

GAI- TP4 – Neo4J

Adrián José Zapater Reig

# Ejercicio 1 – Crea una base de datos con los datos que se adjuntan en el archivo grafo.txt disponible en el curso virtual.

El fichero grafo.txt contiene una serie de instrucciones en *Cypher*. Se ha añadido un ‘;’ al final de la primera y última línea para poder ejecutarlo desde la cypher-shell.

Tabla 1 – grafo.txt

|  |
| --- |
| match(n) detach delete(n);  CREATE (Bertoldo:Cliente {nombre:'Bertoldo', edad:33})  CREATE (Herminia:Cliente {nombre:'Herminia', edad:34})  CREATE (Aniceto:Cliente {nombre:'Aniceto', edad:49})  CREATE (Calixta:Cliente {nombre:'Calixta', edad:19})  CREATE (Melibeo:Cliente {nombre:'Melibeo', edad:43})  CREATE (Genara:Cliente {nombre:'Genara', edad:46})  CREATE (Amadea:Cliente {nombre:'Amadea', edad:33})  CREATE (Clotilde:Cliente {nombre:'Clotilde', edad:19})  CREATE (Tiburcio:Cliente {nombre:'Tiburcio', edad:24})  CREATE (Demetrio:Cliente {nombre:'Demetrio', edad:18})  CREATE (Nicodemo:Cliente {nombre:'Nicodemo', edad:51})  CREATE (TV:Producto {id:1,descr:'TV 3D, 200 pulgadas', precio:4000})  CREATE (Maleta:Producto {id:2,descr:'Maleta inteligente, lava la ropa', precio:370})  CREATE (Tostadora:Producto {id:3,descr:'Tostadora 5G, unta la mantequilla sola', precio:230})  CREATE (Clotilde)-[:BAD {id:1}]->(TV),  (Clotilde)-[:BAD {id:2}]->(Maleta),  (Clotilde)-[:BAD {id:3}]->(Tostadora),  (Clotilde)-[:APOYA {id:4}]->(Melibeo),  (Clotilde)-[:APOYA {id:5}]->(Melibeo),  (Bertoldo)-[:BAD {id:6}]->(TV),  (Bertoldo)-[:BAD {id:7}]->(Maleta),  (Nicodemo)-[:BAD {id:8}]->(TV),  (Nicodemo)-[:GOOD {id:9}]->(Maleta),  (Nicodemo)-[:BAD {id:10}]->(Tostadora),  (Demetrio)-[:GOOD {id:11}]->(TV),  (Demetrio)-[:APOYA {id:12}]->(Tiburcio),  (Demetrio)-[:BAD {id:13}]->(Tostadora),  (Tiburcio)-[:GOOD {id:14}]->(TV),  (Tiburcio)-[:GOOD {id:15}]->(Maleta),  (Tiburcio)-[:BAD {id:16}]->(Tostadora),  (Amadea)-[:BAD {id:17}] ->(TV),  (Amadea)-[:APOYA {id:18}] ->(Clotilde),  (Amadea)-[:BAD {id:19}]->(Tostadora),  (Genara)-[:BAD {id:20}]->(TV),  (Genara)-[:APOYA {id:21}]->(Amadea),  (Genara)-[:BAD {id:22}]->(Maleta),  (Genara)-[:BAD {id:23}]->(Tostadora),  (Melibeo)-[:BAD {id:24}]->(TV),  (Melibeo)-[:APOYA {id:25}]->(Genara),  (Melibeo)-[:BAD {id:26}]->(Tostadora),  (Calixta)-[:GOOD {id:27}]->(TV),  (Calixta)-[:APOYA {id:28}]->(Genara),  (Calixta)-[:BAD {id:29}]->(Tostadora),  (Aniceto)-[:GOOD {id:30}]->(TV),  (Aniceto)-[:APOYA {id:31}]->(Genara),  (Aniceto)-[:BAD {id:32}]->(Tostadora),  (Aniceto)-[:GOOD {id:33}]->(Maleta); |

En primer lugar, levantamos la BD en un contenedor de docker. Creamos 3 volúmenes: la carpeta *neo4j-data* para que Neo4J persista la información de la BD, *neo4j-logs* para los logs de ejecución y *neo4j-src* para la carpeta /tmp donde depositaremos el fichero de inicialización.

