

# Curso NoSQL Neo4J



Consultas y eliminación – Nodos – Relaciones



# Indice

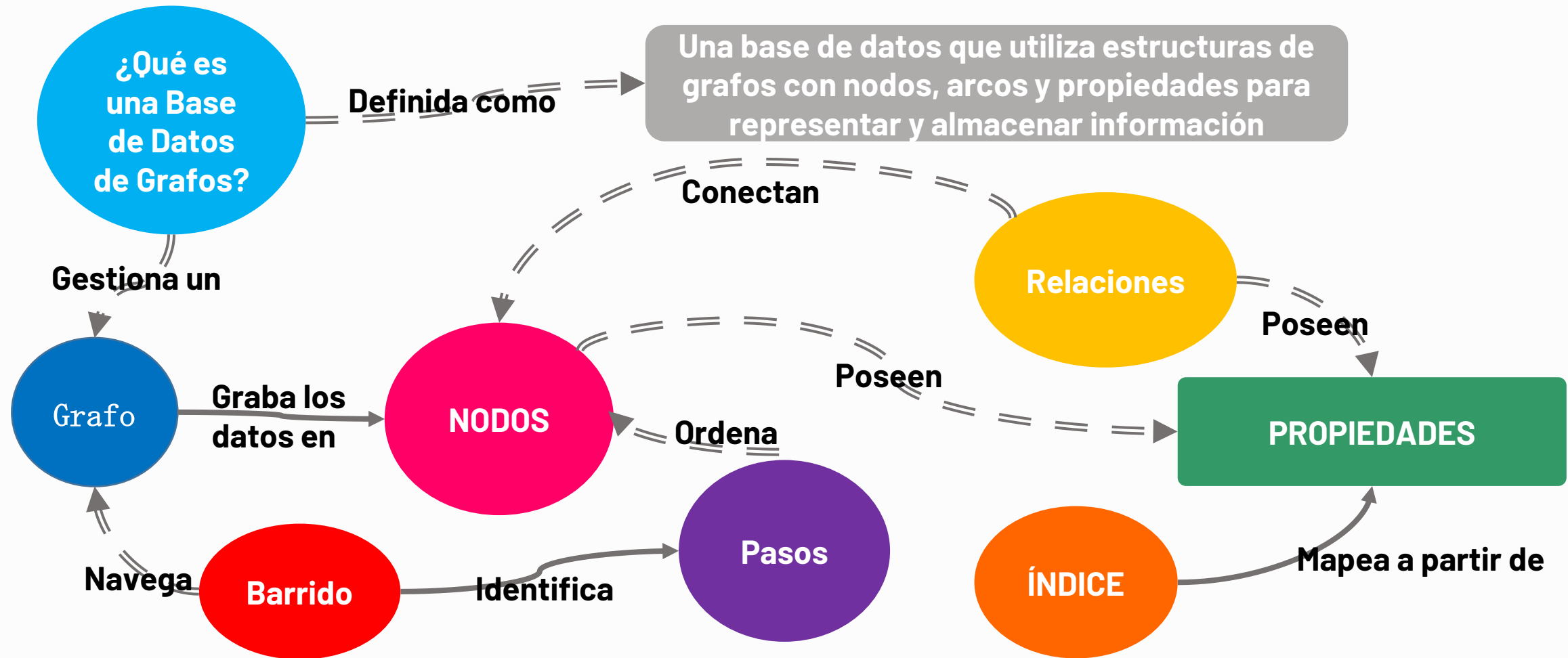
- ★ Repaso de grafos, nodos y relaciones
- ★ Consultar Nodos
- ★ Cláusula WHERE
  - Operadores AND OR XOR y NOT
  - Operadores =, >, <, >=, <=, <>
  - Operador EXISTS y NOT EXISTS
- ★ Cláusula ORDER BY
- ★ Cláusulas LIMIT y SKIP
- ★ Paginados por BDs
- ★ Consultar Relaciones y Nodos
  - Operadores DISTINCT, \*, IN, IS NULL y Cláusula WITH
  - Operadores para Matcheo de Strings: STARTS WITH – ENDS WITH – CONTAINS
  - Expresiones Regulares Case Insensitive y Caracter de Escape
  - Rango Simple y Rango Compuesto
- ★ Eliminar Nodos o Relaciones
  - Eliminar Nodos
  - Eliminar Relaciones
  - Eliminar Nodos y sus Relaciones



# Repaso de grafos, nodos y relaciones



# Graph Based



# Nodos

---

**Unidad fundamental de almacenamiento de información.**

( identificador )

( identificador: etiqueta )

( identificador: etiqueta { propiedades } )

Ejemplo:

```
( nodo1: Persona { apellido: "Corrado", nombre: "Gustavo" } )
```



# Relaciones

---

- **Son claves para identificar la información relacionada en el grafo.**
- **Tienen dirección y tipo.**

