

Nombre: Nicolas Augusto Añazco Pereira

Crear nodos con propiedades

```
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'})
```

Crear relaciones entre nodos

```
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)
```

neo4j@bolt://localhost:7687/neo4j - Neo4j Browser

File Edit View Window Help Developer

neo4j\$

neo4j\$ CREATE (Paco:Person {name:'Paco', born:1964}) CREATE (Juan:Person {name:'Juan', born:1967}) CREATE (Andres:Person {na...

Added 9 labels, created 9 nodes, set 30 properties, created 10 relationships, completed after 9 ms.

neo4j\$ match (n) return n

(no changes, no records)

Added 9 labels, created 9 nodes, set 30 properties, created 10 relationships, completed after 9 ms.

neo4j@bolt://localhost:7687/neo4j - Neo4j Browser

File Edit View Window Help Developer

neo4j\$

neo4j\$ match (n) return n

Graph

Person(7) Company(2)

WORK_AT(4) FRIEND_OF(5) IS_FAMILY_OF(1)

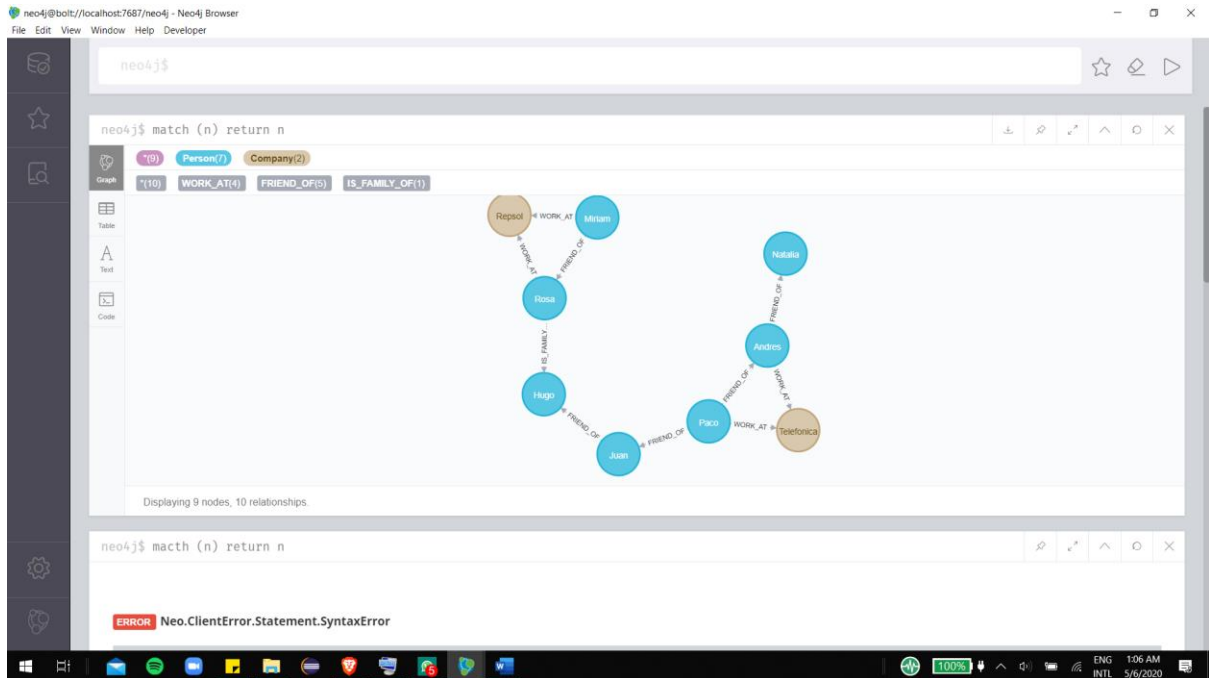
Displaying 9 nodes, 10 relationships.

