



---

CARRERA DE ESPECIALIZACIÓN EN INTELIGENCIA ARTIFICIAL

---

## **Bases de Datos para Inteligencia Artificial**

16Co2024

### **Trabajo Práctico N°3**

#### ***Grafos***

DEGANO, MYRNA LORENA

N° SIU a1618

[myrna.l.degano@gmail.com](mailto:myrna.l.degano@gmail.com)

abril de 2025

Índice

SetUp..... 3

Consultas..... 5

1) ..... 5

2) ..... 6

3) ..... 6

4) ..... 7

5) ..... 8

6) ..... 8

7) ..... 9

8) ..... 10

9) ..... 11

10) ..... 11

11)..... 12

12) ..... 13

13) ..... 14

14) ..... 14

15) ..... 15

16) ..... 16

17) ..... 16

18) ..... 17

19) ..... 17

20) ..... 18

21) ..... 18

22) ..... 19

23) ..... 20

24) ..... 20

---

25) .....	21
26) .....	21
27) .....	22
28) .....	23
29) .....	23
30) .....	25
Anexos.....	26
Referencias.....	26

## SetUp

Para crear el entorno de trabajo con Docker, ejecutar:

```
docker-compose up -d
```

Esto crea el contenedor de Neo4J:

CONTAINER ID	IMAGE	COMMAND	NAMES	CREATED	STATUS	PORTS
d37c19174939	neo4j:latest	"tini -g -- /startup_"		28 seconds ago	Up 18 seconds	0.0.0.0:7474->7474/tcp, [::]:7474->7474/tcp, 7473/tcp, 0.0.0.0:7687->7687/tcp, [::]:7687->7687/tcp
			neo4j			

Para inicializar la base de datos **Personas**:

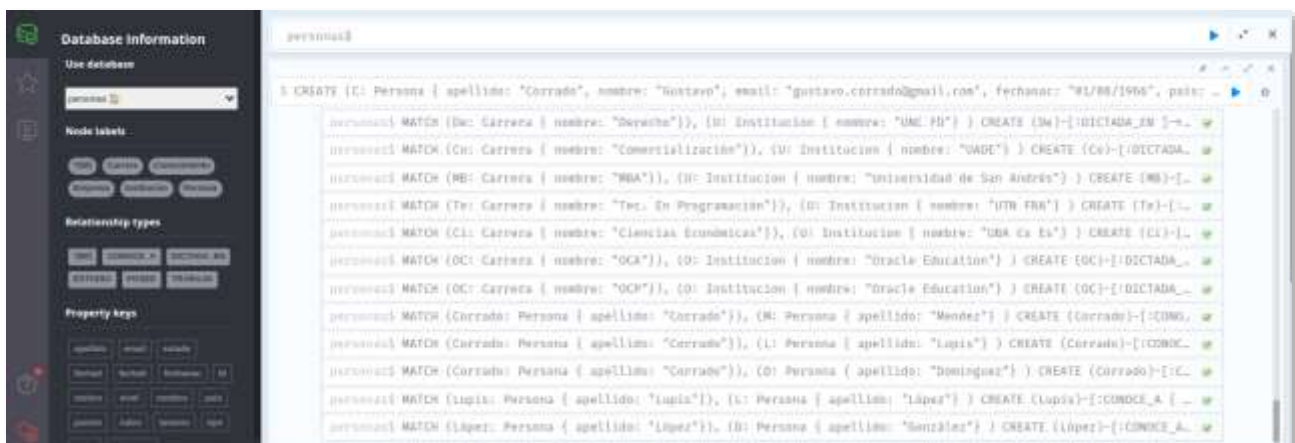
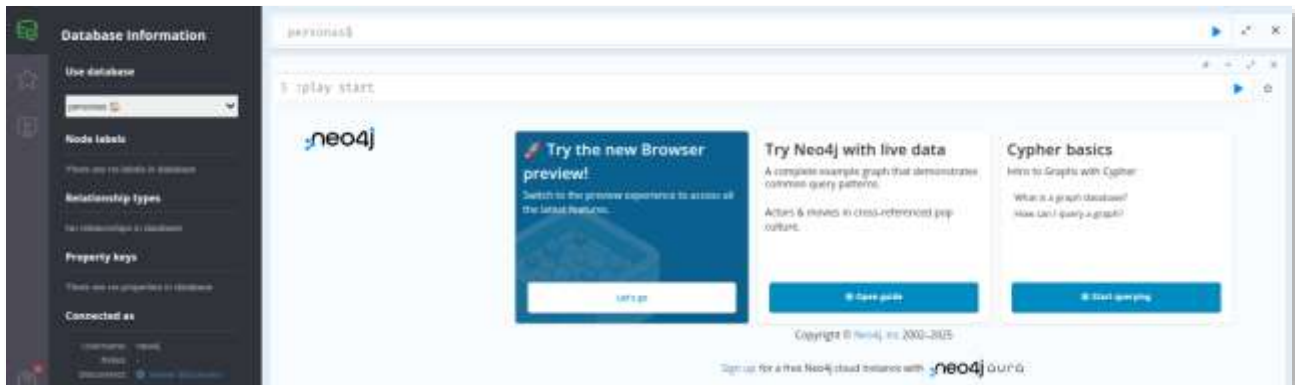
Dado que la versión no permite múltiples bases de datos, modificar el archivo de configuración (`/var/lib/neo4j/conf/neo4j.conf`) y reiniciar el contenedor:

```
# Neo4j configuration
#
# For more details and a complete list of settings, please see
# https://neo4j.com/docs/operations-manual/current/reference/configuration-settings/
#
# The name of the default database
initial.dbms.default_database=Personas

# Paths of directories in the installation.
#server.directories.data=data
#server.directories.plugins=plugins
#server.directories.logs=logs
#server.directories.lib=lib
#server.directories.run=run
#server.directories.licenses=licenses
#server.directories.transaction_logs_root=data/transactions

# This setting constrains all 'LOAD CSV' import files to be under the 'import' directory. Remove or comment it out to
# allow files to be loaded from anywhere in the filesystem; this introduces possible security problems. See the
# 'LOAD CSV' section of the manual for details.
server.directories.import=import
```

Al acceder a <http://localhost:7474>, en el browser posicionado sobre la base de datos “Personas”, copiar y ejecutar el contenido del archivo “Datos\_Ejemplo\_Neo.txt”:

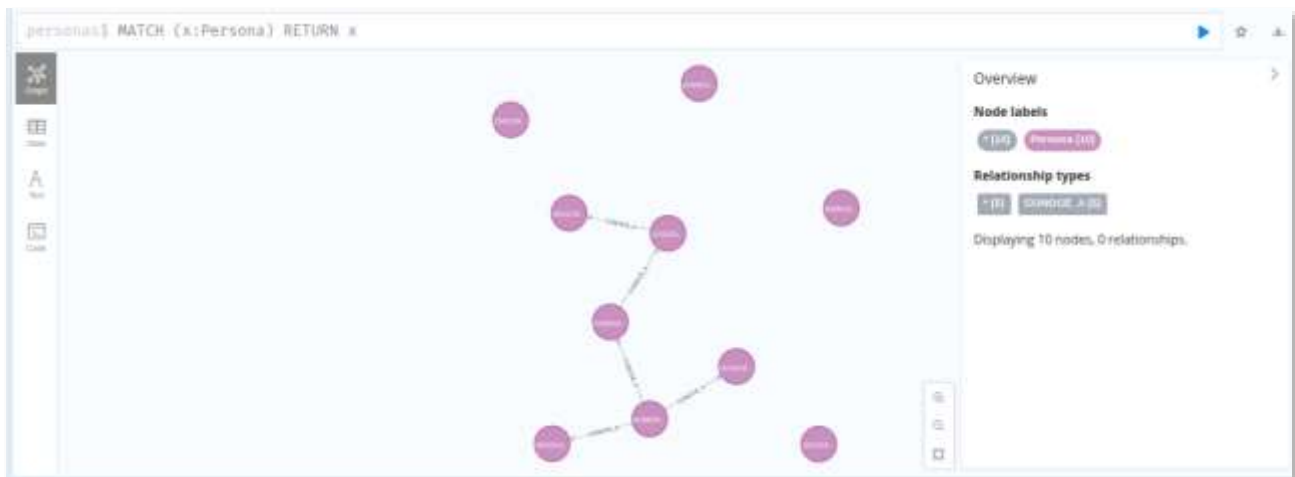


## Consultas

1)

Obtener los nodos de todas las personas de la red.

```
MATCH (x:Persona)  
RETURN x
```



The screenshot shows a graph database interface with a table of results for the query `MATCH (x:Persona) RETURN x`. The table contains 10 rows of data, each representing a person with fields for `fechanac`, `apellido`, `nombre`, `email`, and `pais`.

x
{:Persona {fechanac: "01/08/1968", apellido: "Corrado", nombre: "Gustavo", email: "gustavo.corrado@gmail.com", pais: "Argentina"}}
{:Persona {fechanac: "10/12/1978", apellido: "Diaz", nombre: "Analía", email: "adiaz@hotmail.com", pais: "Argentina"}}
{:Persona {fechanac: "21/10/1990", apellido: "Dominguez", nombre: "Mariana", email: "mariana@yahoo.com", pais: "Chile"}}
{:Persona {fechanac: "28/05/1992", apellido: "Pereyra", nombre: "Claudio", email: "cpereyra200@yahoo.com.ar", pais: "Estados Unidos"}}
{:Persona {fechanac: "11/02/1978", apellido: "López", nombre: "Mario", email: "mario.lopez@gmail.com", pais: "Argentina"}}
{:Persona {fechanac: "07/04/1972", apellido: "Pereira", nombre: "Natalia", email: "nf@hotmail.com", pais: "Argentina"}}
{:Persona {fechanac: "23/01/1985", apellido: "García", nombre: "Eduardo", email: "garcia@live.com.ar", pais: "Chile"}}
{:Persona {fechanac: "09/11/1974", apellido: "González", nombre: "Bibiana", email: "bibiana@live.com.ar", pais: "España"}}
{:Persona {fechanac: "27/09/1989", apellido: "Lupis", nombre: "Jorge", email: "jlup@gmail.com", pais: "Argentina"}}
{:Persona {fechanac: "29/02/1968", apellido: "Mendez", nombre: "Verónica", email: "veromendi@yahoo.com.ar", pais: "Argentina"}}

2)

Obtener el nombre y fecha de nacimiento de la persona de apellido Dominguez.

```
MATCH (a:Persona {apellido: "Dominguez"})  
RETURN a.nombre, a.fechanac
```



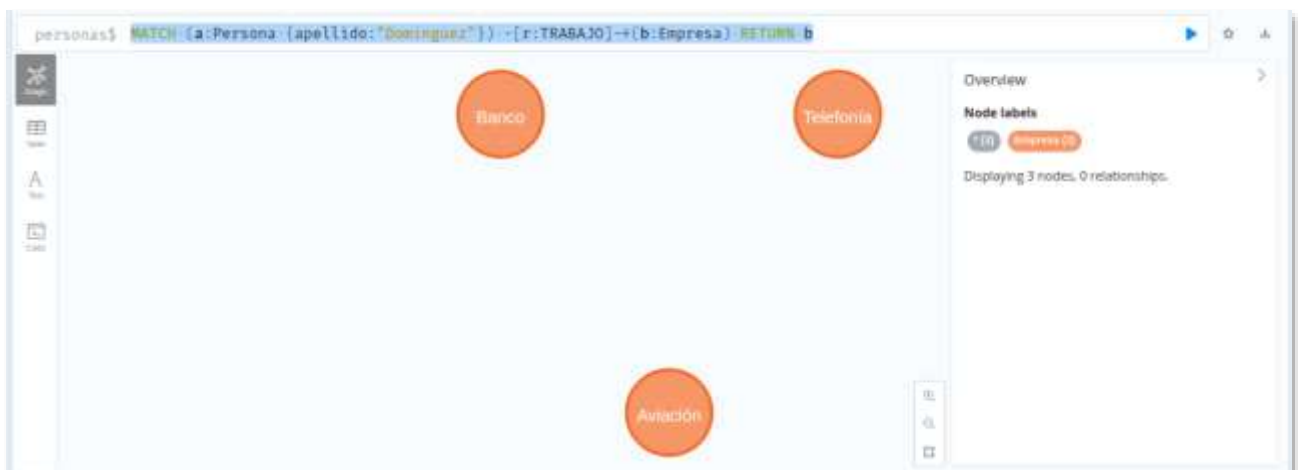
The screenshot shows a query execution interface with a command bar containing the query: `personas$ MATCH (a:Persona {apellido: "Dominguez"}) RETURN a.nombre, a.fechanac`. Below the command bar is a table with two columns: `a.nombre` and `a.fechanac`. The table contains one row with the values `"Myrina"` and `"31/10/1990"`.

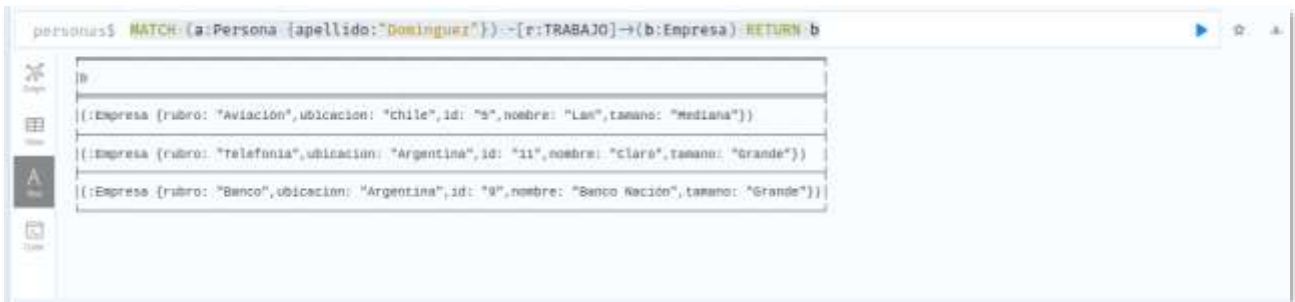
a.nombre	a.fechanac
"Myrina"	"31/10/1990"

3)

Obtener la lista de empresas en las que trabajó Dominguez.

```
MATCH (a:Persona {apellido:"Dominguez"}) -[r:TRABAJO]->(b:Empresa)  
RETURN b
```



