

Graph Databases - Neo4J

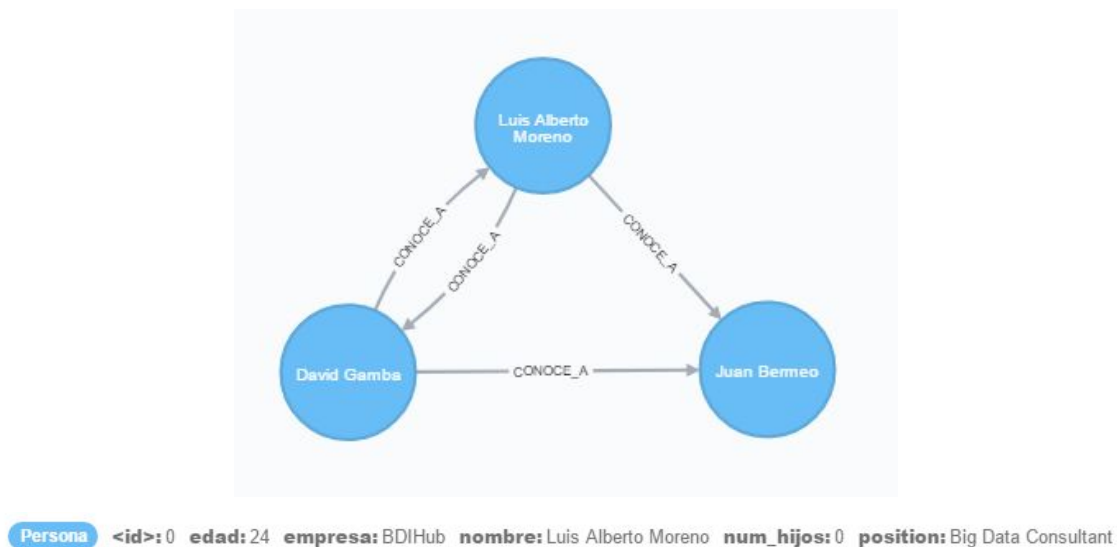
Las bases de datos de grafos abordan una de las grandes tendencias comerciales macroscópicas de la actualidad: aprovechar relaciones complejas y dinámicas en datos altamente conectados para generar conocimiento y ventaja competitiva. Ya sea que deseemos comprender las relaciones entre clientes, elementos en una red de teléfono o centro de datos, productores y consumidores de entretenimiento, o genes y proteínas, la capacidad de comprender y analizar vastos grafos de datos altamente conectados será clave para determinar qué compañías superan a sus competidores en la próxima década.

Para datos de cualquier tamaño o valor significativo, las bases de datos orientadas a grafos son la mejor manera de representar y consultar datos conectados. Los datos conectados son datos cuya interpretación y valor requieren primero comprender las formas en que se relacionan sus elementos constitutivos. La mayoría de las veces, para generar esta comprensión, necesitamos nombrar y calificar las conexiones entre las cosas.

A continuación realizaremos algunos ejemplos con dos grafos distintos.

