

## TUTORIAL GRAFOS NEO4J

## 1.- Insertar elementos en un grafo en Neo4J

```
CREATE (Paco:Person {name:'Paco', born:1964})
CREATE (Juan:Person {name:'Juan', born:1967})
CREATE (Andres:Person {name:'Andres', born:1961})
CREATE (Hugo:Person {name:'Hugo', born:1960})
CREATE (Natalia:Person {name:'Natalia', born:1967})
CREATE (Miriam:Person {name:'Miriam', born:1965})
CREATE (Rosa:Person {name:'Rosa', born:1952})
CREATE
  (Telefonica:Company {name:'Telefonica', central_office:'Madrid',
    sector:'telecomunicaciones'}),
  (Repsol:Company {name:'Repsol', central_office:'Madrid',
    sector:'energia'})

CREATE
  (Paco)-[:FRIEND_OF {role:['Amigo de Trabajo']}]>(Juan),
  (Paco)-[:FRIEND_OF {role:['Amigo de Trabajo']}]>(Andres),
  (Juan)-[:FRIEND_OF {role:['Amigo de la infancia']}]>(Hugo),
  (Andres)-[:FRIEND_OF {role:['Amigo de la infancia']}]>
  (Natalia),
  (Miriam)-[:FRIEND_OF {role:['Amigo de Trabajo']}]>(Rosa)
CREATE
  (Paco)-[:WORK_AT {position:['Director de Marketing']}]>
  (Telefonica),
  (Andres)-[:WORK_AT {position:['Director de Marketing']}]>
  (Telefonica),
  (Miriam)-[:WORK_AT {position:['Director de Marketing']}]>
  (Repsol),
  (Rosa)-[:WORK_AT {position:['Director de Marketing']}]>(Repsol)
CREATE
  (Rosa)-[:IS_FAMILY_OF {position:['Prima']}]>(Hugo)
```

**NOTA:** correr todo el código en una sola sentencia.

neo4j\$ CREATE (Paco:Person {name:'Paco', born:1964}) CREATE (Juan:Person {name:'Juan', born:1967}) CREATE (Andres:Person {name:'Andres', born:1961}) CREATE (Hugo:Person {name:'Hugo', born:1960}) CREATE (Natalia:Person {name:'Natalia', born:1967}) CREATE (Miriam:Person {name:'Miriam', born:1965}) CREATE (Rosa:Person {name:'Rosa', born:1952}) CREATE (Telefonica:Company {name:'Telefonica', central\_office:'Madrid', sector:'telecomunicaciones'}) (Repsol:Company {name:'Repsol', central\_office:'Madrid', sector:'energia'}) CREATE (Paco)-[:FRIEND\_OF {role:'Amigo de Trabajo'}]->(Juan), (Paco)-[:FRIEND\_OF {role:'Amigo de Trabajo'}]->(Andres), (Juan)-[:FRIEND\_OF {role:'Amigo de la infancia'}]->(Hugo), (Andres)-[:FRIEND\_OF {role:'Amigo de la infancia'}]->(Natalia), (Miriam)-[:FRIEND\_OF {role:'Amigo de Trabajo'}]->(Rosa) CREATE (Paco)-[:WORK\_AT {position:'Director de Marketing'}]->(Telefonica), (Andres)-[:WORK\_AT {position:'Director de Marketing'}]->(Telefonica), (Miriam)-[:WORK\_AT {position:'Director de Marketing'}]->(Repsol), (Rosa)-[:WORK\_AT {position:'Director de Marketing'}]->(Repsol) CREATE (Rosa)-[:IS\_FAMILY\_OF {position:'Prima'}]->(Hugo)