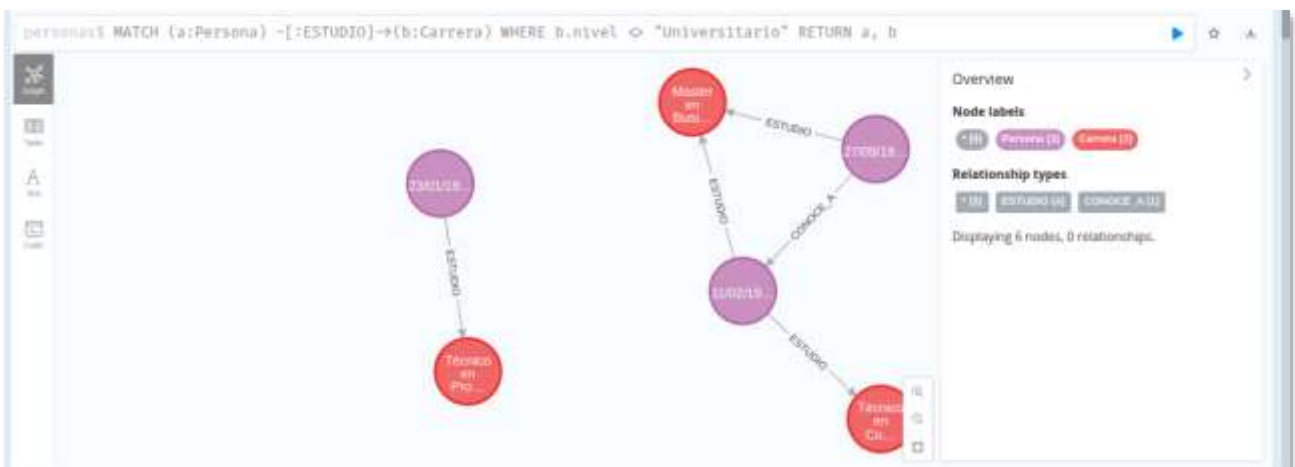


4)

Obtener la lista de personas que estudiaron carreras que no son de nivel "Universitario" y los nombres de las carreras.

```

MATCH (a:Persona) -[:ESTUDIO]->(b:Carrera)
WHERE b.nivel <> "Universitario"
RETURN a, b
  
```



```

personas$ MATCH (a:Persona) -[:ESTUDIO]->(b:Carrera) WHERE b.nivel <> "Universitario" RETURN a, b
  
```

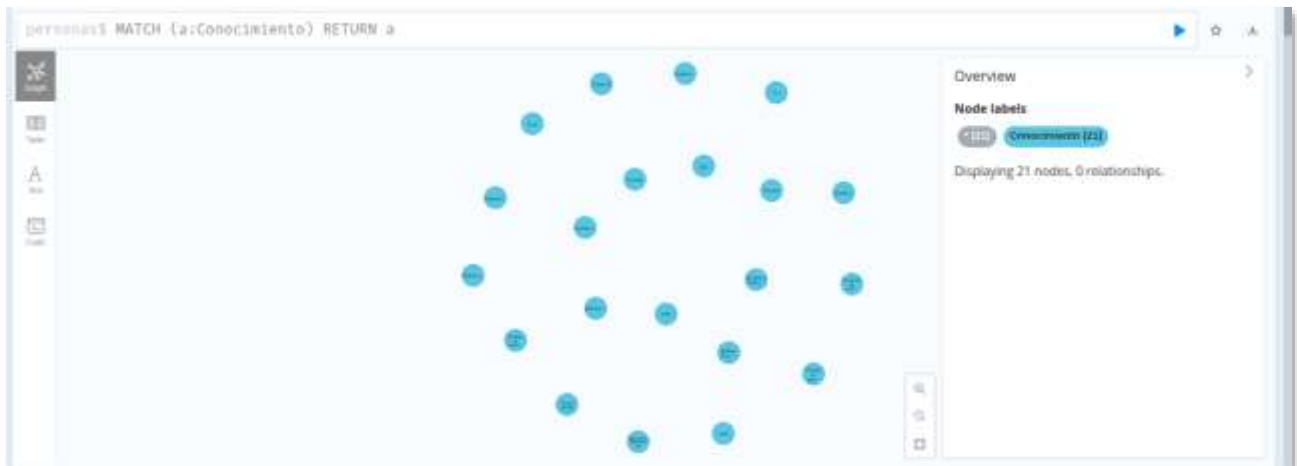
a	b
{:Persona {fecha_nac: "11/02/1970", apellido: "López", nombre: "Mario", email: "mario.lopez@gmail.com", pais: "Argentina"}}	{:Carrera {titulo: "Técnico en Comercialización", nombre: "Comercialización", nivel: "Terciario"}}
{:Persona {fecha_nac: "27/09/1998", apellido: "Luis", nombre: "Jorge", email: "jlup@gmail.com", pais: "Argentina"}}	{:Carrera {titulo: "Master en Business Administration", nivel: "Postgrado", nombre: "MBA"}}
{:Persona {fecha_nac: "11/02/1970", apellido: "López", nombre: "Mario", email: "mario.lopez@gmail.com", pais: "Argentina"}}	{:Carrera {titulo: "Master en Business Administration", nivel: "Postgrado", nombre: "MBA"}}
{:Persona {fecha_nac: "22/01/1995", apellido: "García", nombre: "Eduardo", email: "garcia@live.com.ar", pais: "Chile"}}	{:Carrera {titulo: "Técnico en Programación", nombre: "Tec. En Programación", nivel: "Terciario"}}



5)

Obtener los nodos etiquetados como Conocimiento.

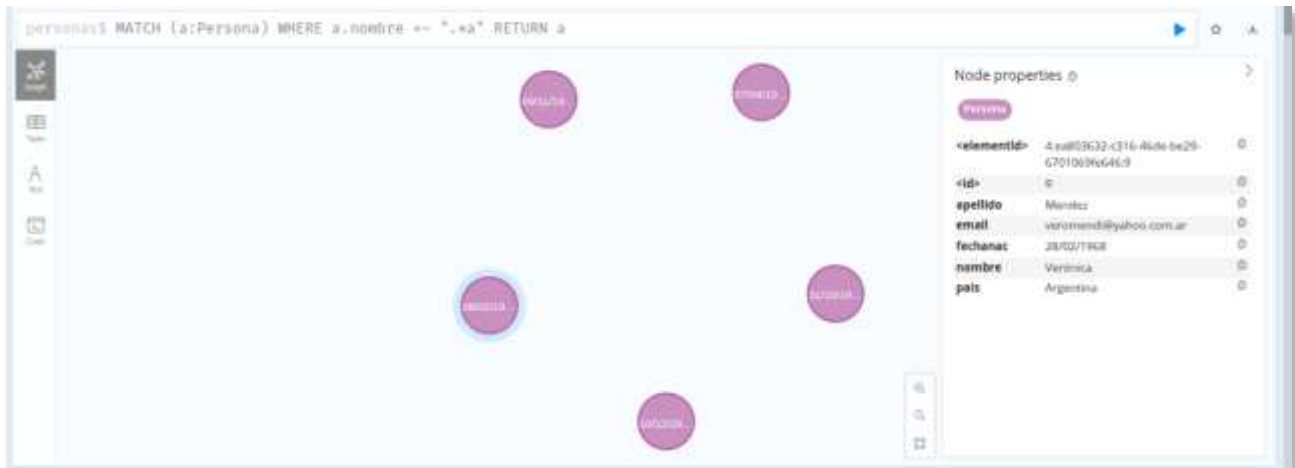
```
MATCH (a:Conocimiento)
RETURN a
```



6)