neo4j\$ match (n) return n

ERROR Neo.ClientError.Statement.SyntaxError

Mostar todo el grafo

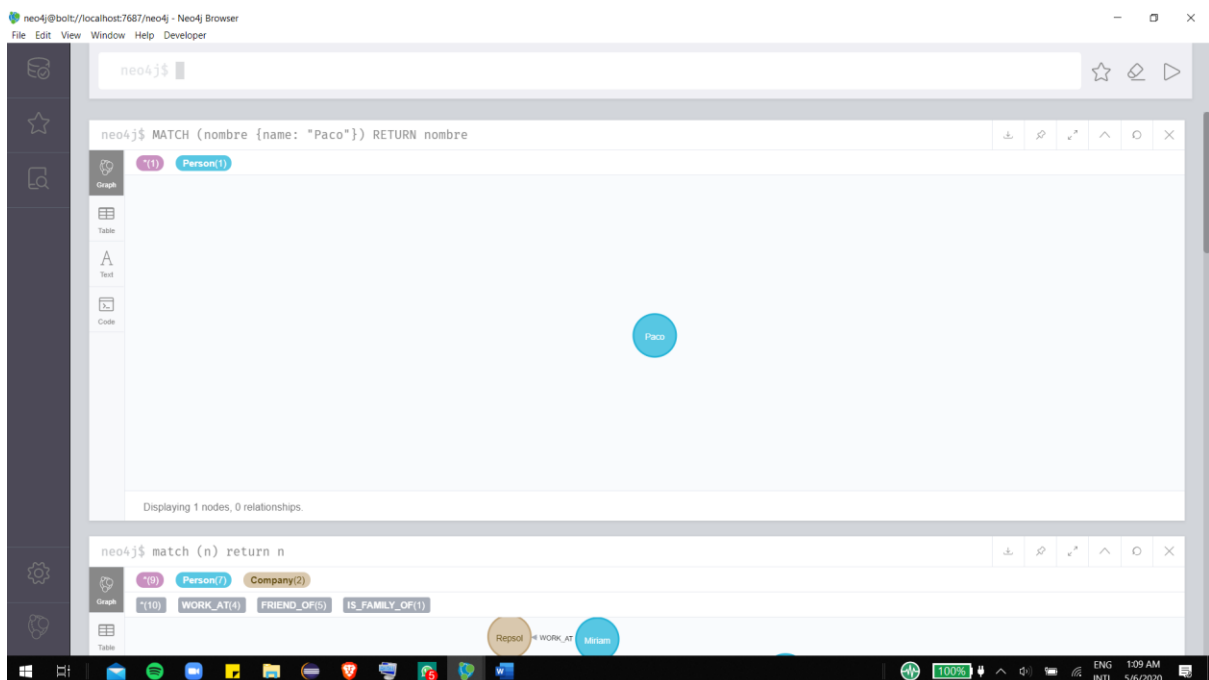
```
MATCH (n) RETURN n
```



Ejemplo 1: Buscar por propiedad de nodo

Buscar a Paco

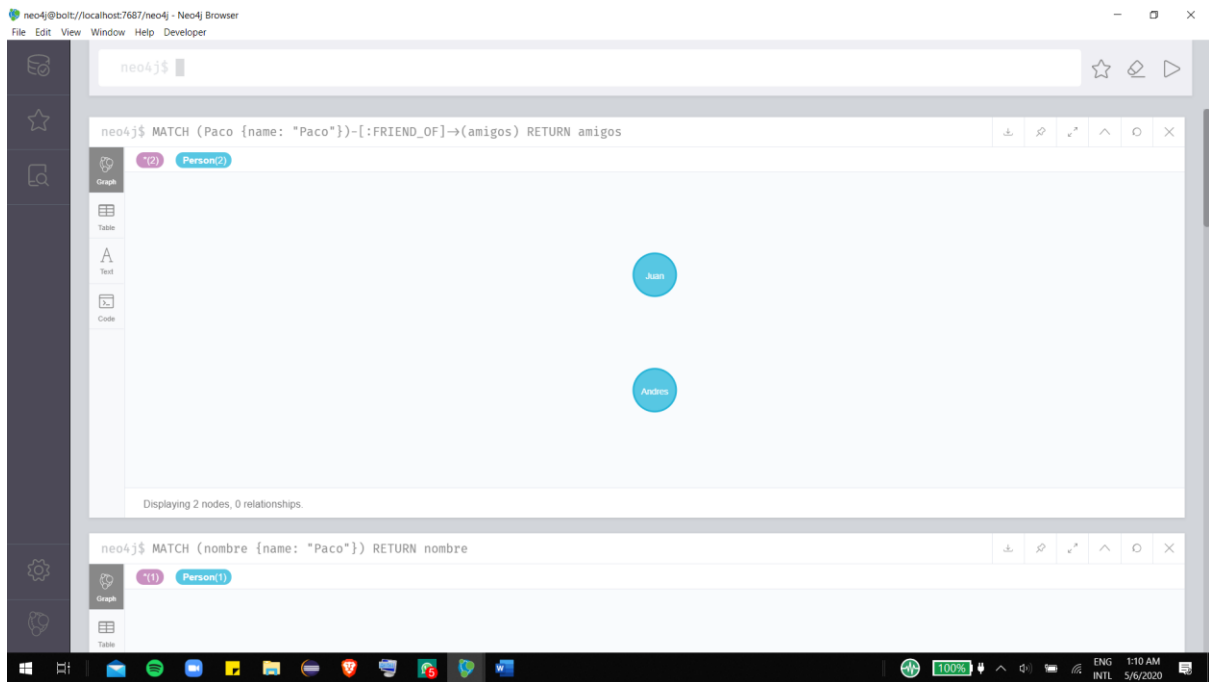