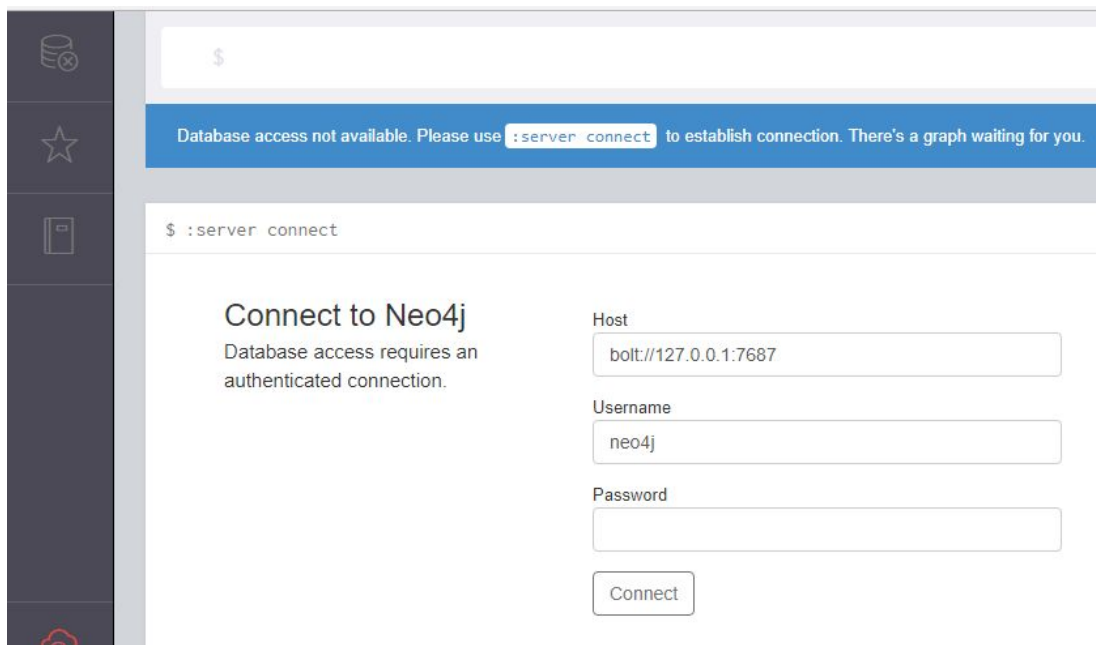
Grafo del workshop

Para este ejemplo crearemos un grafo con la siguiente estructura, en el cual se puede visualizar la relación entre varias personas:



Actividad 1

1. Inicie sesión en el browser de Neo4j de la instancia creada para este laboratorio, para esto visite la página <http://127.0.0.1:7474>, es posible que deba conectarse a una URL diferente, valide los datos de conexión con el instructor, allí visualizará una interfaz similar a la siguiente:



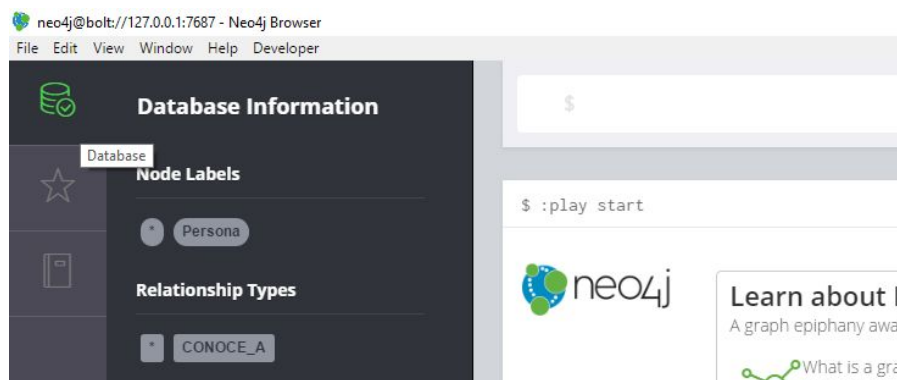
2. Ingrese las credenciales

Host: bolt://127.0.0.1:7687 o la que el instructor le indique

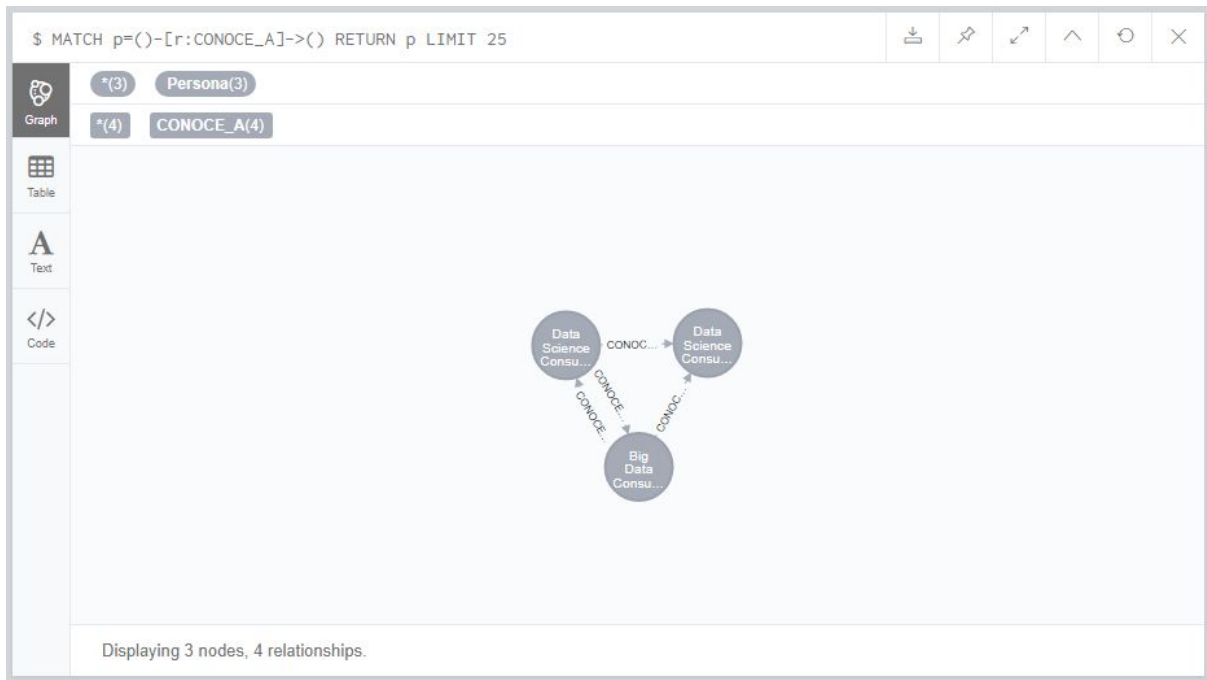
Username: neo4j

Password: bdih_neo4j

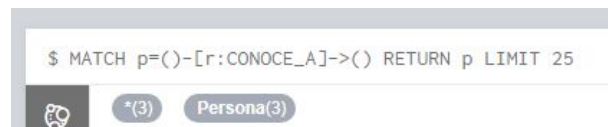
3. En esta interfaz podrá visualizar que los nodos mostrados anteriormente ya se encuentran creados, para esto presione el ícono de base de datos ubicado en la parte izquierda de la pantalla y luego de clic sobre la relación llamada **"CONOCE_A"**



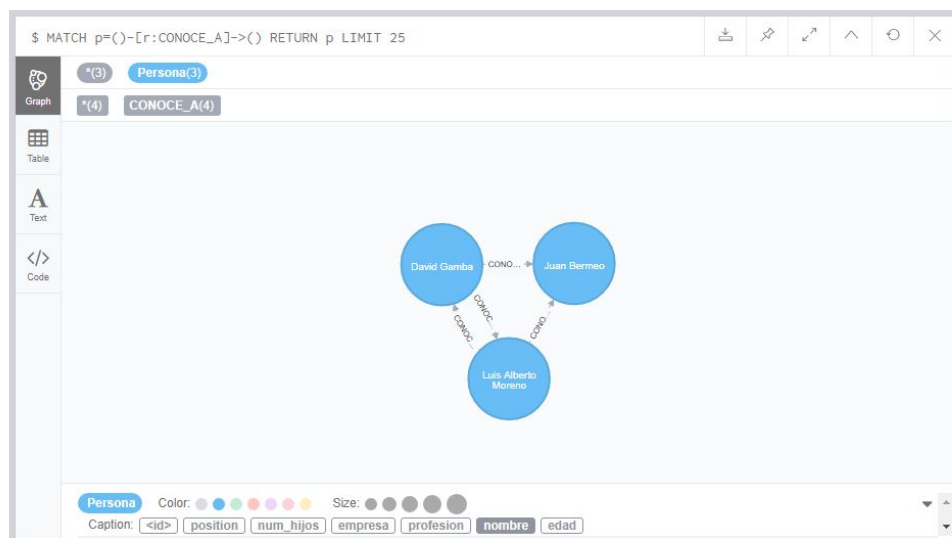
Esta acción mostrará un resultado similar al siguiente:



- Es posible que los nodos no muestren la información deseada, en este caso el nombre de la persona o que se desee cambiar el color de los nodos o las relaciones. Para modificar el estilo del nodo, presione el label "Persona(3)" que aparece en la parte superior del recuadro



Esta acción desplegará las opciones de edición en la parte inferior del recuadro, allí podrá escoger para el nodo, por ejemplo, el tamaño del nodo, el color y el caption



5. Este grafo está compuesto por nodos de tipo **Persona** y relaciones de tipo **"CONOCE_A"**. Los nodos los creamos de la siguiente forma:

```
CREATE (:Persona {nombre:"Luis Alberto Moreno", edad:24})
```