Obtener los nodos de todas las personas con nombre terminado en a.

```
MATCH (a:Persona)
WHERE a.nombre =~ ".*a"
RETURN a
```



a
{:Persona {fechaac: "19/12/1978",apellido: "Diaz",nombre: "Analía",email: "mdiaz@hotmail.com",pais: "Argentina"}}
{:Persona {fechaac: "31/19/1998",apellido: "Dominguez",nombre: "Mariana",email: "sariana@yahoo.com",pais: "Chile"}}
{:Persona {fechaac: "07/04/1973",apellido: "Ferreira",nombre: "Natalia",email: "nf@hotmail.com",pais: "Argentina"}}
{:Persona {fechaac: "09/11/1974",apellido: "González",nombre: "Mibiana",email: "mibiana@live.com.ar",pais: "España"}}
{:Persona {fechaac: "28/02/1968",apellido: "Mendez",nombre: "Verónica",email: "veromendi@yahoo.com.ar",pais: "Argentina"}}

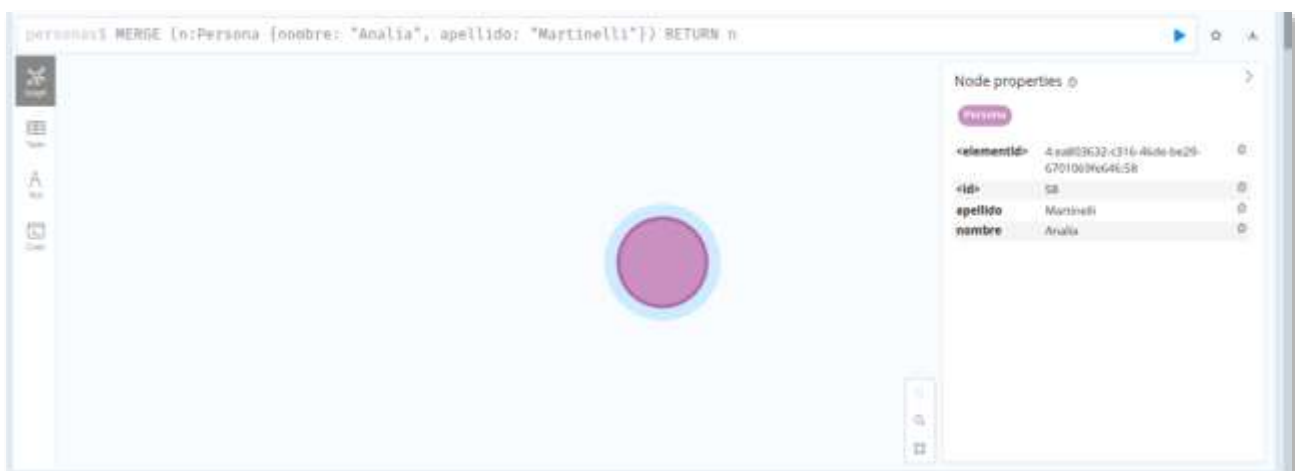
7)

Crear un nodo para la persona Analía Martinelli si no existe.

```

MERGE (n:Persona {nombre: "Analía", apellido: "Martinelli"})
RETURN n

```





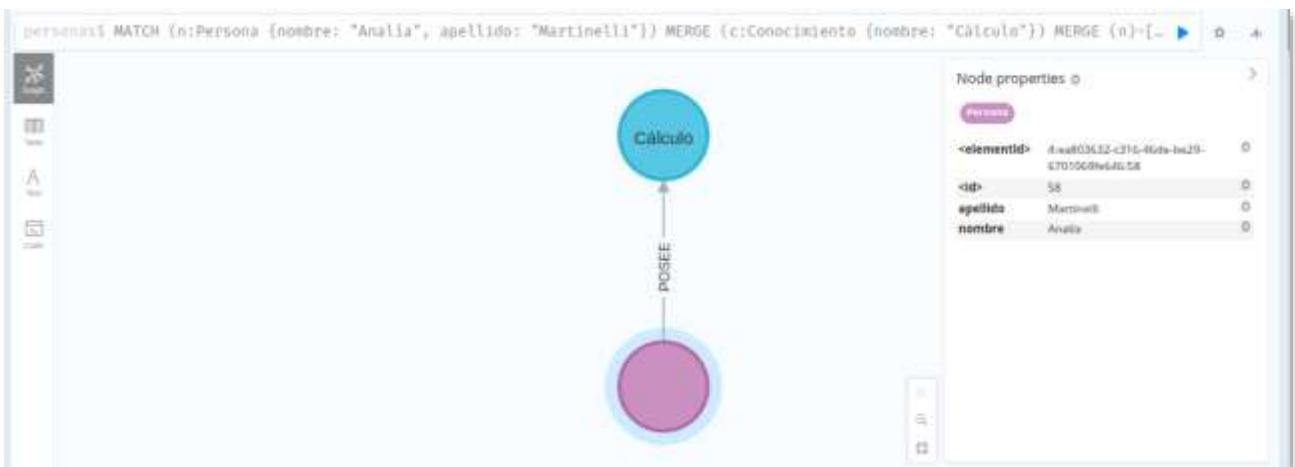
8)

Asociar un conocimiento "Cálculo" a Analía Martinelli si no lo posee.

```

MATCH (n:Persona {nombre: "Analía", apellido: "Martinelli"})
MERGE (c:Conocimiento {nombre: "Cálculo"})
MERGE (n)-[p:POSEE]->(c)
RETURN n, p, c

```



```

personas$ MATCH (n:Persona {nombre: "Analía", apellido: "Martinelli"}) MERGE (c:Conocimiento {nombre: "Cálculo"}) MERGE (n)-[p:POSEE]->(c)

```

The screenshot shows a Cypher query editor with a query that matches a person node, creates a knowledge node named "Cálculo", and creates a relationship named "POSEE" between them. The result is displayed as a table with three columns: n, p, and c.

n	p	c
[:Persona {apellido: "Martinelli", nombre: "Analía"}]	[:POSEE]	[:Conocimiento {nombre: "Cálculo"}]

9)

Verificar si se creó duplicado del conocimiento "Cálculo".