$(\text{nodo1}) \dashrightarrow (\text{nodo2})$

$(\text{nodo1}) \text{--} [\text{identificador}] \rightarrow (\text{nodo2})$

$(\text{nodo1}) \text{--} [\text{identificador} : \text{TIPO}] \rightarrow (\text{nodo2})$

$(\text{nodo1}) \text{--} [\text{identificador} : \text{TIPO} \{ \text{propiedades} \}] \rightarrow (\text{nodo2})$

$(\text{nodo1}) \dashrightarrow (\text{nodo2})$

Ejemplo:

```
(Gonzalez)-[:ESTUDIO { estado: "Completo" }]->(CsEs)
```



# Consultar Nodos



# Consultar Nodos

---

## Consultar un Nodo

- Consultar un nodo con las propiedades dadas.

```
MATCH (n:Etiqueta{ propiedades } )  
WHERE condición  
RETURN n
```

**MATCH:** Especifica los patrones para buscar datos en la Base de Datos.

**WHERE:** Agrega restricciones a los patrones de búsqueda del MATCH.

**RETURN:** Define que se incluye en el Conjunto Resultado.





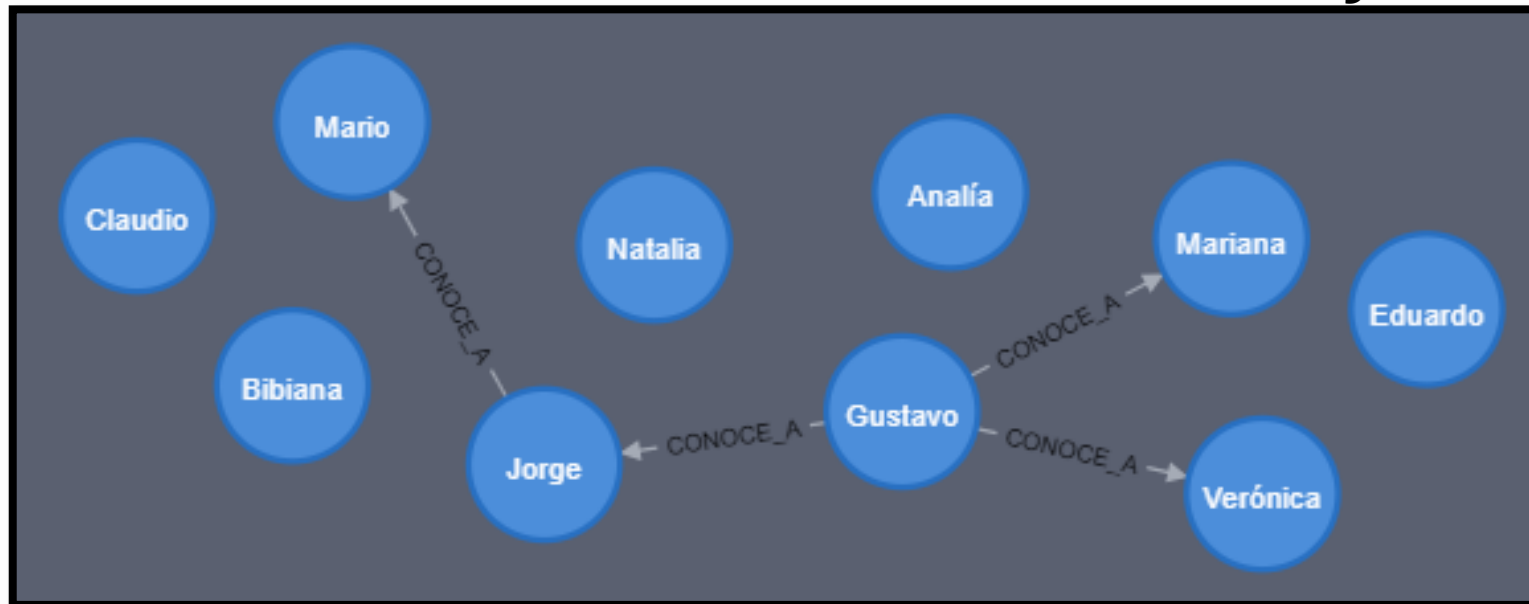
# Consultar Nodos

## Ejemplo 1

- Obtener los **nodos** de todas las **personas** de la red.

**MATCH ( a:Persona )**  
**RETURN a**

## Visualización gráfica



# Consultar Nodos

## Ejemplo 1

- Obtener los **nodos** de todas las **personas** de la red.