Esta sintaxis permite definir el nodo de tipo **Persona** con sus propiedades, en nuestro modelo hemos creado unas propiedades adicionales y lo hemos creado con la siguiente sentencia:

```
CREATE  
(:Persona  
  {  
    nombre:"Luis Alberto Moreno",  
    edad:24,  
    empresa:"BDIHub",  
    position:"Big Data Consultant",  
    profesion:"Ingeniero de sistemas y computación",  
    num_hijos:0  
  }  
)
```

6. Para esta actividad cree un nodo en Neo4j con sus datos.

7. Las relaciones las creamos usando la siguiente sentencia:

```
MATCH (a:Persona), (b:Persona)  
WHERE a.nombre = "Luis Alberto Moreno"  
AND b.nombre = "Juan Bermeo"  
CREATE (a)-[:CONOCE_A]->(b);
```

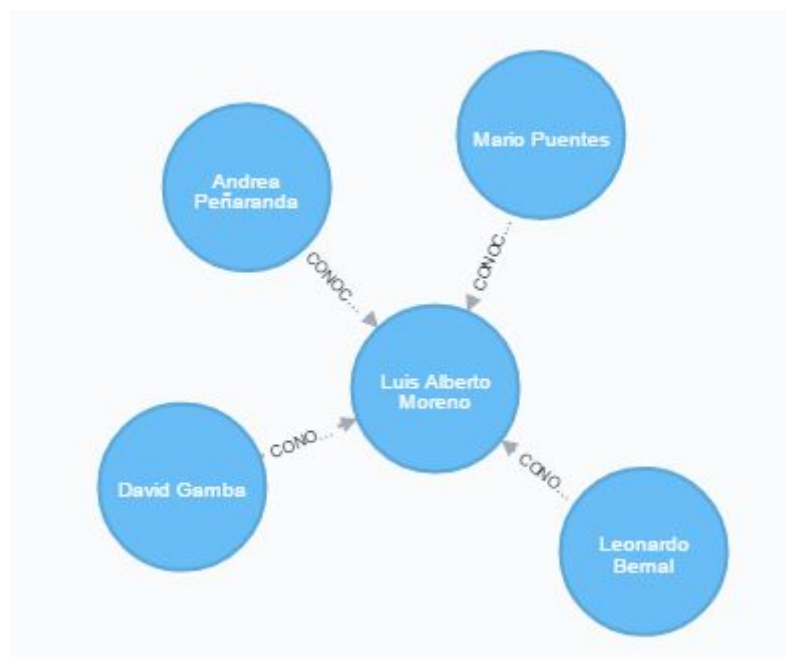
En esta sentencia las primeras tres líneas corresponden a localizar los nodos que queremos conectar y la última establece la relación entre ambos. Para esto definimos los dos nodos queremos, que vamos a nombrar **a** y **b** y luego especificamos los nombres de cada persona en las propiedad "nombre" de cada nodo, luego de tener los nodos identificados creamos la relación indicando la dirección en la cual podemos ver que **a** (Luis Alberto Moreno), conoce a **b** (Juan Bermeo).

8. Cree una relación en la base de datos indicando que usted conoce a Luis Alberto Moreno, si conoce a alguien más del grupo, cree también la relación.

9. Consulte todas las personas que conocen a "Luis Alberto Moreno"

```
MATCH p = (a:Persona)-[r:CONOCE_A]->(b:Persona)  
WHERE b.nombre = "Luis Alberto Moreno"  
RETURN p
```

En esta consulta, la primera línea establece la estructura de la relación, luego establecemos el filtro que nos permite conocer cuáles personas conocen a Luis, por último retornamos el objeto que ha recopilado toda la información de las relaciones, como resultado obtendremos un resultado similar al siguiente:



10. El panel de resultados también nos ofrece otras opciones de visualización como tabla o texto

\$ MATCH p = (a:Persona)-[r:CONOCE_A]->(b:Persona) WHERE b.nombre = "Luis Alberto Mo...

