

Parte 1: Neo4j Introduction 2022

Análisis, Diseño y Procesamiento de Datos Aplicados a las Ciencias y a
las Tecnologías(ADP)

Máster Universitario en Inteligencia Computacional e Internet de las Cosas

Universidad de Córdoba, EPSC
2022/2023



Autor:

Antonio Gómez Giménez (i72gogia@uco.es)

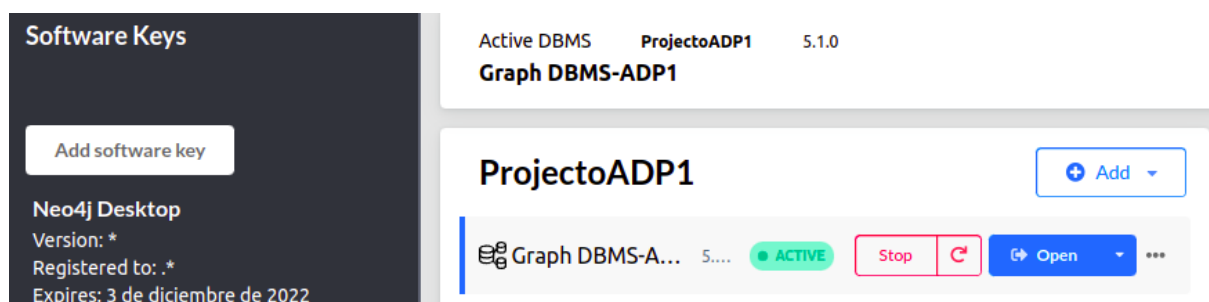
Índice:

1. ¿Qué es Neo4j?	2
2. Primeros Comandos Cypher	3
3. Datos de Grafo	4
3.1. Apartado 1	4
3.2. Apartado 2	4
3.3. Apartado 3	5
3.4. Apartado 4	5
3.5. Apartado 5	6
4. Cypher Hello World	7
5. Scripts	9
6. Movie Graph	10
7. Database Information	13
8. Títulos de Películas	14
9. Película Favorita	16
10. Tom Hanks	17
11. Matrix	18

1. ¿Qué es Neo4j?

Neo4j es un software libre de Base de datos de estructura NoSql basada en grafos, que pretende dar soluciones para aquellos problemas en los que las bases de datos relacionales no son completamente eficientes. Un ejemplo de uso sería la creación de una base de datos en Neo4j para solucionar el almacenamiento de las interacciones entre los usuarios de una red social.

Se pretende utilizar una instalación local para su uso. Se añade la siguiente imagen donde se puede apreciar la clave con la que se registró para su uso en local.

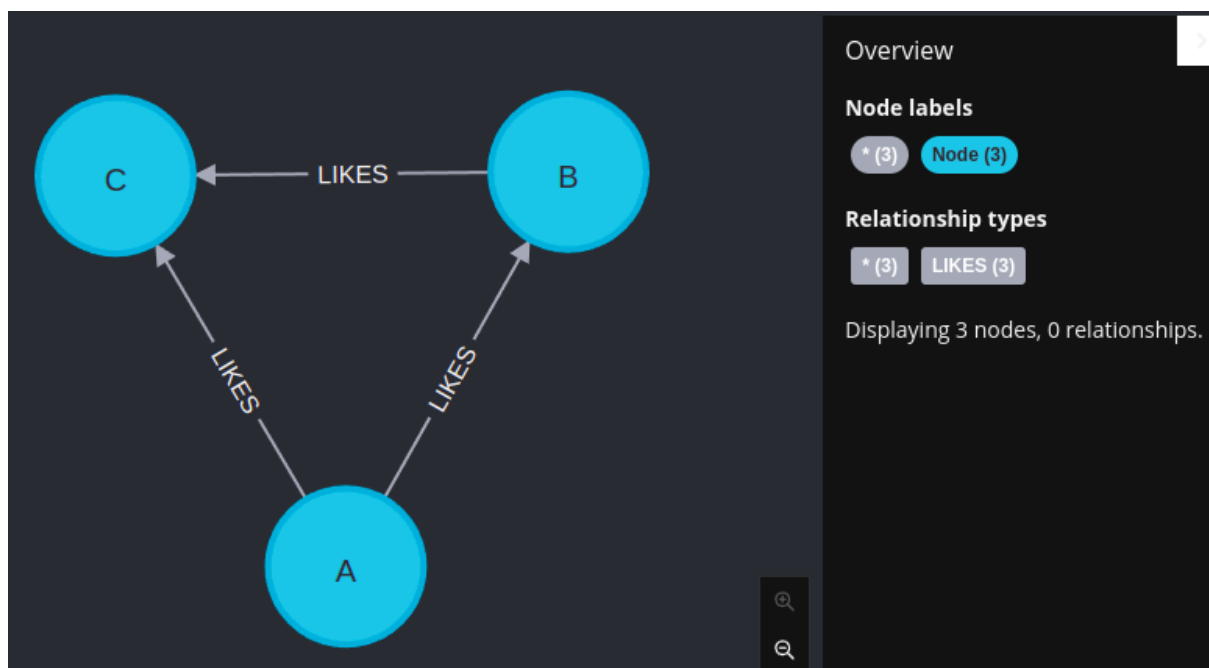


2. Primeros Comandos Cypher

Comando que combina la creación de nodos con la creación de relaciones:

```
CREATE  
(A:Node {name:"A"}),  
(B:Node {name:"B"}),  
(C:Node {name:"C"}),  
(A)-[:LIKES]->(B)-[:LIKES]->(C)-[:LIKES]-(A)
```