**MATCH (a:Persona )**  
**RETURN a**

## Visualización Tabular

```
{
  "nombre": "Gustavo",
  "email": "gustavo.corrado@gmail.com",
  "fechanac": "01/08/1966",
  "apellido": "Corrado",
  "pais": "Argentina"
}
```

```
{
  "nombre": "Analía",
  "email": "adiaz@hotmail.com",
  "fechanac": "10/12/1978",
  "apellido": "Díaz",
  "pais": "Argentina"
}
```

```
{
  "nombre": "Mariana",
  "email": "mariana@yahoo.com",
  "fechanac": "31/10/1990",
  "apellido": "Dominguez",
  "pais": "Chile"
}
```



# Consultar Nodos

## Ejemplo 1

- Obtener los **nodos** de todas las **personas** de la red.

**MATCH ( a:Persona )**

**RETURN a**

## Visualización Textual

```
"a"
{
  "nombre": "Gustavo", "email": "gustavo.corrado@gmail.com", "fechanac": "01/08/1966", "apellido": "Corrado", "pais": "Argentina"
}
{
  "nombre": "Analía", "email": "adiaz@hotmail.com", "fechanac": "10/12/1978", "apellido": "Díaz", "pais": "Argentina"
}
{
  "nombre": "Mariana", "email": "mariana@yahoo.com", "fechanac": "31/10/1990", "apellido": "Dominguez", "pais": "Chile"
}
```



# Consultar Nodos

## Ejemplo 1

- Obtener los **nodos** de todas las **personas** de la red.

**MATCH (a:Persona )**  
**RETURN a**

## Visualización Código

Server version	Neo4j/3.5.8
Server address	localhost:7687
Query	MATCH (a:Persona) RETURN a
Summary ▶	{ "statement": { "text": "MATCH (a:Persona)\nRETURN a", ...
Response ▼	[ { "keys": [ "a" ], "length": 1, "_fields": [ { "identity": { "low": 0, "high": 0



# Consultar Nodos

## Ejemplo 2

- Obtener el **nodo** correspondiente a la persona de **apellido Dominguez**.

```
MATCH (a:Persona{apellido: "Dominguez"})  
RETURN a
```

Se retorna el objeto persona, por lo que puede ser **visualizado** en modo **gráfico**, **table**, **text** ó **code**.

## Ejemplo 3

- Obtener el **nombre** y la fecha de **nacimiento** de la persona de apellido **Domínguez**.

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

Se retornan **propiedades** del objeto persona, **por lo que** puede ser visualizado en modo table, text ó code, pero **no en formato Graph**.



# Consultar Nodos

## Cláusula RETURN / Operador CASE

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

```
MATCH (e: Empresa)
RETURN e.rubro,
CASE e.rubro WHEN "Telefonía"
    THEN "IT"
    ELSE e.rubro
END AS rubroModificado
```

e.rubro	rubroModificado
"Telefonía"	"IT"
"Consultoría"	"Consultoría"
"Gestión de Información Farmacéutica"	"Gestión de Información Farmacéutica"
"Petrolera"	"Petrolera"
"Aviación"	"Aviación"
"Reventa"	"Reventa"
"Capacitación"	"Capacitación"
"IT"	"IT"
"Banco"	"Banco"
"Supermercado"	"Supermercado"
"Telefonía"	"IT"



# Cláusula WHERE



# Consultar Nodos

## Cláusula WHERE

- Agrega restricciones a los patrones en la cláusula MATCH.

```
MATCH (a:Persona)
WHERE id(a)=1
RETURN a.nombre, a.apellido, a.fechanac
```

La búsqueda de Nodos por id puede realizarse con la función id.

Neo4J **reutiliza los id's** cuando un nodo o una relación es borrada. Por lo que **es conveniente agregar una propiedad idPersona** generada por la aplicación.

## Visualización

	a.nombre	a.apellido	a.fechanac
Table	"Analía"	"Díaz"	"10/12/1978"





# Consultar Nodos

## Clausula WHERE - Operadores AND OR XOR y NOT

```
MATCH (n:Persona)
WHERE n.nombre = 'Gustavo'
XOR (n.país = "Chile" AND n.nombre = 'Mariana')
OR NOT (n.nombre = 'Claudio' OR n.nombre = 'Jorge')
RETURN n.nombre, n.país
```

	n.nombre	n.país
Table	"Gustavo"	"Argentina"
Text	"Analía"	"Argentina"
	"Mariana"	"Chile"
	"Mario"	"Argentina"
Code	"Natalia"	"Argentina"
	"Eduardo"	"Chile"



# Consultar Nodos

**Cláusula WHERE - Operadores =, >, <, >=, <=, <>**

```
MATCH (n:Empresa)
WHERE n.nombre <> "CSF"
AND toInteger(n.id) >= 5 AND toInteger(n.id) < 11
RETURN n.nombre, n.rubro, n.id
```

	n.nombre	n.rubro	n.id
Table	"Lan"	"Aviación"	"5"
Text	"Gomez&Asoc"	"Reventa"	"6"
	"Alteon"	"Capacitación"	"7"
	"IBM"	"IT"	"8"
Code	"BancoNación"	"Banco"	"9"
	"Carrefour"	"Supermercado"	"10"