Server version	Neo4j/4.0.3
Server address	localhost:7687
Query	CREATE (Paco:Person {name:'Paco', born:1964}) CREATE (Juan:Person {name:'Juan', born:1967}) CREATE (Andres:Person {name:'Andres', born:1961}) CREATE (Hugo:Person {name:'Hugo', born:1960}) CREATE (Natalia:Person {name:'Natalia', born:1967}) CREATE (Miriam:Person {name:'Miriam', born:1965}) CREATE (Rosa:Person {name:'Rosa', born:1952}) CREATE (Telefonica:Company {name:'Telefonica', central_office:'Madrid', sector:'telecomunicaciones'}) (Repsol:Company {name:'Repsol', central_office:'Madrid', sector:'energia'}) CREATE (Paco)-[:FRIEND_OF {role:'Amigo de Trabajo'}]->(Juan), (Paco)-[:FRIEND_OF {role:'Amigo de Trabajo'}]->(Andres), (Juan)-[:FRIEND_OF {role:'Amigo de la infancia'}]->(Hugo), (Andres)-[:FRIEND_OF {role:'Amigo de la infancia'}]->(Natalia), (Miriam)-[:FRIEND_OF {role:'Amigo de Trabajo'}]->(Rosa) CREATE (Paco)-[:WORK_AT {position:'Director de Marketing'}]->(Telefonica), (Andres)-[:WORK_AT {position:'Director de Marketing'}]->(Telefonica), (Miriam)-[:WORK_AT {position:'Director de Marketing'}]->(Repsol), (Rosa)-[:WORK_AT {position:'Director de Marketing'}]->(Repsol) CREATE (Rosa)-[:IS_FAMILY_OF {position:'Prima'}]->(Hugo)
Summary	{ "query": { "text": "CREATE (Paco:Person {name:'Paco', born:1964})\nCREATE (Juan:Person {name:'Juan', born:1967})\nCREATE (Andres:Person {name:'Andres', born:1961})\nCREATE (Hugo:Person {name:'Hugo', born:1960})\nCREATE (Natalia:Person {name:'Natalia', born:1967})\nCREATE (Miriam:Person {name:'Miriam', born:1965})\nCREATE (Rosa:Person {name:'Rosa', born:1952})\nCREATE (Telefonica:Company {name:'Telefonica', central_office:'Madrid', sector:'telecomunicaciones'})\n(Repsol:Company {name:'Repsol', central_office:'Madrid', sector:'energia'})\n\nCREATE\n(Paco)-[:FRIEND_OF {role:'Amigo de Trabajo'}]->(Juan)\n(Paco)-[:FRIEND_OF {role:'Amigo de Trabajo'}]->(Andres)\n(Juan)-[:FRIEND_OF {role:'Amigo de la infancia'}]->(Hugo)\n(Andres)-[:FRIEND_OF {role:'Amigo de la infancia'}]->(Natalia)\n(Miriam)-[:FRIEND_OF {role:'Amigo de Trabajo'}]->(Rosa)\n\nCREATE\n(Paco)-[:WORK_AT {position:'Director de Marketing'}]->(Telefonica)\n(Andres)-[:WORK_AT {position:'Director de Marketing'}]->(Telefonica)\n(Miriam)-[:WORK_AT {position:'Director de Marketing'}]->(Repsol)\n(Rosa)-[:WORK_AT {position:'Director de Marketing'}]->(Repsol)\n\nCREATE\n(Rosa)-[:IS_FAMILY_OF {position:'Prima'}]->(Hugo)", ...
Response	[] ...

neo4j\$ match (n) return n

\*(9)

Company(2)

Person(7)

\*(10)

FRIEND\_OF(5)

WORK\_AT(4)

IS\_FAMILY\_OF(1)

# Buscar en un grafo

## Buscar por propiedad de nodo

The image shows a Neo4j interface. At the top, a terminal window displays the command `neo4j$ MATCH (nombre {name: "Paco"}) RETURN nombre`. Below the terminal, a sidebar on the left contains icons for Graph, Table, Text, and Code. The main area shows a graph view with a single red circular node labeled "Paco". Above the graph, a status bar indicates `* (1)` and `Person(1)`.