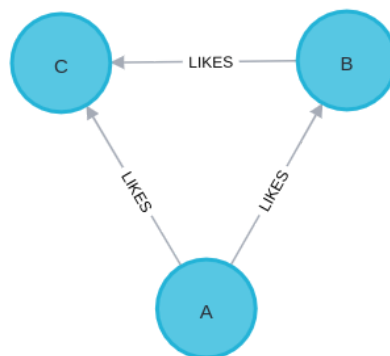
Con el comando match podemos ver el resultado obtenido, siendo el siguiente:



3. Datos de Grafo

3.1. Apartado 1

Creación de una imagen con el grafo completo que realizado en el ejercicio anterior. El comando utilizado sería el descrito en el apartado anterior (tanto la creación del grafo como mostrar el mismo). Se adjunta la imagen descargada de dicho grafo.



3.2. Apartado 2

Mostrar la tabla de JSON con todos los nodos de dicho grafo. Para ello, en vez de ver el resultado en modo gráfica, es necesario poner el modo tabla. Como con el comando match se extrajeron todos los nodos de la base de datos, no es necesario introducir ningún nuevo comando. El resultado es el siguiente:

```
neo4j$ match(n) return (n)
```

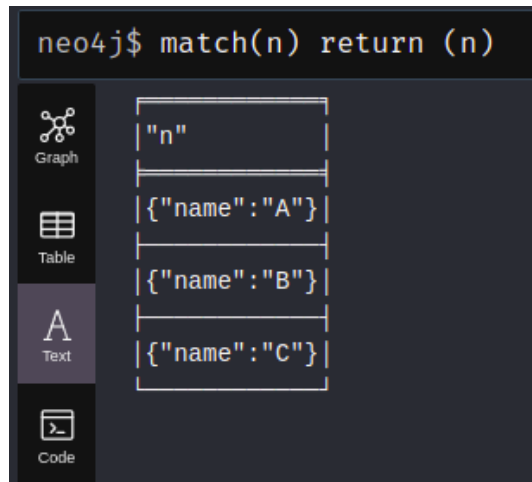
	n
1	<pre>{ "identity": 0, "labels": ["Node"], "properties": { "name": "A" } }</pre>
2	<pre>{ "identity": 1, "labels": ["Node"], "properties": {</pre>

Started streaming 3 records after 7 ms and completed after 30 ms.

La imagen es mucho más extensa, por simplificar se adjunta hasta el segundo nodo.

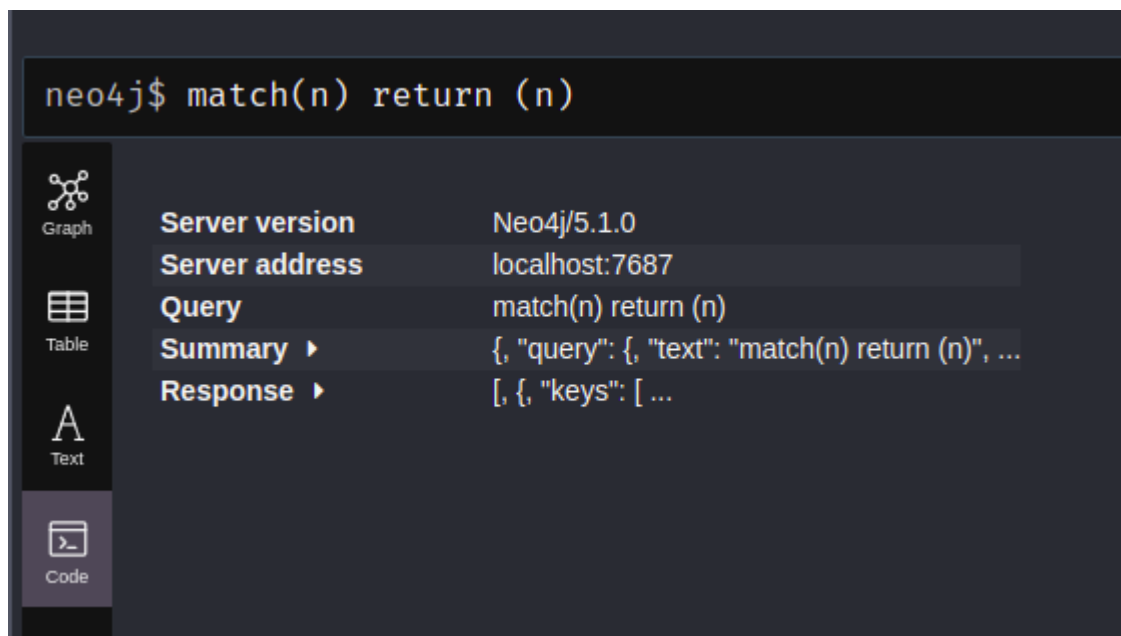
3.3. Apartado 3

En este apartado se pretende mostrar la tabla en modo texto con todos los nodos de dicho grafo. El proceso es similar al apartado anterior pero eligiendo el modo text. La imagen es la siguiente:



3.4. Apartado 4

En este apartado, se pretende mostrar el código enviado y recibido por el servidor y los datos del servidor: server version, server address, query y Summary. Similar al apartado anterior pero eligiendo code. La imagen es la siguiente:




Tanto response cómo summary contienen más información, para mayor claridad se muestra resumido.