# Consultar Nodos

## Cláusula WHERE - Operador EXISTS

```
MATCH (n:Empresa{nombre:"IBM"})  
SET n.cantEmpleados = 10000  
RETURN n.nombre, n.cantEmpleados
```

```
MATCH (n:Empresa)  
WHERE EXISTS (n.cantEmpleados)  
RETURN n.nombre, n.cantEmpleados
```

Agregamos un atributo **cantEmpleados** a un nodo

 Table	n.nombre	n.cantEmpleados
	"IBM"	10000



# Consultar Nodos

```
MATCH (n:Empresa{nombre:"IBM"})
SET n.cantEmpleados=10000
RETURN n.nombre,n.cantEmpleados
```

Agregamos un atributo ***cantEmpleados*** a un nodo

```
MATCH (n:Empresa)
WHERE NOT EXISTS(n.cantEmpleados)
RETURN n.nombre, n.cantEmpleados
```

```
MATCH (n:Empresa)
WHERE n.cantEmpleados IS NULL
RETURN n.nombre, n.cantEmpleados
```

	n.nombre	n.cantEmpleados
Table	"Telefónica de Argentina"	null
Text	"Accenture"	null
	"CSF"	null
Code	"Chevron"	null
	"Lan"	null
	"Gomez & Asoc"	null
	"Alteon"	null
	"Banco Nación"	null
	"Carrefour"	null
	"Claro"	null



# Cláusula ORDER BY



# Consultar Nodos

## Cláusula ORDER BY

Debe estar después del WHERE o del WITH, si estuviesen.

- Obtener los nodos de todas las personas retornando nombre, apellido y fecha de nacimiento, **ordenado** por apellido **ASCENDENTE** (default).

```
MATCH (a:Persona)
RETURN a.nombre,
       a.apellido,
       a.fechanac
ORDER BY a.apellido
```

	a.nombre	a.apellido	a.fechanac
Table	"Gustavo"	"Corrado"	"01/08/1966"
Text	"Mariana"	"Dominguez"	"31/10/1990"
	"Analía"	"Díaz"	"10/12/1978"
Code	"Natalia"	"Ferreira"	"07/04/1972"
	"Eduardo"	"García"	"23/01/1985"
	"Bibiana"	"González"	"09/11/1974"
	"Jorge"	"Lupis"	"27/09/1980"
	"Mario"	"López"	"11/02/1970"
	"Verónica"	"Mendez"	"28/02/1968"
	"Claudio"	"Pereyra"	"18/05/1993"



# Consultar Nodos

## Cláusula ORDER BY

- Obtener los nodos de todas las personas retornando nombre, apellido y fecha de nacimiento, **ordenado** por apellido **DESCENDENTE**.

```
MATCH (a:Persona)
RETURN a.nombre,
       a.apellido,
       a.fechanac
ORDER BY a.apellido DESC
```

	a.nombre	a.apellido	a.fechanac
Table	"Claudio"	"Pereyra"	"18/05/1993"
Text	"Verónica"	"Mendez"	"28/02/1968"
	"Mario"	"López"	"11/02/1970"
	"Jorge"	"Lupis"	"27/09/1980"
Code	"Bibiana"	"González"	"09/11/1974"
	"Eduardo"	"García"	"23/01/1985"
	"Natalia"	"Ferreira"	"07/04/1972"
	"Analía"	"Díaz"	"10/12/1978"
	"Mariana"	"Dominguez"	"31/10/1990"
	"Gustavo"	"Corrado"	"01/08/1966"



# Cláusulas LIMIT y SKIP





# Consultar Nodos

## Cláusula LIMIT

- Limita la cantidad de nodos a enviar, del Conjunto Resultante.
- Obtener información de **5 personas** retornando nombre, apellido y fecha de nacimiento, **ordenado** por apellido **ASCENDENTE**.