Tabla 2 – Comando de docker para levantar la BD.

|  |
| --- |
| docker run --name neo4j -d \  --publish=7474:7474 --publish=7687:7687 \  --volume=neo4j-logs:/logs \  --volume=neo4j-data:/data \  --volume=neo4j-tmp:/tmp/src \  neo4j:latest |

Tabla 3 – Comando de copiado

|  |
| --- |
| docker cp src/grafo.txt neo4j:/tmp/src/grafo.txt |

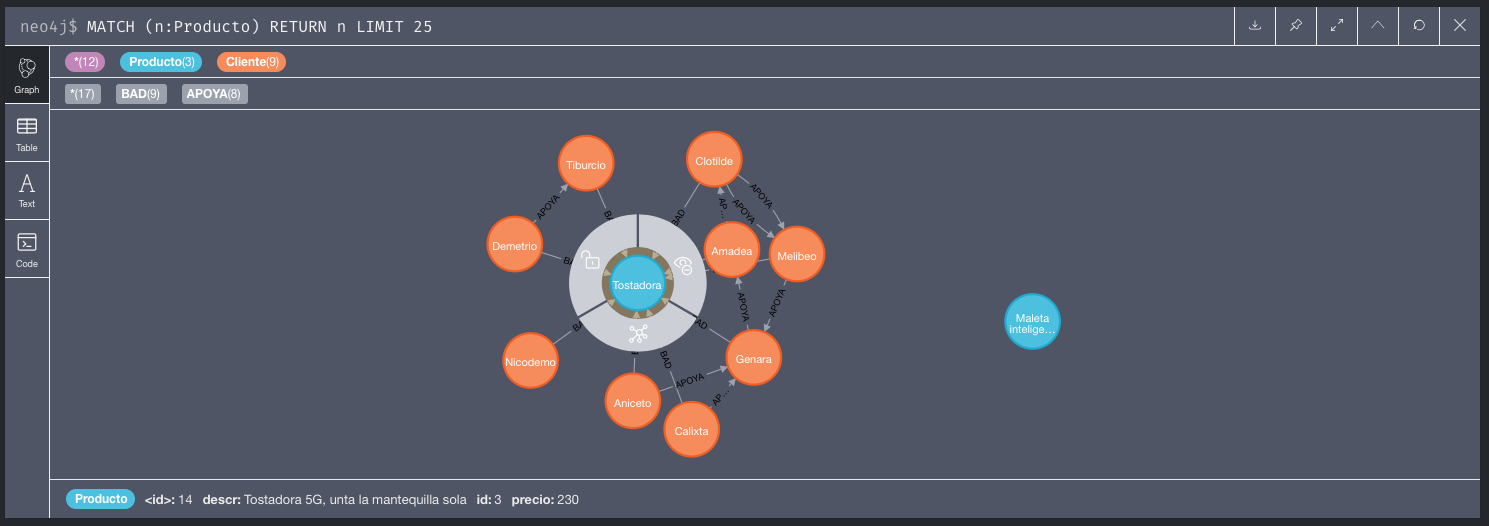
Tabla 4 – Comando de ejecución de fichero grafo.txt

|  |
| --- |
| docker exec -it neo4j cypher-shell -u neo4j -p uned1 -f /tmp/src/grafo.txt |

Tras la ejecución del fichero grafo.txt la cypher-shell nos devuelve:

|  |
| --- |
| 0 rows available after 3 ms, consumed after another 0 ms  Deleted 14 nodes, Deleted 33 relationships  0 rows available after 95 ms, consumed after another 0 ms  Added 14 nodes, Created 33 relationships, Set 64 properties, Added 14 labels |

Y en la herramienta Neo4J browser ([http://localhost:7474](http://localhost:7474/)) se puede comprobar que se han introducido los nodos correctamente:



## Ejercicio 2 - Sobre dicha base de datos, implementa consultas para conseguir la siguiente información:

### 2.1. (2 puntos) Queremos conocer nodos tipo Cliente que hayan emitido una opinión tipo BAD del producto con id=2 (hay 3 clientes con estas características).

### 2.2. (2 puntos) Queremos buscar clientes que formen un “circuito” de relaciones de tipo “APOYA”, es decir una secuencia de relaciones, todas en la misma dirección, que acaba y empieza en el mismo Cliente.

### 2.3. (2 puntos) Demetrio ha decidido mostrar su apoyo a Genara. Añadir la relación APOYA entre los nodos correspondientes. No se debe suponer que el nodo existe (ni que no existe), solo se debe asegurar que tras la instrucción ambos existen (sin duplicarlos) y con la relación APOYA indicada.

# 3. (3 puntos) Crea tu propia base de datos sobre Neo4j, e implementa al menos 2 consultas sobre dichos datos. Se valorará la originalidad y la complejidad de la base de datos y de las consultas creadas.