3.5. Apartado 5

En este último apartado, se pide borrar todos los nodos y relaciones del grafo. Para ello es necesario escribir el siguiente comando:

```
match(n) detach delete(n)
```

Una vez introducido el comando, el resultado será el siguiente:

A screenshot of a Neo4j Cypher shell interface. The command 'neo4j\$ match(n) detach delete(n)' is entered in the top bar. Below the command bar, there is a sidebar with two options: 'Table' (selected) and 'Code'. The main area displays the result of the command: 'Deleted 3 nodes, deleted 3 relationships, completed after 8 ms.'

```
neo4j$ match(n) detach delete(n)
```

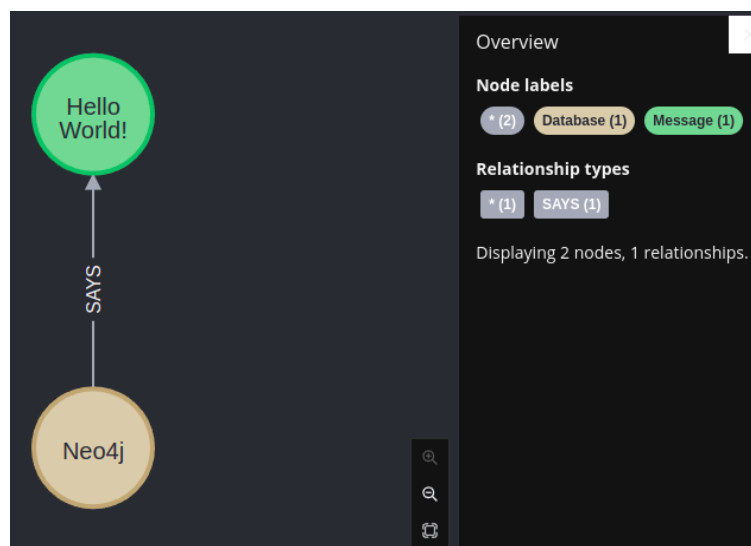
Deleted 3 nodes, deleted 3 relationships, completed after 8 ms.

4. Cypher Hello World

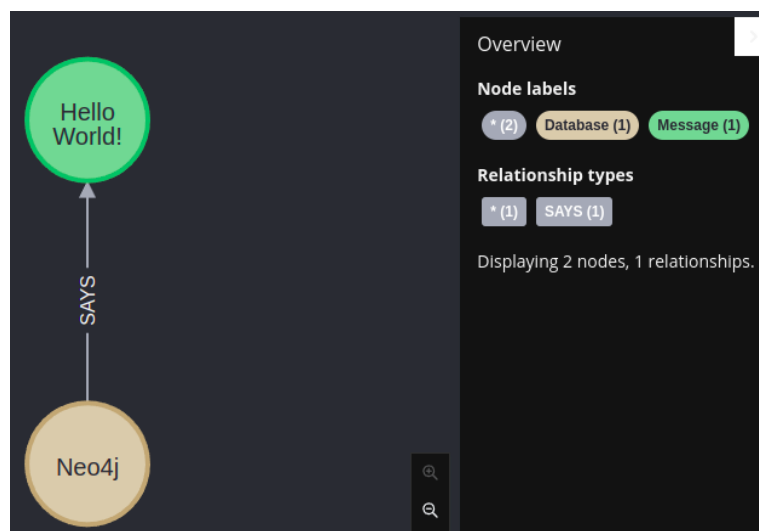
En este apartado se pretende ejecutar el siguiente comando hasta 3 veces, siendo el comando:

```
CREATE (database:Database {name:"Neo4j"})-[r:SAYS]->(message:Message {name:"Hello World!"})  
RETURN database, message, r
```

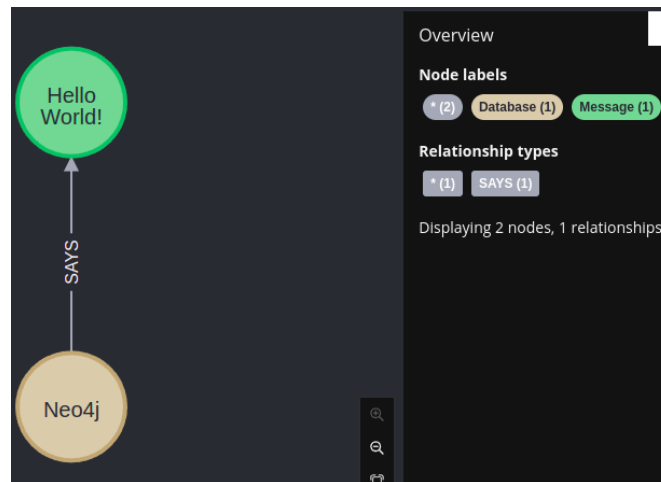
Primera ejecución:



Segunda ejecución:



Tercera ejecución:



Con el comando anterior, estamos creando dos nodos.