## Buscar por nodo y relación

The image shows a Neo4j interface. At the top, a terminal window displays the command `neo4j$ MATCH (Paco {name: "Paco"})-[:FRIEND_OF]->(amigos) RETURN amigos`. Below the terminal, a sidebar on the left contains icons for Graph, Table, Text, and Code. The main area shows a graph view with two red circular nodes labeled "Juan" and "Andres". Above the graph, a status bar indicates `* (2)` and `Person(2)`.

## Buscar por nodo y relación

neo4j\$

neo4j\$ MATCH (personas)-[:WORK\_AT]→ (Telefonica {name: "Telefonica"}) RETURN personas

personas.name
"Andres"
"Paco"

## Listar buscando por dos relaciones encadenadas

neo4j\$

neo4j\$ MATCH (Paco {name: 'Paco'})-[:FRIEND\_OF]→()-[:FRIEND\_OF]→(amigos\_de\_amigo) RETURN amigos\_de\_amigo.name

amigos_de_amigo.name
"Natalia"
"Hugo"

## Listado buscando por una relación

neo4j\$

neo4j\$ MATCH (persona)-[:WORK\_AT]→(compania) RETURN persona.name, compania.name

persona.name	compania.name
"Rosa"	"Repsol"
"Miriam"	"Repsol"
"Andres"	"Telefonica"
"Paco"	"Telefonica"

## Buscar con restricciones

neo4j\$

neo4j\$ MATCH (p:Person) WHERE p.born > 1960 RETURN p.name

Table

Text

Code

p.name

"Paco"

"Juan"

"Andres"

"Natalia"

"Miriam"

## Contar elementos de dos relaciones concatenadas

neo4j\$

neo4j\$ MATCH (p1:Person)-[:FRIEND\_OF]→(p2:Person)-[:FRIEND\_OF]→(p3:Person) RETURN p1.name AS Persona, COUNT(p2) + COU...