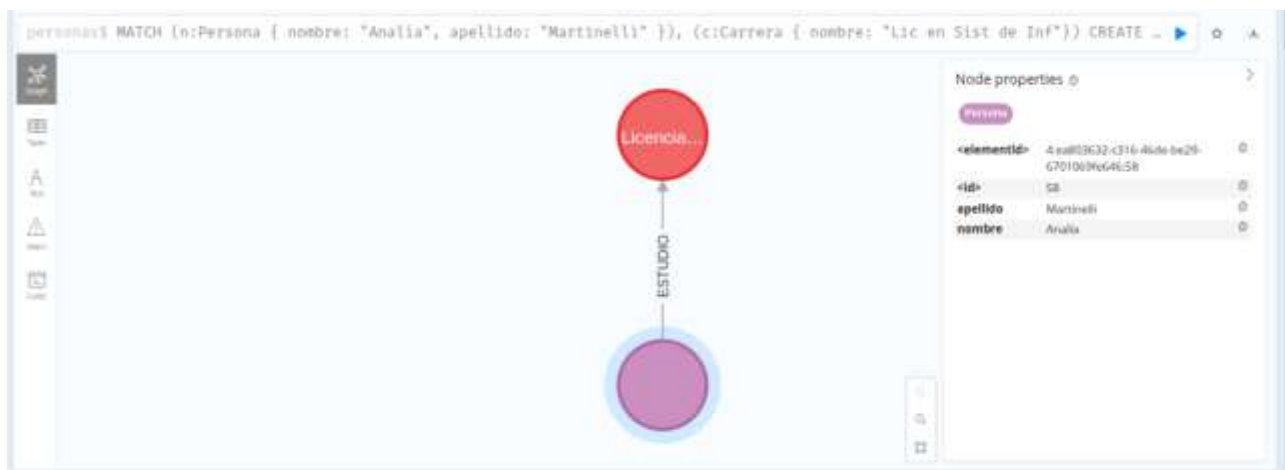
```
MATCH (n:Conocimiento {nombre: "Cálculo"})  
RETURN n
```



10)

Crear una relación ESTUDIO para Analía Martinelli con la carrera "Lic en Sist de Inf", estado "En curso".

```
MATCH (n:Persona {nombre: "Analía", apellido: "Martinelli"}), (c:Carrera  
{nombre: "Lic en Sist de Inf"})  
CREATE (n)-[r:ESTUDIO {estado: "En curso"}]->(c)  
RETURN n, r, c
```



The screenshot shows a graph database interface. At the top, a Cypher query is entered: `PERSONAS MATCH (n:Persona { nombre: "Analía", apellido: "Martinelli" }), (c:Carrera { nombre: "Lic en Sist de Inf" }) CREATE ...`. Below the query, a table visualization is shown with the following data:

n	r	c
{:Persona {apellido: "Martinelli", nombre: "Analía"}}	{:ESTUDIO {estado: "En curso"}}	{:Carrera {título: "Licenciado en Sistemas de Información", nombre: "Lic en Sist de Inf", nivel: "Universitario"}}

11)

Crear un nodo para Verónica Mendez.

```
MERGE (n:Persona {nombre: "Verónica", apellido: "Mendez"})
RETURN n
```





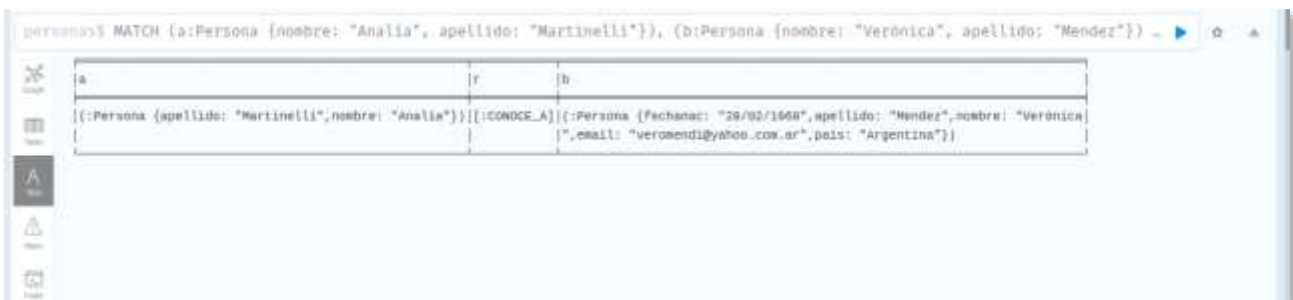
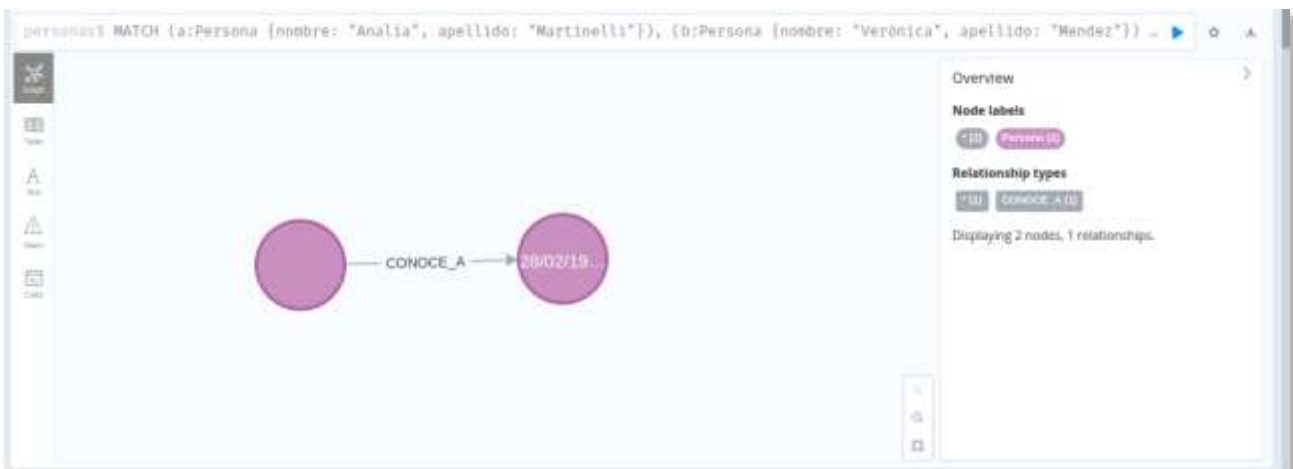
12)

Crear una relación CONOCE\_A entre Analía y Verónica, asegurando que solo se cree una vez.

```

MATCH (a:Persona {nombre: "Analía", apellido: "Martinelli"}), (b:Persona
{nombre: "Verónica", apellido: "Mendez"})
MERGE (a)-[r:CONOCE_A]->(b)
RETURN a, r, b

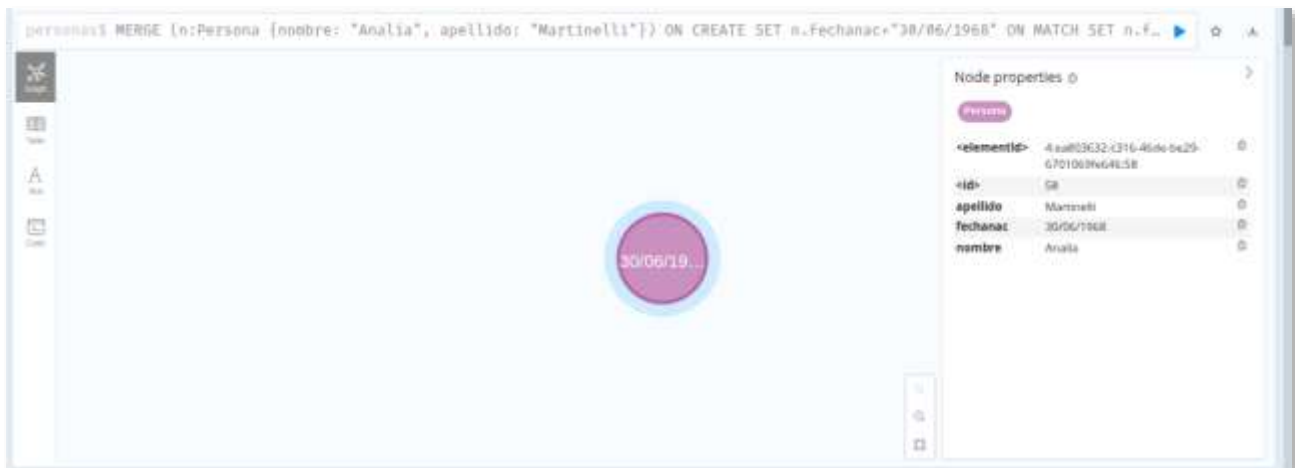
```