**MATCH (a:Persona)**

**RETURN a.nombre, a.apellido, a.fechanac**

**ORDER BY a.apellido**

**LIMIT 5**

	a.nombre	a.apellido	a.fechanac
Table	"Gustavo"	"Corrado"	"01/08/1966"
Text	"Mariana"	"Dominguez"	"31/10/1990"
	"Analía"	"Díaz"	"10/12/1978"
	"Natalia"	"Ferreira"	"07/04/1972"
Code	"Eduardo"	"García"	"23/01/1985"



# Consultar Nodos

## Cláusula SKIP

- Salta la cantidad de nodos indicada sin enviar al Conjunto Resultante.
- Obtener información de las personas retornando nombre, apellido y fecha de nacimiento **saltando las 3 primeras, ordenado** por apellido **ASCENDENTE**.

```
MATCH (a:Persona)
RETURN a.nombre,
       a.apellido,
       a.fechanac
ORDER BY a.apellido
SKIP 3
```

	a.nombre	a.apellido	a.fechanac
→	"Gustavo"	"Corrado"	"01/08/1966"
→	"Mariana"	"Dominguez"	"31/10/1990"
→	"Analía"	"Díaz"	"10/12/1978"
	"Natalia"	"Ferreira"	"07/04/1972"
	"Eduardo"	"García"	"23/01/1985"
	"Bibiana"	"González"	"09/11/1974"
	"Jorge"	"Lupis"	"27/09/1980"
	"Mario"	"López"	"11/02/1970"
	"Verónica"	"Mendez"	"28/02/1968"
	"Claudio"	"Pereyra"	"18/05/1993"



# Consultar Nodos

## Cláusula SKIP

- Salta una cantidad de nodos aleatoria, enviando al Conjunto Resultante los nodos siguientes.

```
MATCH (a:Persona)
RETURN a.apellido
ORDER BY a.apellido
SKIP toInteger(10*rand())
```

SKIP 3	SKIP 2	SKIP 1
a.apellido	a.apellido	a.apellido
"Ferreira"	"Díaz"	"Dominguez"
"García"	"Ferreira"	"Díaz"
"González"	"García"	"Ferreira"
"Lupis"	"González"	"García"
"López"	"Lupis"	"González"
"Mendez"	"López"	"Lupis"
"Pereyra"	"Mendez"	"López"
	"Pereyra"	"Mendez"
		"Pereyra"



# Paginados por BDs



# Consultar Nodos

## Paginado por Base de Datos

- Salta una cantidad de nodos controlada, paginando el resultado a enviar al Conjunto Resultante.

```
MATCH (a:Persona)
RETURN a.apellido
ORDER BY a.apellido
SKIP toInteger(pagina*5)
LIMIT 5
```

Página 0

	a.apellido
Tabla	"Corrado"
Text	"Dominguez"
Text	"Díaz"
Text	"Ferreira"
Text	"García"

Página 1

	a.apellido
Tabla	"González"
Text	"Lupis"
Text	"López"
Text	"Mendez"
Text	"Pereyra"

Página 2

	(no changes, no records)
Tabla	
Text	
Text	
Text	
Text	

a.apellido

"Corrado"

"Dominguez"

"Díaz"

"Ferreira"

"García"

"González"

"Lupis"

"López"

"Mendez"

"Pereyra"



DBlandIT



# Consultar Nodos

## Paginado por Base de Datos con parámetro string para definir página

- Salta una cantidad de nodos controlada, paginando el resultado a enviar al Conjunto Resultante.

```
:param pag => '0'
```

```
MATCH (a:Persona)
```

```
RETURN a.nombre, a.apellido, a.fechanac
```

```
ORDER BY a.apellido
```

```
SKIP toInteger($pag) * 5
```

```
LIMIT 5
```

```
$ :param pag => '0'
```

```
{  
  "pag": "0"  
}
```

```
$ MATCH (a:Persona) RETURN a.nombre, a.apellido, a.fechanac ORDER BY a.apellido SKIP toInteger($pag)*5 LIMIT 5
```

	a.nombre	a.apellido	a.fechanac
Tabla	"Gustavo"	"Corrado"	"01/08/1986"
A	"Mariana"	"Dominguez"	"31/10/1990"
Text	"Analía"	"Díaz"	"10/12/1978"
Code	"Natalia"	"Ferreira"	"07/04/1972"
	"Eduardo"	"García"	"23/01/1985"



DBlandIT



# Consultar Nodos

## Paginado por Base de Datos con parámetro integer para definir página

- Salta una cantidad de nodos controlada, paginando el resultado a enviar al Conjunto Resultante.

```
:param pag => 0
```

```
MATCH (a:Persona)
RETURN a.nombre, a.apellido, a.fechanac
ORDER BY a.apellido
SKIP $pag*5
LIMIT 5
```

```
$ :param pag => 0
```

```
{
  "pag": 0
}
```

```
$ MATCH (a:Persona) RETURN a.nombre, a.apellido, a.fechanac ORDER BY a.apellido SKIP $pag*5 LIMIT 5
```

	a.nombre	a.apellido	a.fechanac
Tabla	"Gustavo"	"Corrado"	"01/08/1966"
Text	"Mariana"	"Dominguez"	"31/10/1990"
	"Analía"	"Díaz"	"10/12/1978"
	"Natalia"	"Ferreira"	"07/04/1972"
Code	"Eduardo"	"García"	"23/01/1985"



# Ver todos los parámetros creados en la sesión

---

**:params** nos permite consultar los parámetros definidos durante la sesión del browser.

```
$ :params

$ :params
{
  "pag": 0.0,
  "size": 5.0
}

See :help param for usage of the :param command.
```





# Consultar Relaciones y Nodos



# Consultar Relaciones y Nodos

## Consultar Relaciones y Nodos

- Consultar una relación.

```
MATCH (n:Etiqueta{propiedades}) -[r:Relacion{propiedades}] -> (m)
WHERE condición
```

## Ejemplo 1 –Consulta de Nodos Relacionados

- Obtener la lista de empresas en las que trabajó Domínguez.