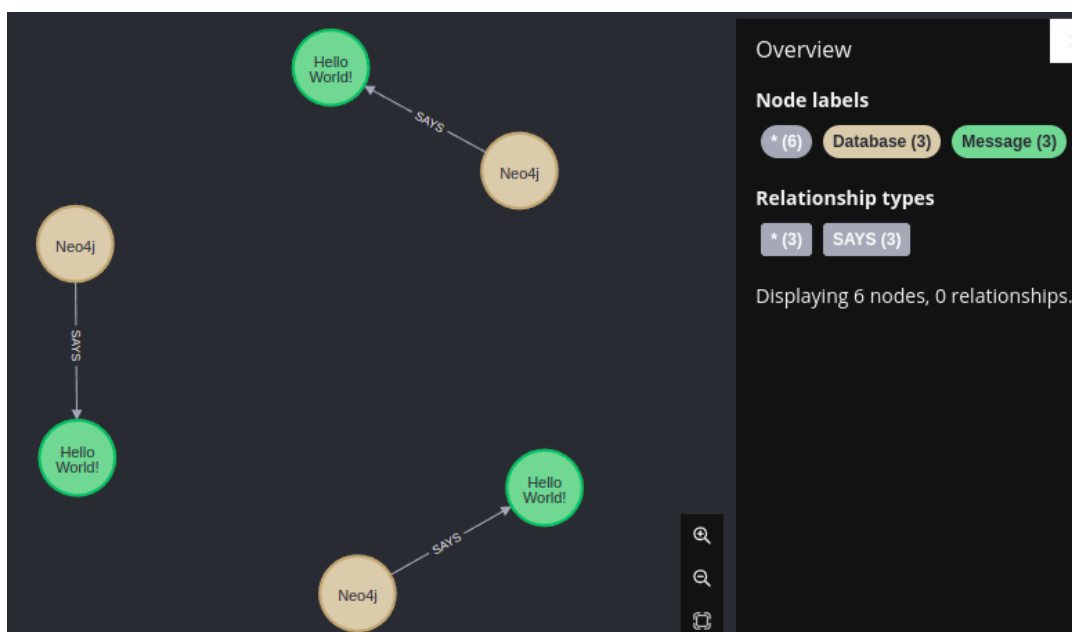
El primer nodo es el nodo Database, el cual, tiene la propiedad name (siendo Neo4j).

El segundo nodo es el nodo Message, el cual, tiene la propiedad name (siendo Hello World!)

Además entre ellos se crea la relación SAYS, que no tiene ninguna propiedad.

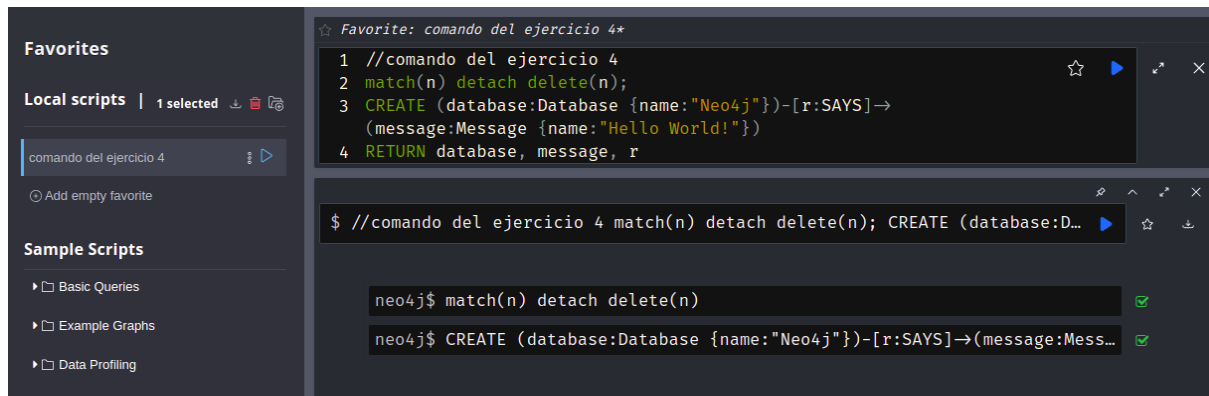
Tanto a los nodos como la relación anterior, están representados con variables para que finalmente al realizar el return se puedan visualizar.

Cada vez que realizamos una ejecución estamos creando dos nodos nuevos con su relación. Esto se puede ver más claro si se realiza un match de toda la base de datos. Se puede apreciar en la siguiente imagen:



5. Scripts

Apartado donde se crea un Script con el código del apartado 4. En la siguiente imagen se puede apreciar el script guardado.