13)

Actualizar o crear el nodo de Analía Martinelli con fecha de nacimiento 30/06/1968.

```
MERGE (n:Persona {nombre: "Analía", apellido: "Martinelli"})  
ON CREATE SET n.fechanac="30/06/1968"  
ON MATCH SET n.fechanac="30/06/1968"  
RETURN n
```



14)

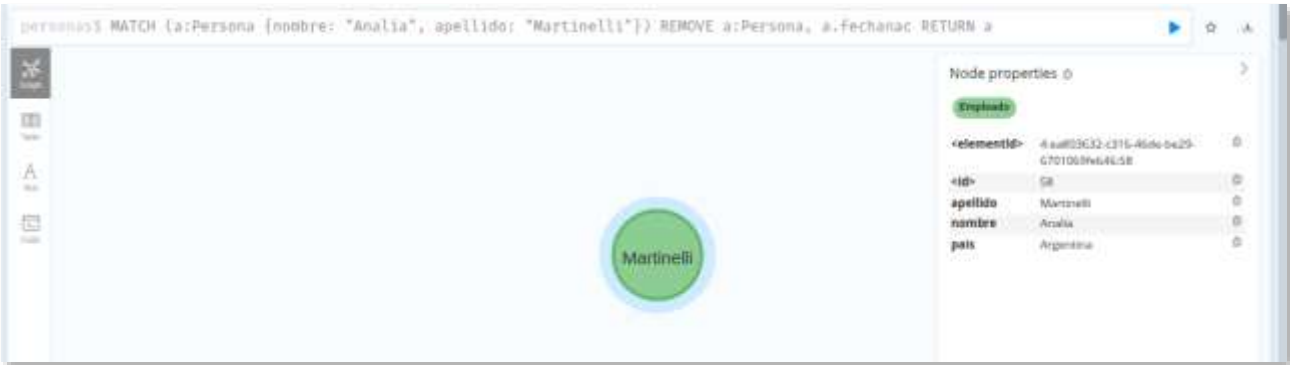
Agregarle la etiqueta "Empleado" y el país Argentina a Analía.

```
MATCH (a:Persona {nombre: "Analía", apellido: "Martinelli"})  
SET a:Empleado, a.pais="Argentina"  
RETURN a
```



15)  
Eliminar la fecha de nacimiento y la etiqueta Persona de Analía.

```
MATCH (a:Persona {nombre: "Analía", apellido: "Martinelli"})
REMOVE a:Persona, a.fechanac
RETURN a
```







16)

Eliminar el nodo de Analía y todas sus relaciones.

```
MATCH (a {nombre: "Analía", apellido: "Martinelli"})-[]-()
DELETE r, a
```



17)

Contar los nodos en total.

```
MATCH (n)
RETURN count(*)
```



18)

Contar los tipos de relaciones.

```
MATCH (n)-[r]->()
RETURN type(r), count(*)
```



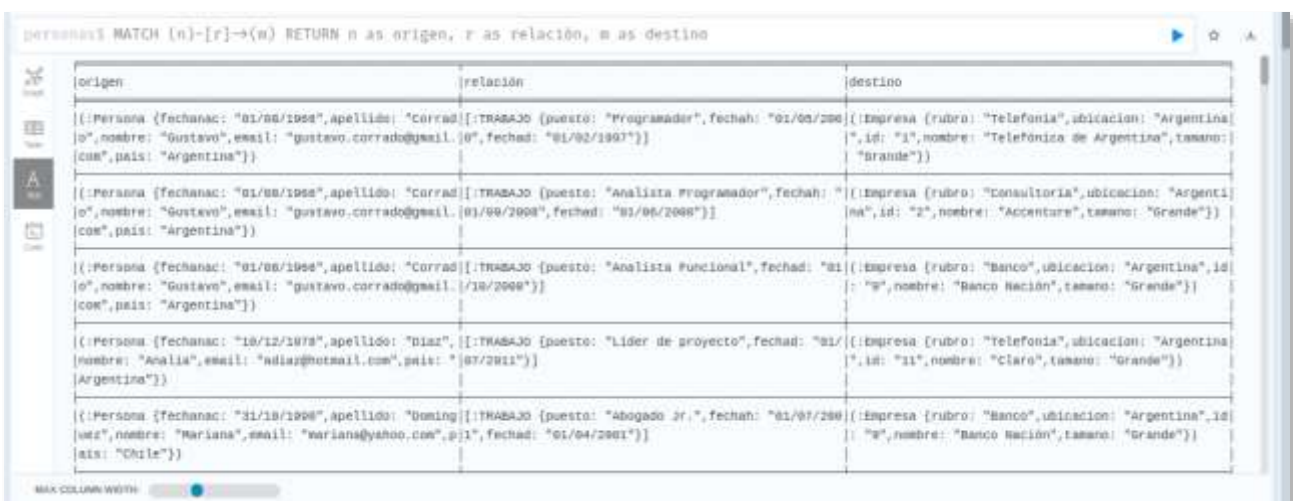
Query: `MATCH (n)-[r]->() RETURN type(r), count(*)`

type(r)	count(*)
"TRABAJO"	22
"PIESE"	33
"ESTUDIO"	13
"DICTADA_Es"	9
"CONOCE_A"	5

19)

Listar todos los nodos y sus relaciones.

```
MATCH (n)-[r]->(m)
RETURN n as origen, r as relación, m as destino
```