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

```
{
  "rubro": "Telefonia",
  "ubicacion": "Argentina",
  "id": "11",
  "nombre": "Claro",
  "tamano": "Grande"
}
```

```
{
  "rubro": "Aviación",
  "ubicacion": "Chile",
  "id": "5",
  "nombre": "Lan",
  "tamano": "Mediana"
}
```



# Consultar Relaciones y Nodos

## Consultar Relaciones y Nodos

- Consultar una relación.

```
MATCH (n:Etiqueta{propiedades}) -[r:Relacion{propiedades}] -> (m)
WHERE condición
```

## Ejemplo 1 – Consulta de Nodos Relacionados

- Obtener la lista de empresas en las que trabajó Domínguez.

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



# Consultar Relaciones y Nodos

## Ejemplo 2 – Consulta de nodos relacionados

- Obtener la lista **nombres** y **rubros** de las **empresas** en las que trabajó Dominguez.

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

\$ MATCH (a:Persona{apellido:"Dominguez"})-[r:TRABAJO]->(b:Empresa) RETURN b.nombre...										
<div><div><div></div></div><div>Table</div></div>										
<div><div><div></div></div><div>Text</div></div>										
<div><div><div></div></div><div>Code</div></div>										
<table><thead><tr><th>b.nombre</th><th>b.rubro</th></tr></thead><tbody><tr><td>"Claro"</td><td>"Telefonía"</td></tr><tr><td>"Lan"</td><td>"Aviación"</td></tr></tbody></table>					b.nombre	b.rubro	"Claro"	"Telefonía"	"Lan"	"Aviación"
b.nombre	b.rubro									
"Claro"	"Telefonía"									
"Lan"	"Aviación"									

