TUTORIAL de NEO4J

En este tutorial vamos a aprender los comandos básicos y la sintaxis de Cypher. También utilizaremos la herramienta de visualización Screencast.

Lo primero que debemos hacer es descargar Neo4j de la página web. Trabajaremos con la community edition, que es gratuita y ofrece todas las funcionalidades necesarias apra proyectos pequeños:

https://neo4j.com/download/community-edition/

Una vez instalado, lo abrimos y utilizamos la ubicación por defecto para lanzar Neo4j en el navegador. En Chrome o Firefox navegamos a:

http://localhost:7474/

Y ya podemos empezar.

Introducción a Cypher:

Cypher es el lenguaje propio de Neo4j. Su sintaxis utiliza un estilo ascii-art para hacer los comandos muy intuitivos.

Los nodos se representan con círculos y las relaciones con flechas. Su representación en ascii-art consiste en poner nodos entre paréntesis y relaciones como flechas etiquetadas por corchetes:

```
(nodo)-[:RELACIÓN]->(nodo)
```

Las propiedades de los nodos o las relaciones se indican con una estructura parecida a un diccionario:

```
(nodo {name:'Oscar', surname:'Garcia'})
```

las etiquetas se indican después de definir la variable:

```
(nodo:Person {name:'Oscar', surname:'Garcia'})
```

Empezamos por crear un nodo:

```
CREATE (os {name:'Óscar', surname:'García'})
```

Veamos si se ha creado:

```
MATCH (n) RETURN n
```

Creemos un par de nodos más:

```
CREATE
    (va {name:'Vanessa', surname:'Gómez'}),
    (co {name:'Coffee', type:'Drink'}),
    (na {name:'Napolitana', type:'Dessert'})
```

Visualizemos el grafo:

```
MATCH (n) RETURN n
```

Vamos a añadir algunas etiquetas (ejecutar cada linea por separado):

```
MATCH (os {name:'Óscar'}) SET os:Person
MATCH (va {name:'Vanessa'}) SET va:Person
MATCH (co {name:'Coffee'}) SET co:Food
MATCH (na {name:'Napolitana'}) SET na:Food
```

Visualicemos otra vez para ver el cambio.

Ahora vamos a establezer una relación entre dos nodos:

```
MATCH (o {name:'Óscar'}), (c {name:'Coffee'})
CREATE (o)-[:ADDICTED_TO]->(c)
```

Podemos crear relaciones entre nodos de forma simultanea:

```
MATCH (p:Person), (f:Food)
CREATE (p)-[:CONSUME]->(f)
```

Ahora vamos a crear un par de nodos más y relacionarlos en un sólo bloque:

```
CREATE
    (sug:Ingredient {name:'Sugar', type:'Liquid'}),
    (wat:Ingredient {name:'Water', type:'Liquid'})
WITH sug, wat
MATCH
    (na {name:'Napolitana'}),
    (co {name:'Coffee'})
CREATE
    (sug)-[:REQUIRED]->(na),
    (wat)-[:REQUIRED]->(co)
```

Ahora hagamos una query sencilla para visualizar los nodos con la etiqueta Person:

```
MATCH (n:Person) RETURN n
```

Hagamos la misma query pero visualizando un par de atributos en forma tabular:

```
MATCH (n:Person) RETURN n.name AS Names, n.surname AS Surnames
```

Probemos una query relacional:

```
MATCH (o {name:'Óscar'})-[:CONSUME]->(f) RETURN f
```

La misma query pero visualizando un atributo en formato tabular:

```
MATCH (o {name:'Óscar'})-[:CONSUME]->(f) RETURN f.name AS Food
```

Un par de queries más. ¿Qué hacen?:

```
MATCH (o {name:'Óscar'})-[r]-(f) RETURN type(r)
MATCH (n:Person) WHERE n.name = 'Vanessa' RETURN n.surname AS
Surname
```

Ya estamos listos para hacer el tutorial con el movie graph. Primero eliminemos completamente nuestro grafo:

```
MATCH (n) DETACH DELETE n
```

Este comando sólo es útil con grafos pequeños, pero de momento nos sirve.

A continuación iniciaremos el tutorial de Neo4j que emplea una base de datos de películas, actores y directores:

```
:play movie graph
```

Gracias!