```
MATCH (nombre {name: "Paco"}) RETURN nombre
```



Ejemplo 2: Buscar por nodo y relación

Buscar amigos de Paco

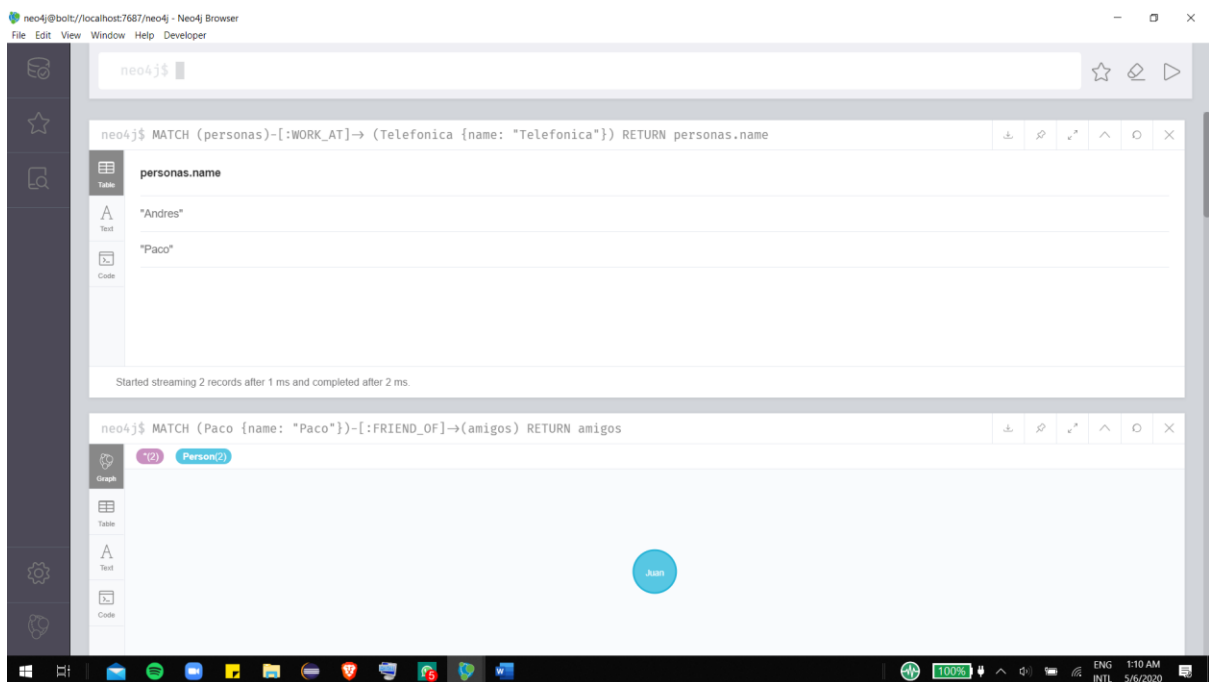
```
MATCH (Paco {name: "Paco"})-[:FRIEND_OF]->(amigos) RETURN amigos
```