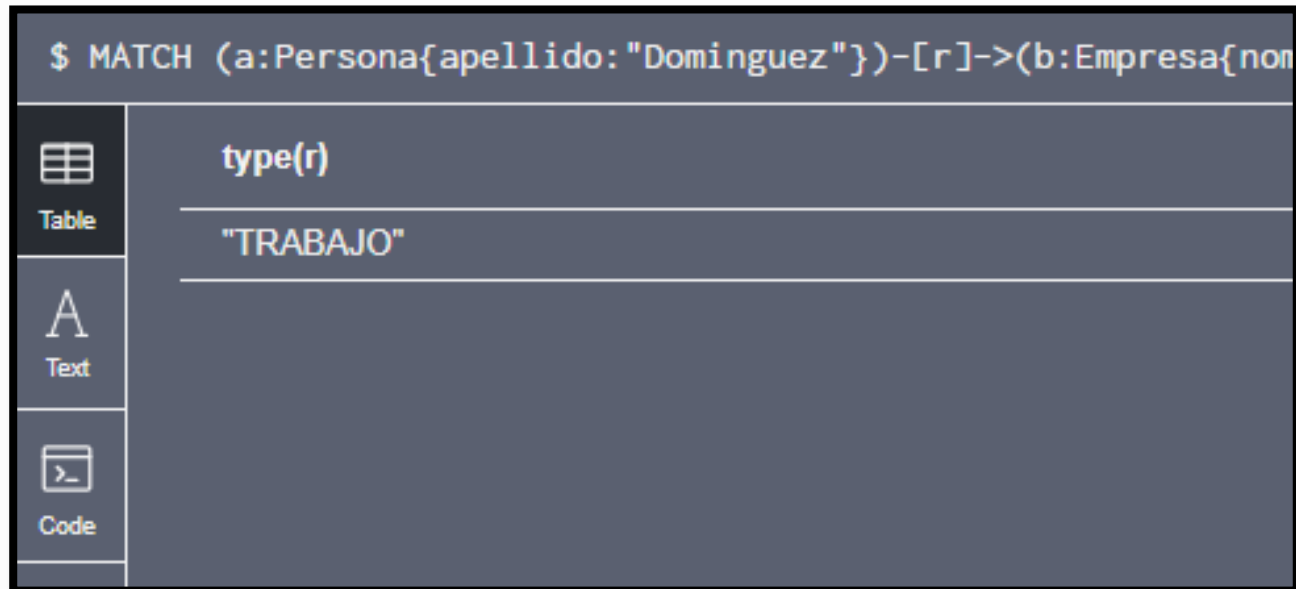


# Consultar Relaciones y Nodos

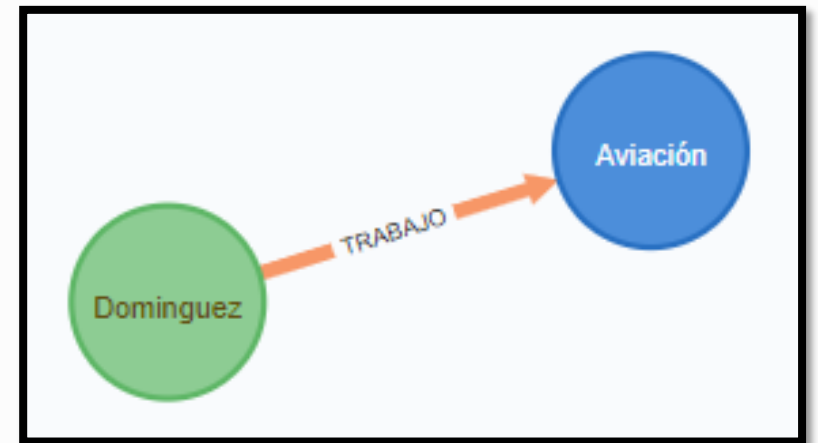
## Ejemplo 3 – Función type()

- Obtener el **tipo de relación** que tuvo **Dominguez** con la empresa **Lan**

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



type(r)
"TRABAJO"



# Consultar Relaciones y Nodos

## Cláusula RETURN / Operador DISTINCT

- La consulta devuelve los **países** a los que pertenecen las personas de la Base.

```
MATCH (n:Persona)  
RETURN DISTINCT n.pais
```

\$ MATCH (n:Persona) RETURN DISTINCT n.pais	
Table	n.pais
	"Argentina"
	"Chile"
	"EstadosUnidos"
	"España"
Text	
Code	

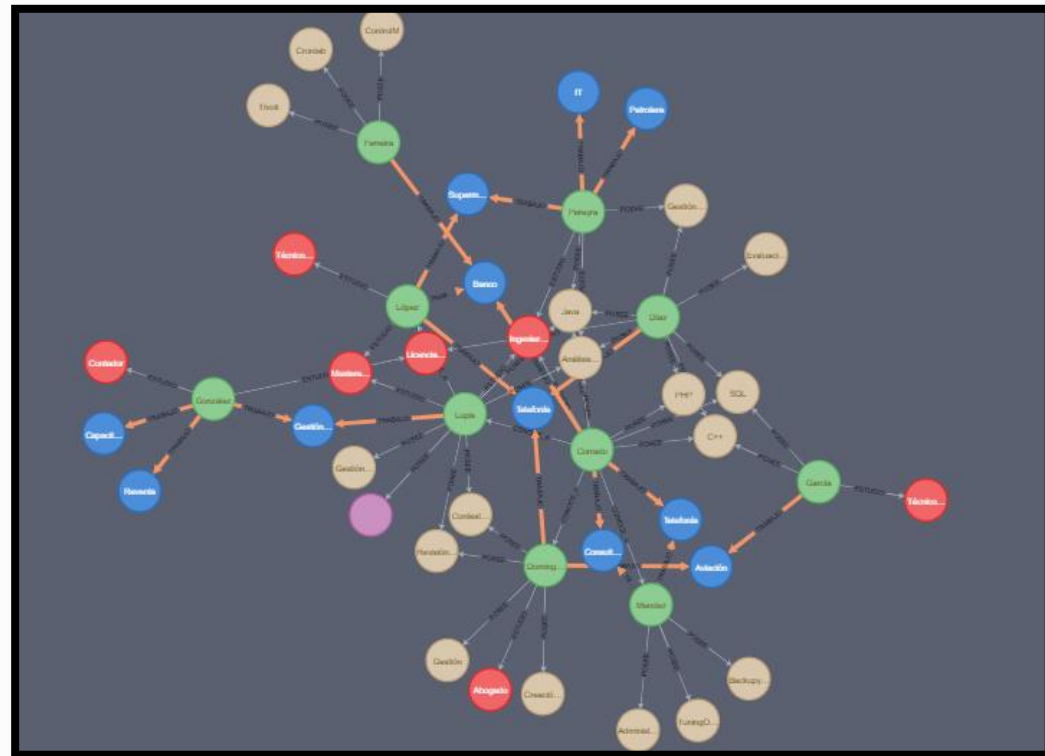


# Consultar Relaciones y Nodos

## Clausula RETURN / Operador \*

- El **return** \* devuelve todos los atributos del nodo o las propiedades de las relaciones involucradas.

```
MATCH (n:Persona)  
  -[r]->  
    (m)  
RETURN *
```