Query: `MATCH (n)-[r]->(m) RETURN n as origen, r as relación, m as destino`

origen	relación	destino
{:Persona {fechanac: "01/06/1994", apellido: "Corrado", nombre: "Gustavo", email: "gustavo.corrado@gmail.com", pais: "Argentina"}}	[:TRABAJO {puesto: "Programador", fechah: "01/06/2006", fechad: "01/02/1991"}]	{:Empresa {rubro: "Telefonia", ubicacion: "Argentina", id: "1", nombre: "Telefónica de Argentina", tamaño: "Grande"}}
{:Persona {fechanac: "01/06/1994", apellido: "Corrado", nombre: "Gustavo", email: "gustavo.corrado@gmail.com", pais: "Argentina"}}	[:TRABAJO {puesto: "Analista Programador", fechah: "01/06/2008", fechad: "01/06/2008"}]	{:Empresa {rubro: "Consultoria", ubicacion: "Argentina", id: "2", nombre: "Accenture", tamaño: "Grande"}}
{:Persona {fechanac: "01/06/1994", apellido: "Corrado", nombre: "Gustavo", email: "gustavo.corrado@gmail.com", pais: "Argentina"}}	[:TRABAJO {puesto: "Analista Funcional", fechah: "01/06/2008", fechad: "01/06/2008"}]	{:Empresa {rubro: "Banco", ubicacion: "Argentina", id: "3", nombre: "Banco Nación", tamaño: "Grande"}}
{:Persona {fechanac: "10/12/1978", apellido: "Diaz", nombre: "Analia", email: "ndiaz@hotmail.com", pais: "Argentina"}}	[:TRABAJO {puesto: "Lider de proyecto", fechah: "01/07/2011", fechad: "07/2011"}]	{:Empresa {rubro: "Telefonia", ubicacion: "Argentina", id: "11", nombre: "Claro", tamaño: "Grande"}}
{:Persona {fechanac: "31/10/1998", apellido: "Dominguez", nombre: "Mariana", email: "mariana@yahoo.com", pais: "Chile"}}	[:TRABAJO {puesto: "Abogado Jr.", fechah: "01/07/2009", fechad: "01/04/2001"}]	{:Empresa {rubro: "Banco", ubicacion: "Argentina", id: "8", nombre: "Banco Nación", tamaño: "Grande"}}

20)

Obtener los nombres y rubros de las empresas registradas, reemplazando el rubro "Telefonía" por IT.

```
MATCH (e: Empresa)
RETURN e.nombre as empresa,
CASE e.rubro
WHEN "Telefonía" THEN "IT"
ELSE e.rubro
END as rubro
```

PERSONS MATCH (e: Empresa) RETURN e.nombre as empresa, CASE e.rubro WHEN "Telefonía" THEN "IT" ELSE e.rubro END as rubro

empresa	rubro
"telefónica de Argentina"	"IT"
"Accenture"	"consultoría"
"CDI"	"gestión de información farmacéutica"
"Chevron"	"Petrólera"
"Lan"	"Aviación"
"Gómez & Asoc"	"Venta"
"Alteon"	"Capacitación"
"IBN"	"IT"
"Banco Nación"	"Banco"
"Carrefour"	"Supermercado"

MAX COLUMN WIDTH: 100

21)

Determinar qué etiquetas tienen los nodos que son destino de la relación ESTUDIO.

```
MATCH ()-[:ESTUDIO]->(b)
RETURN DISTINCT labels(b)
```

PERSONS MATCH ()-[:ESTUDIO]->(b) RETURN DISTINCT labels(b)

labels(b)
["Carrera"]

22)

Verificar las etiquetas de la carrera en la relación ESTUDIO.

```
MATCH ()-[:ESTUDIO]->(c)
RETURN labels (c);
```



```
MATCH ()-[:ESTUDIO]->(c)
RETURN c.nombre as carrera;
```



23)

Usar UNWIND para transformar una colección en filas individuales.

```
UNWIND['F1','F2','F3','F4','F5'] AS filas  
RETURN filas
```



24)

Contar la cantidad de personas que estudiaron una carrera en cualquier estado.

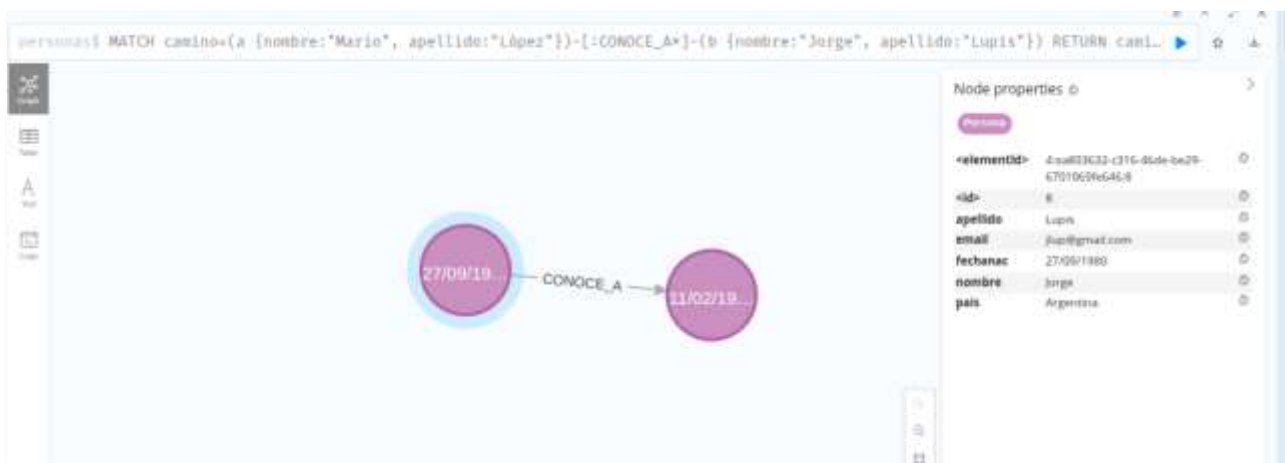
```
MATCH (n:Persona)-[:ESTUDIO]->(c:Carrera)  
RETURN count(distinct n) as estudiantes
```



25)

Identificar si puede llegarse directa o indirectamente desde Mario López hasta Jorge Lupis mediante la relación CONOCE\_A.

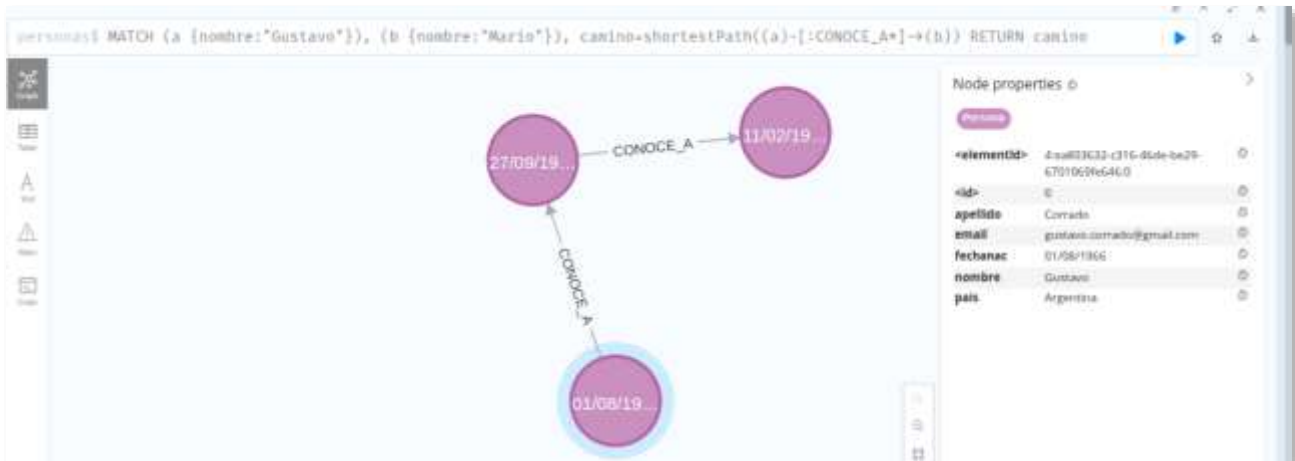
```
MATCH camino=(a {nombre:"Mario", apellido:"López"})-[:CONOCE_A*]-(b
{nombre:"Jorge", apellido:"Lupis"})
RETURN camino
```