Ejemplo 3: Buscar por nodo y relación

Todas las personas que trabajan en Telefonica

```
MATCH (personas)-[:WORK_AT]-> (Telefonica {name: "Telefonica"}) RETURN personas.name
```



Ejemplo 4: Listar buscando por dos relaciones encadenadas

Listado de amigos de los amigos de Paco

```
MATCH (Paco {name: 'Paco'})-[:FRIEND_OF]->()-[:FRIEND_OF]->(amigos_de_amigo) RETURN amigos_de_amigo.name
```

The screenshot shows the Neo4j Browser interface. The top bar indicates the user is logged in as 'neo4j@bolt://localhost:7687/neo4j - Neo4j Browser'. The main query editor contains the Cypher query: `MATCH (Paco {name: 'Paco'})-[:FRIEND_OF]->()-[:FRIEND_OF]->(amigos_de_amigo) RETURN amigos_de_amigo.name`. The results are displayed in a table with the header `amigos_de_amigo.name`. The table contains two rows: `"Natalia"` and `"Hugo"`. Below the table, a status message reads: "Started streaming 2 records in less than 1 ms and completed after 2 ms." The bottom of the screen shows a Windows taskbar with various application icons and system status indicators.

amigos_de_amigo.name
"Natalia"
"Hugo"

Ejemplo 5: Listado buscando por una relación

Listado de personas y que trabajan en compañías

```
MATCH (persona)-[:WORK_AT]->(compania) RETURN persona.name, compania.name
```

The screenshot shows the Neo4j Browser interface. The top bar indicates the user is logged in as 'neo4j@bolt://localhost:7687/neo4j - Neo4j Browser'. The main query editor contains the Cypher query: `MATCH (persona)-[:WORK_AT]->(compania) RETURN persona.name, compania.name`. The results are displayed in a table with two columns: `persona.name` and `compania.name`. The table contains four rows: `"Andres"` and `"Telefonica"`, `"Paco"` and `"Telefonica"`, `"Miriam"` and `"Repsol"`, and `"Rosa"` and `"Repsol"`. Below the table, a status message reads: "Started streaming 4 records after 1 ms and completed after 3 ms." The bottom of the screen shows a Windows taskbar with various application icons and system status indicators.

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

Ejemplo 6: Buscar con restricciones

Listado de personas nacidas después de los 60

```
MATCH (p:Person) WHERE p.born > 1960 RETURN p.name
```

The screenshot shows the Neo4j Browser interface. The query entered is `neo4j$ MATCH (p:Person) WHERE p.born > 1960 RETURN p.name`. The results are displayed in a table with one column, `p.name`, containing the following values: "Paco", "Juan", "Andres", "Natalia", and "Miriam". A status message at the bottom of the results area states: "Started streaming 5 records in less than 1 ms and completed after 7 ms."

p.name
"Paco"
"Juan"
"Andres"
"Natalia"
"Miriam"

Ejemplo 7: Contar elementos de dos relaciones concatenadas

Buscar las personas con más amigos teniendo en cuenta amigos de amigos

```
MATCH (p1:Person)-[:FRIEND_OF]->(p2:Person)-[:FRIEND_OF]->(p3:Person)
RETURN p1.name AS Persona, COUNT(p2) + COUNT(p3) AS Amigos
```

The screenshot shows the Neo4j Browser interface. The query entered is `neo4j$ MATCH (p1:Person)-[:FRIEND_OF]->(p2:Person)-[:FRIEND_OF]->(p3:Person) RETURN p1.name AS Persona, COUNT(p2) + COUNT(p3) AS Amigos`. The results are displayed in a table with two columns: `Persona` and `Amigos`. The first row shows "Paco" as the person and 4 as the number of friends. A status message at the bottom of the results area states: "Started streaming 1 records in less than 1 ms and completed after 2 ms."