Table

Text

Persona	Amigos
"Paco"	4

## Búsqueda avanzada en grafos

```
CREATE (Paco:Person {name:'Paco', born:1964}),
      (Juan:Person {name:'Juan', born:1967}),
      (Andres:Person {name:'Andres', born:1961}),
      (Hugo:Person {name:'Hugo', born:1960}),
      (Natalia:Person {name:'Natalia', born:1967}),
      (Miriam:Person {name:'Miriam', born:1965}),
      (Rosa:Person {name:'Rosa', born:1952})

CREATE
  (Telefonica:Company {name:'Telefonica', central_office:'Madrid',
sector:'telecomunicaciones'}),
  (Repsol:Company {name:'Repsol', central_office:'Madrid',
sector:'energia'}),
  (Mercadona:Company {name:'Mercadona', central_office:'Valencia',
sector:'alimentacion'})

CREATE
  (Paco)-[:FRIEND_OF {role:['Amigo de Trabajo']}]>(Juan),
  (Paco)-[:FRIEND_OF {role:['Amigo de Trabajo']}]>(Andres),
  (Juan)-[:FRIEND_OF {role:['Amigo de la infancia']}]>(Hugo),
  (Andres)-[:FRIEND_OF {role:['Amigo de la infancia']}]>(Natalia),
  (Miriam)-[:FRIEND_OF {role:['Amigo de Trabajo']}]>(Rosa),
  (Natalia)-[:FRIEND_OF {role:['Amigo de gimnasio']}]>(Juan),
  (Rosa)-[:FRIEND_OF {role:['Amigo de Trabajo']}]>(Hugo)

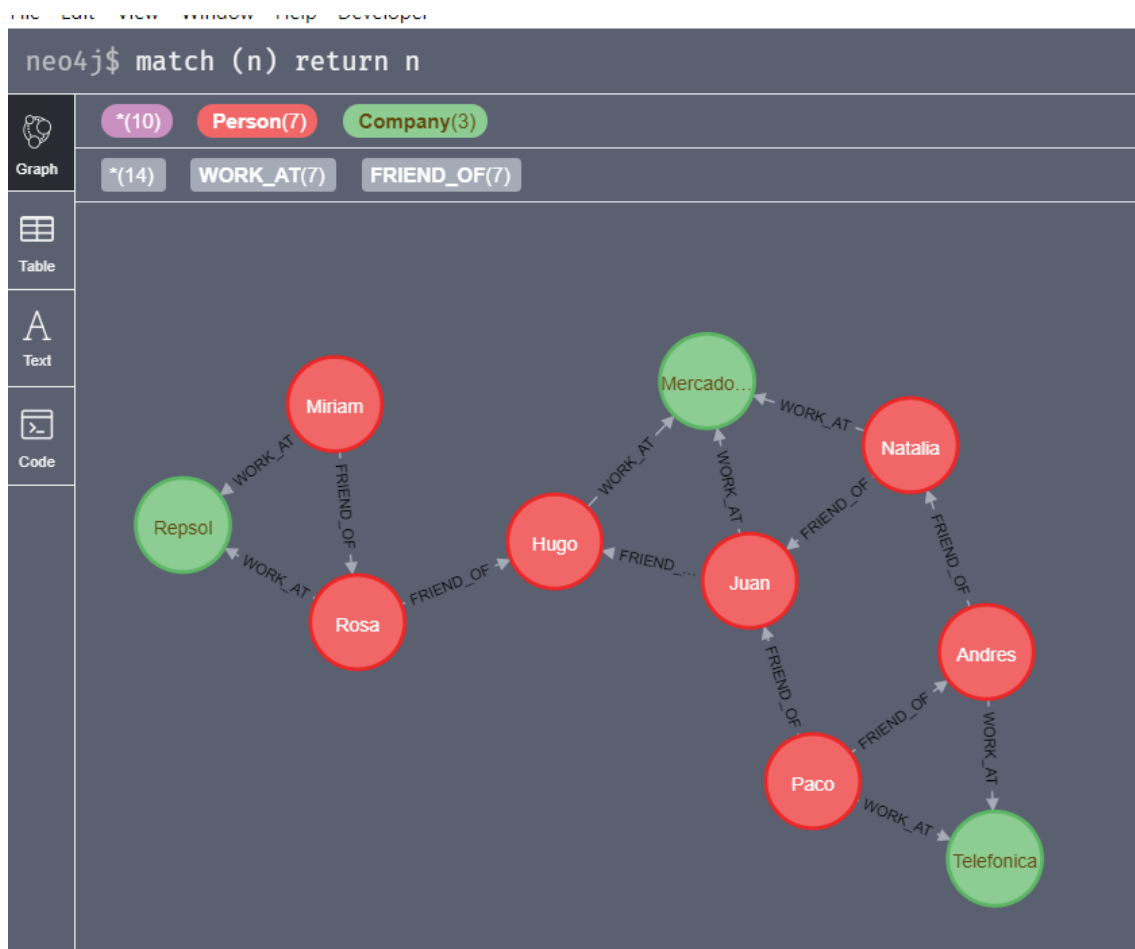
CREATE
  (Paco)-[:WORK_AT {position:['Director de Marketing']}]>
  (Telefonica),
  (Andres)-[:WORK_AT {position:['Director de Marketing']}]>
  (Telefonica),
  (Miriam)-[:WORK_AT {position:['Director de Marketing']}]>
  (Repsol),
  (Rosa)-[:WORK_AT {position:['Director de Marketing']}]>(Repsol),
  (Hugo)-[:WORK_AT {position:['Director de Marketing']}]>
  (Mercadona),
  (Juan)-[:WORK_AT {position:['Director de Marketing']}]>
  (Mercadona),
  (Natalia)-[:WORK_AT {position:['Director de Marketing']}]>
  (Mercadona)
```

## Resultado

```
neo4j$ CREATE (Paco:Person {name:'Paco', born:1964}), (Juan:Person {name:'Juan', bo
```



Added 10 labels, created 10 nodes, set 37 properties, created 14 relationships, completed after 17 ms.