26)

Obtener el camino más corto entre Gustavo y Mario en la relación CONOCE\_A.

```
MATCH (a {nombre:"Gustavo"}), (b {nombre:"Mario"}),
camino=shortestPath((a)-[:CONOCE_A*]->(b))
RETURN camino
```



27)

Listar los caminos de relaciones de un camino determinado.

```
MATCH camino=(a {nombre:"Mario", apellido:"López"})-[:CONOCE_A*]-(b
{nombre:"Jorge", apellido:"Lupis"})
RETURN relationships(camino)
```



28)

Verificar si una persona trabajó o trabajó en empresas que otro determinado profesional trabajo, para sugerir contactos.

```
MATCH (a:Persona)-[:TRABAJO]->( )<-[:TRABAJO]-(b:Persona)
WHERE id(a) <> id(b)
AND NOT (a)-[:CONOCE_A]-(b)
RETURN DISTINCT a.nombre, a.apellido, b.nombre, b.apellido
```

The screenshot shows a query execution window with the following Cypher query:

```
personas MATCH (a:Persona)-[:TRABAJO]->( )<-[:TRABAJO]-(b:Persona) WHERE id(a) <> id(b) AND NOT (a)-[:CONOCE_A]-(b) RETURN DI...
```

The results are displayed in a table with 4 columns: a.nombre, a.apellido, b.nombre, b.apellido. The table contains 10 rows of data:

a.nombre	a.apellido	b.nombre	b.apellido
"Natalia"	"Ferreira"	"Gustavo"	"Corrado"
"Mario"	"López"	"Gustavo"	"Corrado"
"Mario"	"López"	"Analia"	"Díaz"
"Mariana"	"Dominguez"	"Analia"	"Díaz"
"Eduardo"	"García"	"Mariana"	"Dominguez"
"Mario"	"López"	"Mariana"	"Dominguez"
"Analia"	"Díaz"	"Mariana"	"Dominguez"
"Natalia"	"Ferreira"	"Mariana"	"Dominguez"
"Mario"	"López"	"Claudio"	"Pereyra"
"Natalia"	"Ferreira"	"Mario"	"López"

29)

Obtener los conocimientos más compartidos en cada carrera.

```
MATCH (a:Persona)-[:POSEE]->(c:Conocimiento), (a)-[:ESTUDIO]->(d:Carrera)
RETURN d.nombre as carrera, c.nombre as conocimiento, count(distinct a) AS
cantidad
ORDER by d.nombre, cantidad DESC;
```



personas1 MATCH (a:Persona)-[:POSEE]->(c:Conocimiento), (a)-[:ESTUDIO]->(d:Carrera) RETURN d.nombre AS carrera, c.nombre AS c...

carrera	conocimiento	cantidad
"Derecho"	"Gestión"	1
"Derecho"	"Contestación de demandas"	1
"Derecho"	"Creación de sociedades"	1
"Derecho"	"Revisión y elaboración de contratos"	1
"Ing en Sistemas de Información"	"Análisis Funcional"	3
"Ing en Sistemas de Información"	"Java"	3
"Ing en sistemas de información"	"Gestión de proyectos"	2
"Ing en sistemas de información"	"SQL"	1
"Ing en Sistemas de Información"	"C++"	1
"Ing en Sistemas de Información"	"PHP"	1

// Un único conocimiento por carrera:

```

MATCH (a:Persona)-[:POSEE]->(c:Conocimiento), (a)-[:ESTUDIO]->(d:Carrera)
WITH DISTINCT d.nombre AS carrera

CALL {
  WITH carrera
  MATCH (a:Persona)-[:POSEE]->(c:Conocimiento), (a)-[:ESTUDIO]->(d:Carrera)
  WHERE d.nombre = carrera
  WITH c.nombre AS conocimiento, count(DISTINCT a) AS cant
  ORDER BY cant DESC
  RETURN conocimiento, cant AS topCantidad
  LIMIT 1
}

RETURN carrera, conocimiento, topCantidad AS cantidad
ORDER BY carrera

```

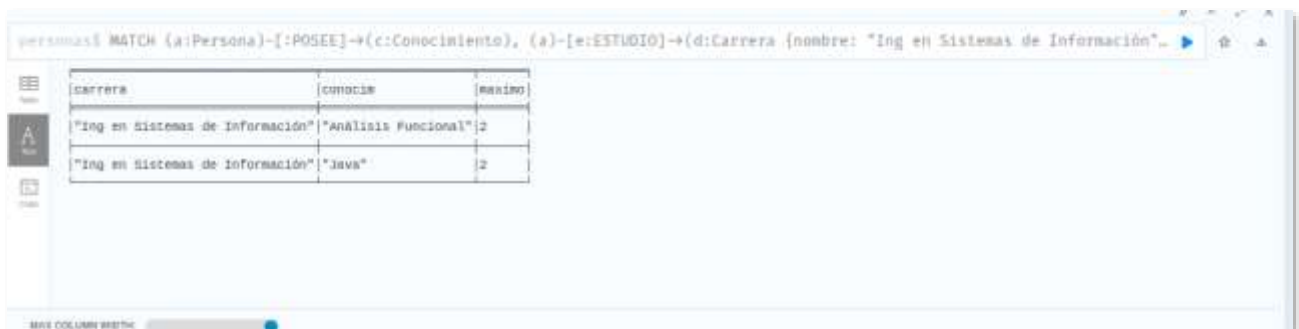
personas1 // Paso 1: obtenemos la lista de carreras MATCH (a:Persona)-[:POSEE]->(c:Conocimiento), (a)-[:ESTUDIO]->(d:Carrera)

carrera	conocimiento	cantidad
"Derecho"	"Gestión"	1
"Ing en Sistemas de Información"	"Análisis Funcional"	3
"Lic en Sist de Inf"	"Evaluación de proveedores"	1
"MBA"	"Gestión de recursos"	1
"Tec. En Programación"	"SQL"	1

30)



Ranking de los primeros 2 conocimientos de la carrera "Ing en Sistemas de Información".

```
MATCH (a:Persona)-[:POSEE]->(c:Conocimiento), (a)-[:ESTUDIO]->(d:Carrera {nombre: "Ing en Sistemas de Información"})
WHERE e.estado = "Completo"
WITH d.nombre as carrera, c.nombre as conocim, count(DISTINCT a) AS cantidad
RETURN carrera, conocim, max(cantidad) as maximo
ORDER BY carrera, maximo DESC LIMIT 2
```



carrera	conocim	maximo
"Ing en Sistemas de Información"	"Análisis Funcional"	2
"Ing en Sistemas de Información"	"Java"	2

## Anexos

	<ul style="list-style-type: none"><li>• <a href="#">Archivo .txt</a> con las consultas</li></ul>
	<ul style="list-style-type: none"><li>• <a href="#">Repositorio GitHub</a> con el archivo de configuración del entorno y los inputs.</li></ul>

## Referencias

- [https://github.com/FIUBA-Posgrado-Inteligencia-Artificial/BDIA/tree/main/clase\\_6](https://github.com/FIUBA-Posgrado-Inteligencia-Artificial/BDIA/tree/main/clase_6)
- <https://neo4j.com/>