Persona	Amigos
"Paco"	4

Below this, the same query as in Example 6 is shown again for reference.

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

p.name
"Paco"
"Juan"
"Andres"
"Natalia"
"Miriam"

Base de datos utilizada

Se define primeramente la base de datos a utilizar en el ejemplo.

```
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)
```

neo4j@bolt://localhost:7687/neo4j - Neo4j Browser

neo4j\$ CREATE (Paco:Person {name:'Paco', born:1964}), (Juan:Person {name:'Juan', born:1967}), (Andres:Person {name:'Andres',...})

Added 10 labels, created 10 nodes, set 37 properties, created 14 relationships, completed after 9 ms.

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

Persona	Amigos
"Paco"	4

Added 10 labels, created 10 nodes, set 37 properties, created 14 relationships, completed after 9 ms.

neo4j@bolt://localhost:7687/neo4j - Neo4j Browser

neo4j\$ match (n) return n

Graph

Person(14) Company(5)

WORK_AT(11) FRIEND_OF(12) IS_FAMILY_OF(1)

Displaying 19 nodes, 24 relationships.

neo4j\$ CREATE (Paco:Person {name:'Paco', born:1964}), (Juan:Person {name:'Juan', born:1967}), (Andres:Person {name:'Andres',...})

Added 10 labels, created 10 nodes, set 37 properties, created 14 relationships, completed after 9 ms.

Ejemplo 1: Contar elementos derivados de dos relaciones

Contar para cada persona el número de amigos que tiene trabajando en los diferentes sectores

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


neo4j@bolt://localhost:7687/neo4j - Neo4j Browser

neo4j\$ MATCH (p1:Person)-[:FRIEND_OF]-(p2:Person)-[:WORK_AT]-(c:Company) RETURN p1.name AS Persona , COUNT(c.sector) AS Sectores

Persona	Sectores
"Paco"	3
"Miriam"	2
"Andres"	1
"Natalia"	1
"Juan"	1
"Rosa"	1

Started streaming 6 records in less than 1 ms and completed after 2 ms.

neo4j\$ match (n) return n

Ejemplo 2: Generar relaciones a partir de consultas

Crear la relación de compañeros de trabajo «coworkers» para las personas que trabajen en la misma compañía.

```
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
```

neo4j@bolt://localhost:7687/neo4j - Neo4j Browser

neo4j\$ 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

neo4j\$ MATCH (p1:Person)-[:FRIEND_OF]-(p2:Person)-[:WORK_AT]-(c:Company) RETURN p1.name AS Persona , COUNT(c.sector) AS Sectores

Persona	Sectores
"Paco"	3

Ejemplo 3: Agrupaciones

Para cada persona agrupar por sectores en que trabajan sus amigos

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

The screenshot shows the Neo4j Browser interface. The top panel displays the Cypher query: `neo4j$ MATCH (p1:Person)-[:FRIEND_OF]-(p2:Person)-[:WORK_AT]-(c:Company) RETURN p1.name, COLLECT(c.sector)`. The results are shown in a table with two columns: `p1.name` and `COLLECT(c.sector)`. The table contains six rows of data.

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

Below the table, a status message reads: "Started streaming 6 records in less than 1 ms and completed after 3 ms." The bottom panel shows a graph view with nodes for `Person` and `Company`, and relationships for `FRIEND_OF` and `WORK_AT`.