## Contar elementos derivados de dos relaciones

```
MATCH (p1:Person)-[:FRIEND_OF]->(p2:Person)-[:WORK_AT]->(c:Company)
RETURN p1.name AS Persona , COUNT(c.sector) AS Sectores
```

```
neo4j$
```

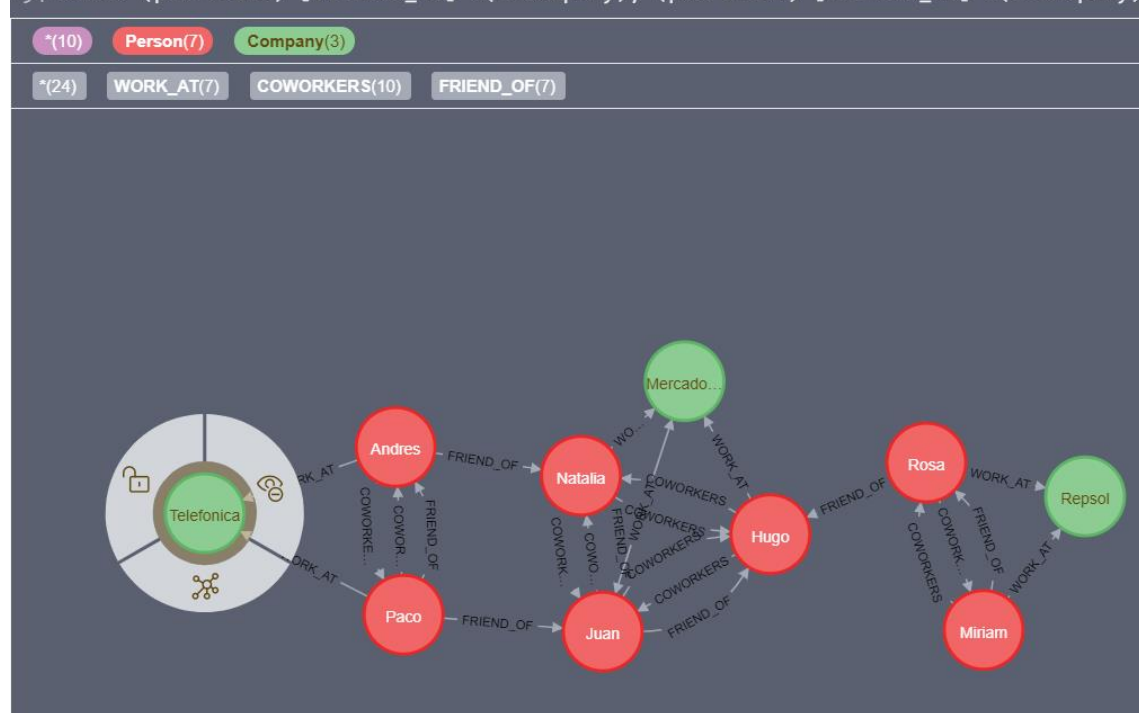
```
neo4j$ MATCH (p1:Person)-[:FRIEND_OF]->(p2:Person)
```

Persona
"Paco"
"Miriam"
"Rosa"
"Juan"
"Natalia"
"Andres"

Started streaming 6 records after 1 ms and completed after 4 ms.

