROVEE {precio_unidad: 20500, cantidad: 15}]->(p4),
(prov3)-[:PROVEE {precio_unidad: 90000, cantidad: 100}]->(p3),
(prov4)-[:PROVEE {precio_unidad: 14000, cantidad: 1}]->(p2),
(prov5)-[:PROVEE {precio_unidad: 10000, cantidad: 10}]->(p1),

(emp1)-[:ATIENDE {fecha: "05-03-2024"}]->(c3),
(emp2)-[:ATIENDE {fecha: "23-07-2023"}]->(c3),
(emp2)-[:ATIENDE {fecha: "12-12-2023"}]->(c3),
(emp3)-[:ATIENDE {fecha: "15-08-2024"}]->(c2),
(emp1)-[:ATIENDE {fecha: "23-11-2024"}]->(c1),
(emp1)-[:ATIENDE {fecha: "12-10-2020"}]->(c1),

(promo1)-[:APLICA_A]->(p2),
(promo2)-[:APLICA_A]->(p3);

d. Consultas de algunas casuísticas comunes

1. Mostrar la dirección de los proveedores que proveen buzos

```
MATCH(prov:Proveedor)-[:PROVEE]->(prod:Producto)-[:PERTENECE_A]->(cat:Categoria {nombre: "buzo"})
RETURN prov.direccion AS Direccion_proveedor;
```

2. Mostrar todos los números de empleados que atendieron a “Iker Romero”

```
MATCH(e:Empleado)-[:ATIENDE]->(c:Cliente{nombre:"Iker", apellido:"Romero"})
RETURN e.nro_empleado AS numero_empleado;
```

3. Mostrar los apellidos de los clientes que compraron camisetas

```
MATCH(c:Cliente)-[:COMPRO]->(p:Producto)-[:PERTENECE_A]->(cat:Categoria{nombre: "camiseta"})
RETURN c.apellido AS apellido_cliente;
```

4. Mostrar todos los nombres de los productos y sus marcas que aplican la promoción “2x1”

```
MATCH(promo:Promocion{nombre: "2x1"})-[:APLICA_A]->(p:Producto)
RETURN p.nombre AS nombre_producto, p.marca AS marca_producto;
```

5. Mostrar el nombre de los productos que aplican a la promoción de “Cyber Week” y que los compró el cliente llamado “Carlos Larrea”

```
MATCH(promo:Promocion{nombre:"Cyber Week"})-[:APLICA_A]->(p:Producto),
(c:Cliente {nombre: "Carlos", apellido: "Larrea"})-[:COMPRO]->(p)
RETURN p.nombre AS nombre_producto;
```

6. Mostrar el nombre y stock disponible de los productos con un precio mayor a $50000

```
MATCH(p:Producto) WHERE p.precio > 50000
RETURN p.nombre AS Nombre_producto, p.stock AS Stock_disponible;
```