Modificar propiedades a un nodo

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

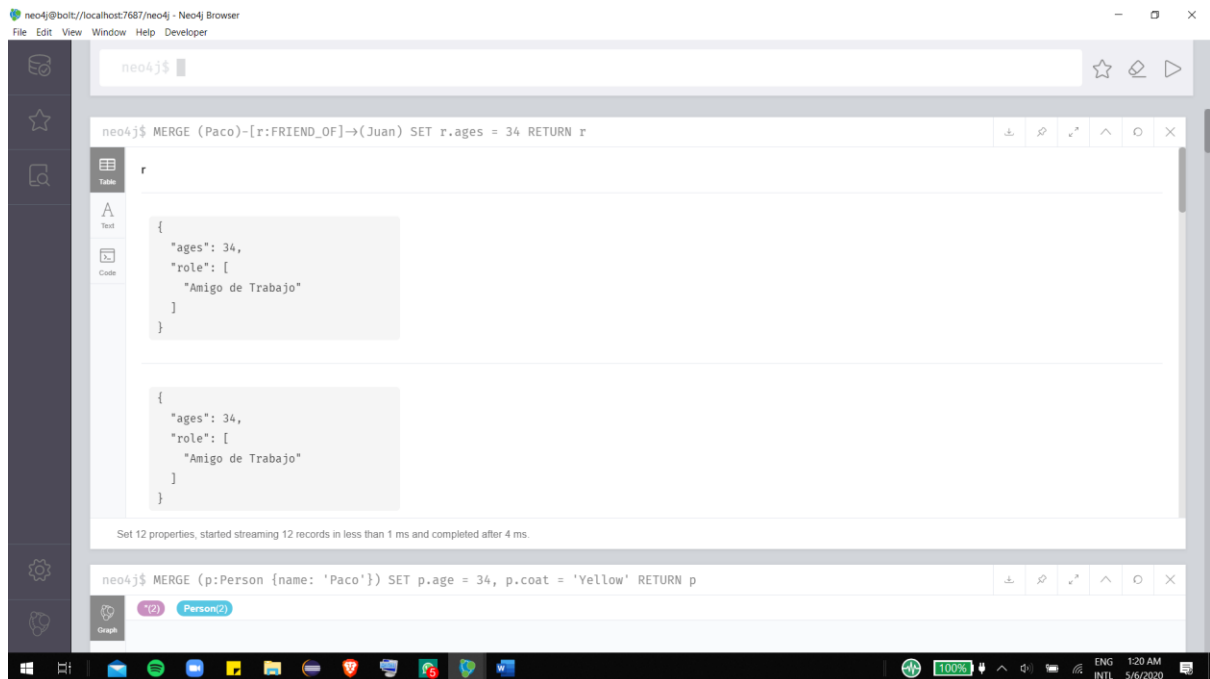
The screenshot shows the Neo4j Browser interface. The top panel displays the Cypher query: `neo4j$ MERGE (p:Person {name: 'Paco'}) SET p.age = 34, p.coat = 'Yellow' RETURN p`. The results are shown in a table with one column: `Person`. The table contains one row of data.

Person
<id>: 0 age: 34 born: 1964 coat: Yellow name: Paco

Below the table, a status message reads: "Started streaming 1 record in less than 1 ms and completed after 3 ms." The bottom panel shows a graph view with nodes for `Person` and `Company`, and relationships for `FRIEND_OF` and `WORK_AT`.

Modificar propiedades a una relación

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



Borrar nodos

```
MATCH (p:Person {name: «Paco»}), (c:Company {name: «Telefonica»}) DELETE p, c
```

Nota: Para poder borrar los nodos se tienen que borrar las relaciones entre ellos

Borrar relaciones entre nodos

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

neo4j@bolt://localhost:7687/neo4j - Neo4j Browser

File Edit View Window Help Developer

neo4j\$

neo4j\$ match (n) return n


Table

(no changes, no records)

Code

Completed after 2 ms.

\$:play start

 neo4j

Learn about Neo4j

A graph epiphany awaits you.

- What is a graph database?
- How can I query a graph?
- What do people do with Neo4j?

[Start learning](#)

Jump into code

Use Cypher, the graph query language.

- Code walk-throughs
- RDBMS to Graph

[Write code](#)

System information

Key system health and status metrics.

- Store sizes
- ID allocation
- Page cache
- Transaction count
- Cluster status

[Monitor](#)