## Generar relaciones a partir de consultas

```
MATCH
  (p1:Person)-[r1:WORK_AT]->(c:Company),
  (p2:Person)-[r2:WORK_AT]->(c:Company)
CREATE (p1)-[r3:COWORKERS]->(p2)
RETURN p1,p2,c,r1,r2,r3
```



## Agrupaciones

```
MATCH (p1:Person)-[:FRIEND_OF]->(p2:Person)-[:WORK_AT]->(c:Company)
RETURN p1.name, COLLECT(c.sector)
```

neo4j\$

neo4j\$ MATCH (p1:Person)-[:FRIEND\_OF]->(p2:Person)-[:WORK\_AT]->(c:Company)

p1.name	COLLECT(c.sector)
"Paco"	["telecomunicaciones", "alimentacion"]
"Miriam"	["energia"]
"Rosa"	["alimentacion"]
"Juan"	["alimentacion"]
"Natalia"	["alimentacion"]
"Andres"	["alimentacion"]

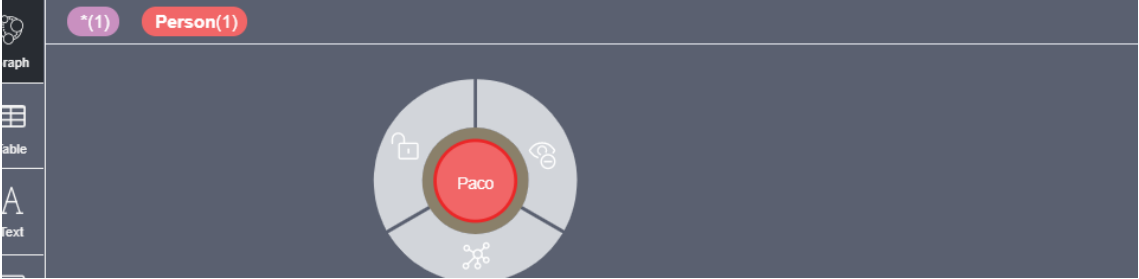
Started streaming 6 records after 1 ms and completed after 12 ms.



## Modificar elementos del grafo

### Modificar propiedades a un nodo

```
neo4j$ MERGE (p:Person {name: 'Paco'}) SET p.age = 34, p.coat = 'Yellow' RETURN p
```



The screenshot shows the Neo4j web interface. At the top, a Cypher query is entered: `MERGE (p:Person {name: 'Paco'}) SET p.age = 34, p.coat = 'Yellow' RETURN p`. Below the query, the results are displayed in a graph view. A single node is shown, labeled 'Paco', with a red background and a yellow border. The node is connected to a central hub by four lines, representing its connections to other nodes in the graph. The interface includes a sidebar on the left with icons for Graph, Table, Text, and Code.

### Modificar propiedades a una relación

```
MERGE (Paco)-[r:FRIEND_OF]->(Juan) SET r.ages = 34 RETURN r
```

```
neo4j$ MERGE (Paco)-[r:FRIEND_OF]->(Juan) SET
```



The screenshot shows the Neo4j web interface. At the top, a Cypher query is entered: `MERGE (Paco)-[r:FRIEND_OF]->(Juan) SET r.ages = 34 RETURN r`. Below the query, the results are displayed in a table view. The table has a single column labeled 'r'. The data is shown in two rows, each containing a JSON object representing the relationship properties. The first row shows `{ "ages": 34, "role": [ "Amigo de gimnasio" ] }`. The second row shows `{ "ages": 34, "role": [ "Amigo de Trabajo" ] }`. The interface includes a sidebar on the left with icons for Table, Text, and Code.

## Borrar elementos del grafo

### Borrar relaciones entre nodos

```
MATCH (Miriam)-[:FRIEND_OF]->(Rosa) DELETE r
```

### Borrar todo el grafo

```
MATCH (n) OPTIONAL MATCH (n)-[r]-() DELETE n,r
```

```
neo4j$ MATCH (n) OPTIONAL MATCH (n)-[r]-() DELETE n,r
```



Table

Deleted 10 nodes, deleted 24 relationships, completed after 4 ms.