6. Movie Graph

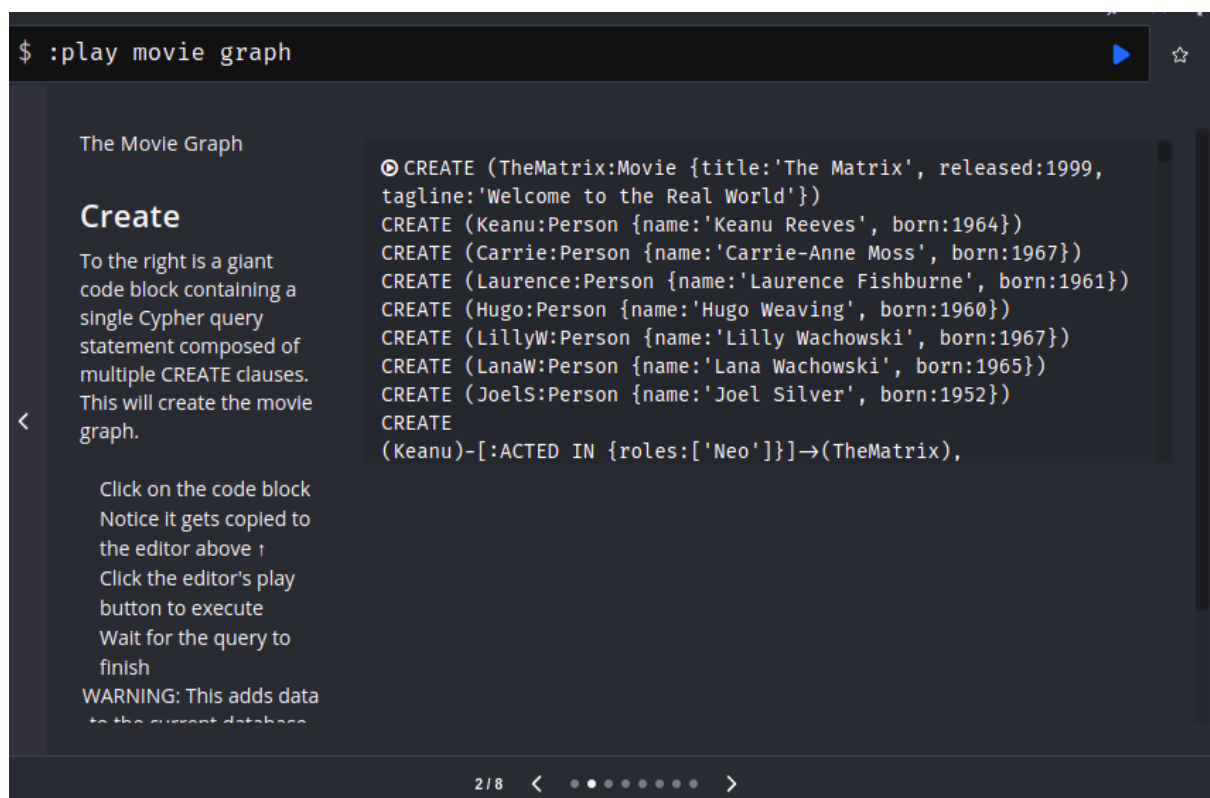
En este ejercicio se pide cargar la base de datos de Movie Graph. Para ello primero es necesario eliminar los nodos y relaciones anteriormente creados con el siguiente comando:

```
match(n) detach delete(n)
```

Posteriormente usamos el comando que activa el tutorial donde se encuentra la base de datos, siendo el comando:

```
:play movie graph
```

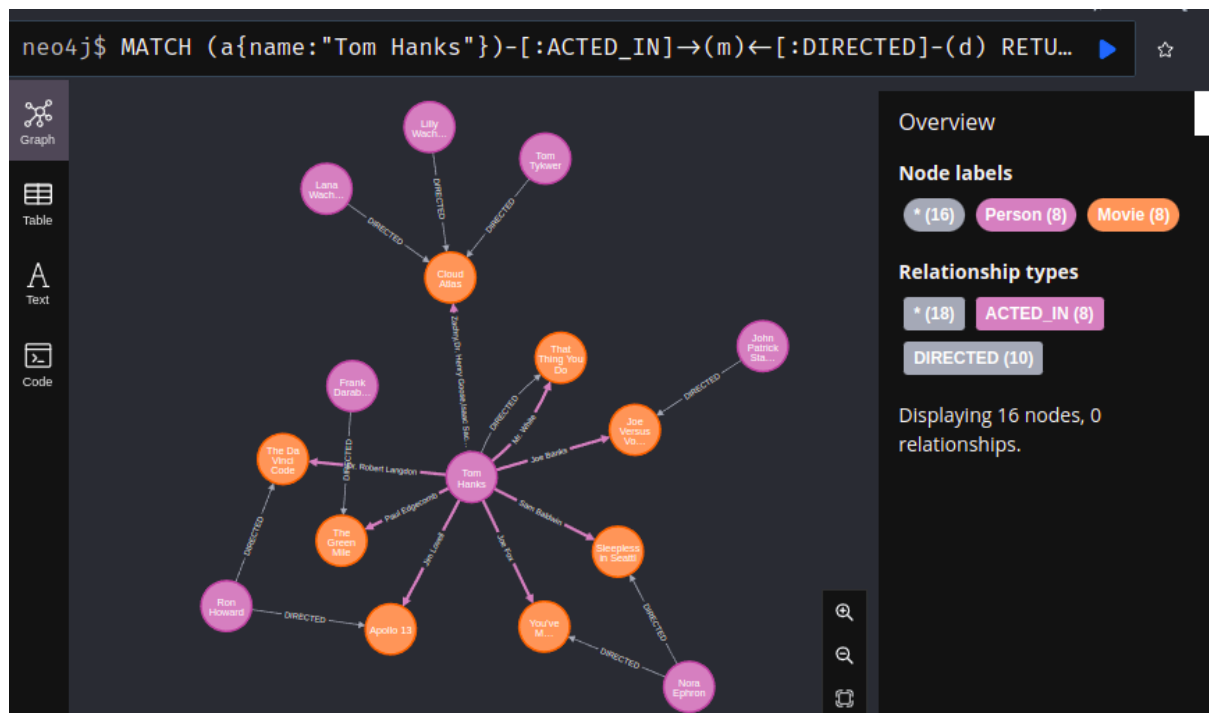
En la segunda pestaña que nos proporciona, ejecutamos el comando que inicia la base de datos, siendo el siguiente:



Ahora ya si podemos probar el comando que nos proporciona el ejercicio, siendo el siguiente:

```
MATCH (a{name:"Tom Hanks"})-[:ACTED_IN]->(m)<-[:DIRECTED]-(d)
RETURN a,m,d LIMIT 10
```

El resultado obtenido es el siguiente:



Como podemos ver, esta instrucción realiza un match donde todos los nodos con nombre Tom Hanks que hayan actuado en una película donde exista un nodo que sea el director de la misma. Devuelve las 10 primeras coincidencias devolviendo el nodo actor, el nodo director y el nodo película.

Si ejecutamos el siguiente comando:

`MATCH (p:Person) RETURN p.name LIMIT 5`

```
neo4j$ MATCH (p:Person) RETURN p.name LIMIT 5
```

	p.name
1	"Milos Forman"
2	"Diane Keaton"
3	"Nancy Meyers"
4	"Chris Columbus"
5	"Julia Roberts"

Se devuelve una tabla con 5 nodos, dando únicamente los nombres de dichos nodos.

Si se quisiera probar el mismo comando pero en vez de devolver solo 5 nodos fueran 10, sería necesario cambiar el LIMIT de comando de 5 a 10. El resultado sería el siguiente:

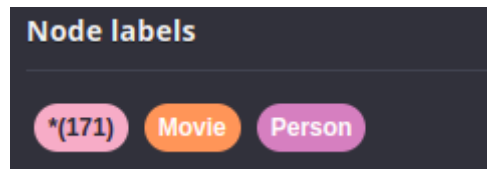
```
neo4j$ MATCH (p:Person) RETURN p.name LIMIT 10
```

	p.name
2	"Diane Keaton"
3	"Nancy Meyers"
4	"Chris Columbus"
5	"Julia Roberts"
6	"Madonna"
7	"Geena Davis"

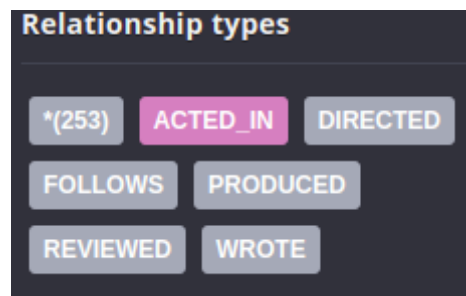
7. Database Information

En este apartado se pretende comprender cómo obtener información más general de la base de datos, por ello se realizan las siguientes preguntas:

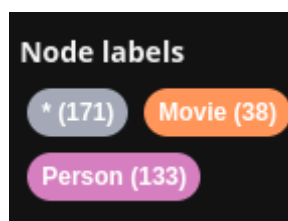
- Cuántos nodos tiene el grafo.



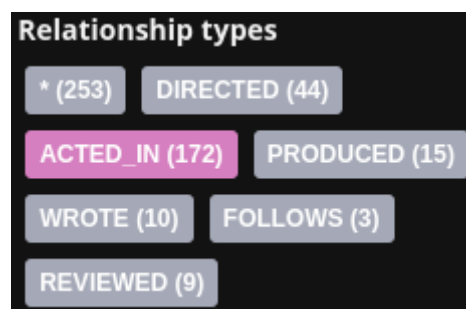
- Cuántas relaciones.



- Pulsa sobre "Node Label" y comprueba el resultado en el panel de resultados.



- Pulsa sobre "Relationship Type" y comprueba el resultado en el panel de resultados.