Graph
Table
Text
Code

```

{
  "p": [
    {
      "position": "Big Data Consultant",
      "num_hijos": 1,
      "profesion": "Ingeniero de sistemas",
      "empresa": "BDIHub",
      "nombre": "Leonardo Bernal",
      "edad": 46,
      "relationship": {
        "position": "Big Data Consultant",
        "num_hijos": 0,
        "empresa": "BDIHub",
        "profesion": "Ingeniero de sistemas y computaci\u00f3n",
        "nombre": "Luis Alberto Moreno",
        "edad": 24
      }
    },
    {
      "position": "Data Science Consultant",
      "num_hijos": 0,
      "profesion": "F\u00edsico",
      "empresa": "BDIHub",
      "edad": 26,
      "nombre": "Mario Puentes",
      "relationship": {
        "position": "Big Data Consultant",
        "num_hijos": 0,
        "empresa": "BDIHub",
        "profesion": "Ingeniero de sistemas y computaci\u00f3n",
        "nombre": "Luis Alberto Moreno",
        "edad": 24
      }
    },
    {
      "position": "Big Data Consultant",
      "num_hijos": 0,
      "profesion": "Ingeniero de sistemas",
      "empresa": "BDIHub",
      "nombre": "Andrea Pe\u00f1aranda",
      "edad": 33,
      "relationship": {
        "position": "Big Data Consultant",
        "num_hijos": 0,
        "empresa": "BDIHub",
        "profesion": "Ingeniero de sistemas y computaci\u00f3n",
        "nombre": "Luis Alberto Moreno",
        "edad": 24
      }
    },
    {
      "position": "Data Science Consultant",
      "num_hijos": 0,
      "profesion": "F\u00edsico",
      "empresa": "BDIHub",
      "edad": 24,
      "nombre": "David Gamba",
      "relationship": {
        "position": "Big Data Consultant",
        "num_hijos": 0,
        "empresa": "BDIHub",
        "profesion": "Ingeniero de sistemas y computaci\u00f3n",
        "nombre": "Luis Alberto Moreno",
        "edad": 24
      }
    }
  ]
}
  
```

MAX COLUMN WIDTH: [Control deslizante]

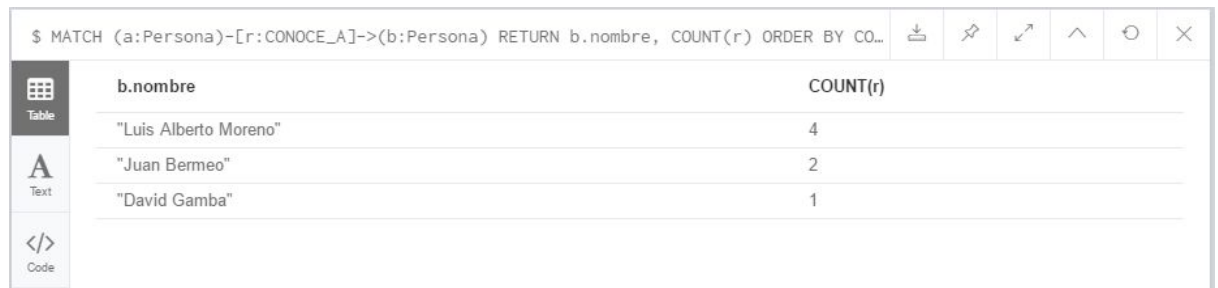
11. Consulte el n\u00famero de personas que conocen a cada uno de los integrantes registrados en la base de datos

```

MATCH (a:Persona)-[r:CONOCE_A]->(b:Persona)
RETURN b.nombre, COUNT(r)
ORDER BY COUNT(r) DESC

```

El resultado de la consulta será similar al siguiente:



b.nombre	COUNT(r)
"Luis Alberto Moreno"	4
"Juan Bermeo"	2
"David Gamba"	1

En esta consulta estamos usando la función COUNT() para contar el número de relaciones entrantes a un nodo, de cierta forma estamos buscando quiénes son las personas más conocidas en nuestro grafo.

12. Consulte el número de hijos en total de todas las personas registradas en el grafo.

```

MATCH (a:Persona)
RETURN SUM(a.num_hijos) as TOTAL

```

En esta consulta estamos buscando todos los nodos de tipo PERSONA y luego estamos sumando el número de hijos y retornando este valor, cuando se trabaja en grafos sencillos como este, en el que sólo existe un tipo de nodo y un tipo de relación, no es necesario especificar los tipos de nodo o de relación, la consulta podría realizarse de la siguiente forma también:

```

MATCH (a)
RETURN SUM(a.num_hijos) as TOTAL

```