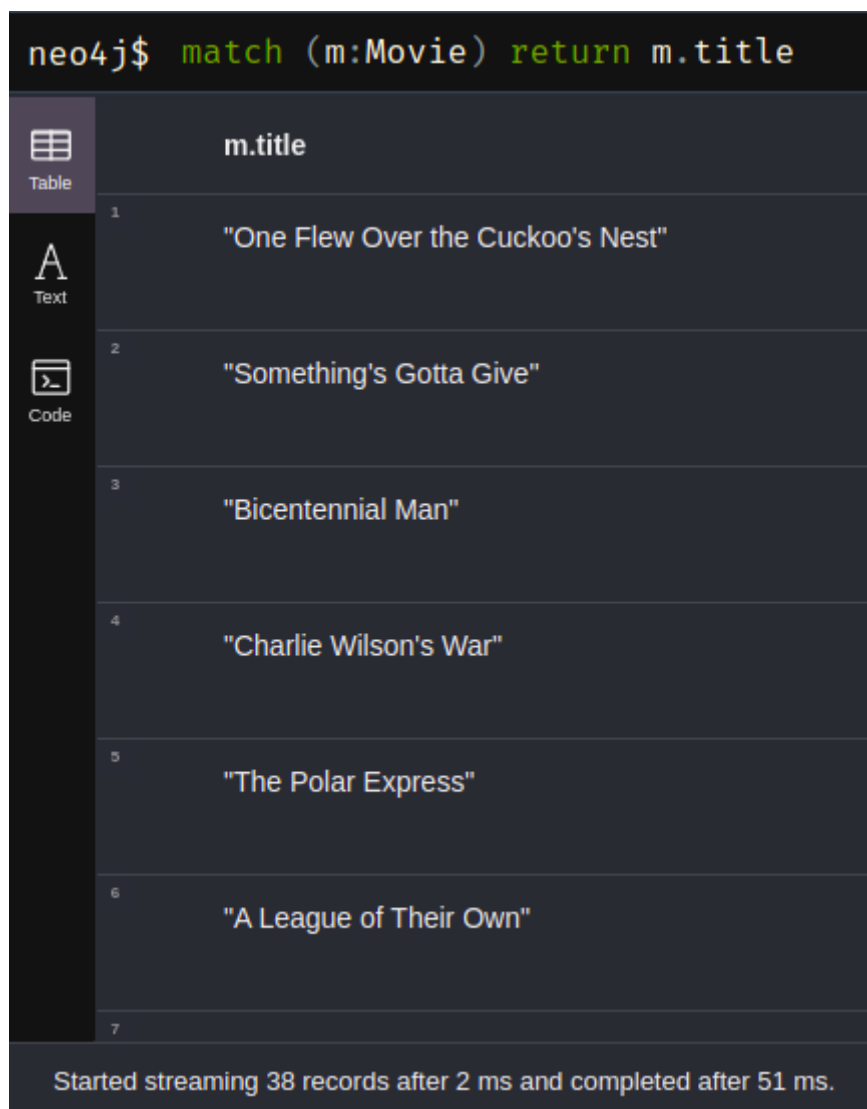


8. Títulos de Películas

En este apartado se pide tanto el Json como el formato tabla para obtener el nombre de todas las películas. Para ello se utilizó el siguiente comando para lograrlo:

```
match (m:Movie) return m.title
```

Los resultados obtenidos para json como tabla son los siguientes:



The screenshot shows the Neo4j Cypher query interface. At the top, the query `neo4j$ match (m:Movie) return m.title` is entered. Below the query, a sidebar on the left contains three icons: a table icon labeled 'Table', a text icon labeled 'Text', and a code icon labeled 'Code'. The 'Table' icon is selected. The main area displays the results of the query in a table format. The table has a single column labeled 'm.title'. The results are listed in rows, numbered 1 through 7. The titles are: "One Flew Over the Cuckoo's Nest", "Something's Gotta Give", "Bicentennial Man", "Charlie Wilson's War", "The Polar Express", and "A League of Their Own". At the bottom of the table, a status bar indicates: "Started streaming 38 records after 2 ms and completed after 51 ms."

	m.title
1	"One Flew Over the Cuckoo's Nest"
2	"Something's Gotta Give"
3	"Bicentennial Man"
4	"Charlie Wilson's War"
5	"The Polar Express"
6	"A League of Their Own"
7	

Started streaming 38 records after 2 ms and completed after 51 ms.

```
neo4j$ match (m:Movie) return m.title
```



Table



Text



Code

"m.title"

"One Flew Over the Cuckoo's Nest"

"Something's Gotta Give"

"Bicentennial Man"

"Charlie Wilson's War"

"The Polar Express"

"A League of Their Own"

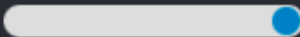
"The Matrix"

"The Matrix Reloaded"

"The Matrix Revolutions"

"The Devil's Advocate"

MAX COLUMN WIDTH:

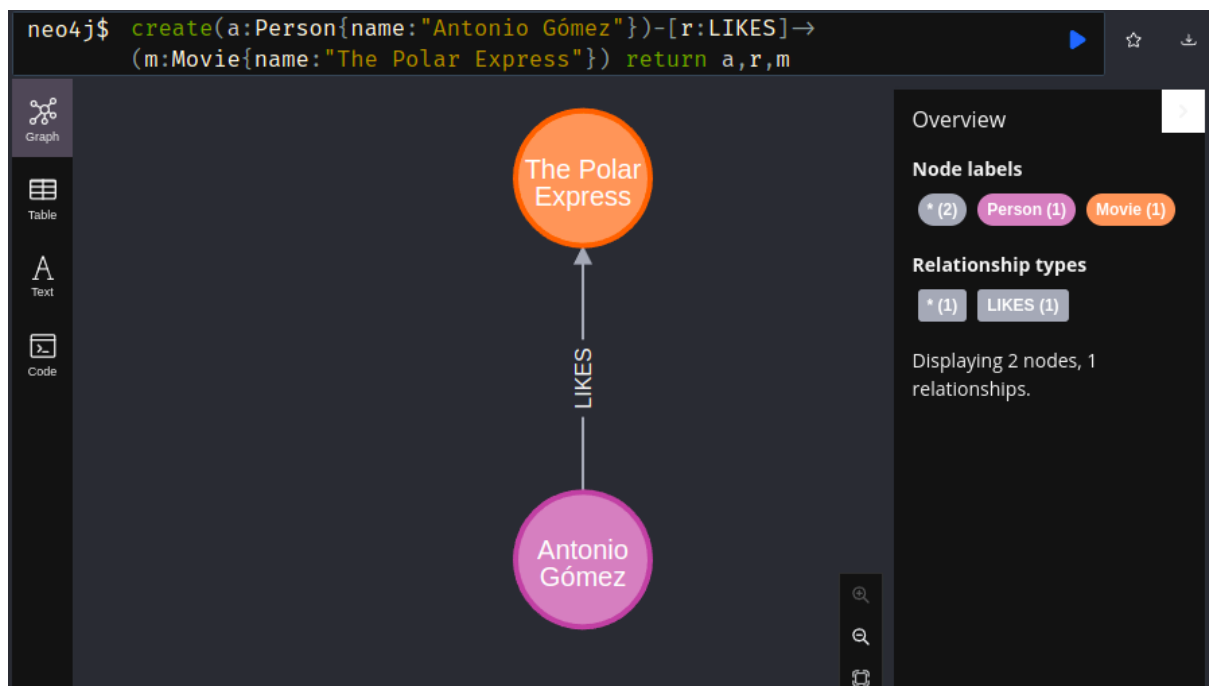


9. Película Favorita

En este apartado se pide añadir un nuevo nodo persona, siendo yo, señalando mi película favorita. En el resultado se debe únicamente mostrar, la película y mi nodo. El comando realizado para llevar a cabo este problema es el siguiente:

```
create(a:Person{name:"Antonio Gómez"})-[r:LIKES]->(m:Movie{name:"The Polar Express"}) return a,r,m
```

La imagen resultante es la siguiente:



10. Tom Hanks

En este ejercicio se pide obtener el título de las películas que ha dirigido o en las que ha actuado Tom Hanks. Para ello se creó el siguiente comando:

```
match (:Person{name:"Tom Hanks"})-[:ACTED_IN|DIRECTED]->(m:Movie) return m.title
```

Este comando da bien los resultados, aun así, neo4j comenta que en futuras versiones no funcionará por estar obsoleto. El resultado es el siguiente:



The screenshot shows the Neo4j Cypher Shell interface. At the top, the command prompt 'neo4j\$' is followed by the Cypher query: `match (:Person{name:"Tom Hanks"})-[:ACTED_IN|DIRECTED]->(m:Movie) return m.title`. Below the query, a sidebar on the left contains icons for 'Table', 'Text', 'Warn', and 'Code'. The 'Table' icon is selected. The main area displays the results of the query as a table with one column, 'm.title'. There are six rows of results, numbered 1 through 6 in the left margin of the table.

	m.title
1	"That Thing You Do"
2	"The Da Vinci Code"
3	"Sleepless in Seattle"
4	"Apollo 13"
5	"A League of Their Own"
6	"You've Got Mail"


11. Matrix

En este apartado se pretende resolver la siguiente pregunta, ¿qué personas han tenido relación con la película 'The Matrix'? Para ello se creó el siguiente comando:

```
match (x:Person)-[]->(:Movie{title:"The Matrix"}) return x.name
```

El comando anterior devuelve todas aquellas personas con algún tipo de relación con la película The Matrix. De esos resultados, escogemos únicamente el nombre.

La imagen resultante es la siguiente:



The screenshot shows a Neo4j Cypher query interface. At the top, the query is entered: `neo4j$ match (x:Person)-[]->(:Movie{title:"The Matrix"}) return x.name`. Below the query, there is a sidebar with three icons: a table icon labeled 'Table', a text icon labeled 'Text', and a code icon labeled 'Code'. The 'Table' icon is selected. The main area displays a table with two columns: an index column and a column labeled 'x.name'. The table contains six rows of results, each with an index and a name in quotes.

	x.name
1	"Emil Eifrem"
2	"Joel Silver"
3	"Lana Wachowski"
4	"Lilly Wachowski"
5	"Hugo Weaving"
6	"Laurence Fishburne"