# Consultar Relaciones y Nodos

## Operador IN

Buscamos todos los nodos de Persona cuyo nombre pertenezca al criterio de búsqueda que establezcamos.

```
MATCH (n:Persona)
WHERE n.nombre IN ["Jorge","Mariana"]
RETURN n.nombre, n.apellido
```

n.nombre	n.apellido
"Mariana"	"Dominguez"
"Jorge"	"Lupis"

## Operador IS NULL

Buscamos todas las personas con edad de 27 años o sin atributo edad o edad nula.

```
MATCH (n:Persona)
WHERE n.edad = 27 OR n.edad IS NULL
RETURN n.nombre, n.apellido, n.edad
```

n.nombre	n.apellido	n.edad
"Gustavo"	"Corrado"	null
"Analía"	"Díaz"	null
"Mariana"	"Dominguez"	null
"Claudio"	"Pereyra"	null





# Consultar Relaciones y Nodos

## Cláusula WITH

La cláusula WITH separa explícitamente partes del query, permitiendo declarar qué identificadores pasar hacia la parte siguiente. Un uso habitual es para **limitar la cantidad de resultados que pasarán a otra cláusula MATCH**. Otro uso es para **filtrar valores agregados**.

## Ejemplo:

- Obtener la lista de personas y cantidad de conocimientos registrados de cada una de ellas, para las que registraron más de 3 conocimientos.

**MATCH (a:Persona)-[:POSEE]->(c:Conocimiento)**

**WITH a, count(c) AS conocimientos**

**WHERE conocimientos > 3**

**RETURN a.apellido,a.nombre,konocimientos**

a.apellido	a.nombre	conocimientos
"Corrado"	"Gustavo"	5
"Díaz"	"Analía"	7
"Dominguez"	"Mariana"	4
"Lupis"	"Jorge"	5

*También se puede combinar con ORDER BY, SKIP y LIMIT.*

*Otro uso puede ser para separar cláusulas de lectura y escritura del grafo.*



# Consultar Relaciones y Nodos

## Cláusula WHERE

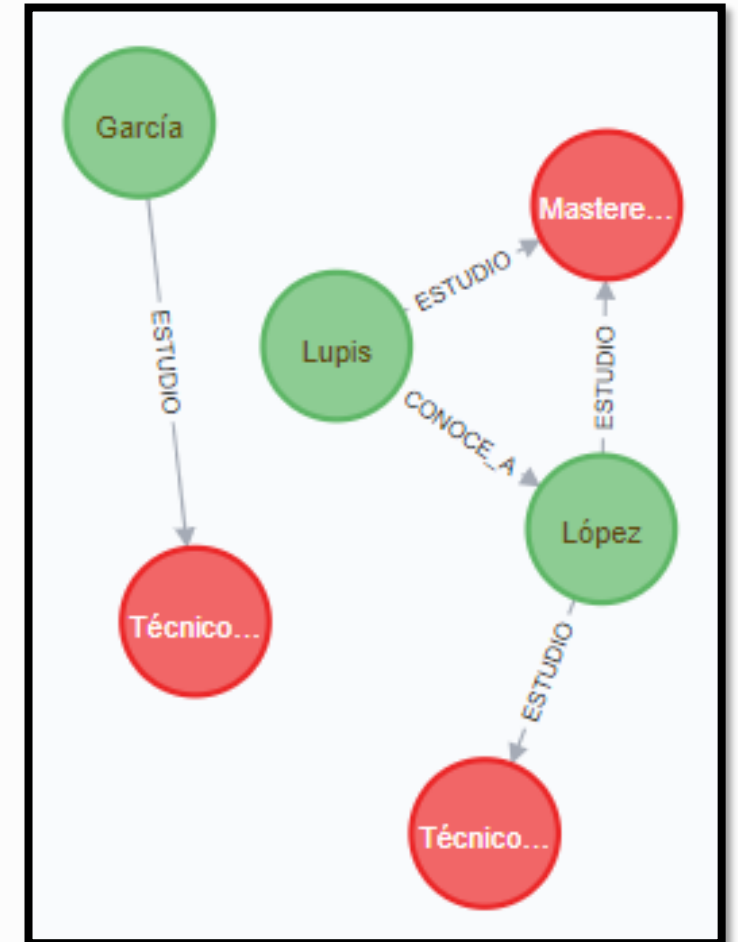
**WHERE** propiedad **operador** valor

Los predicados se utilizan para filtrar. WHERE siempre es parte de otra cláusula, a la cual modifica.

### Ejemplo1:

- 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
```

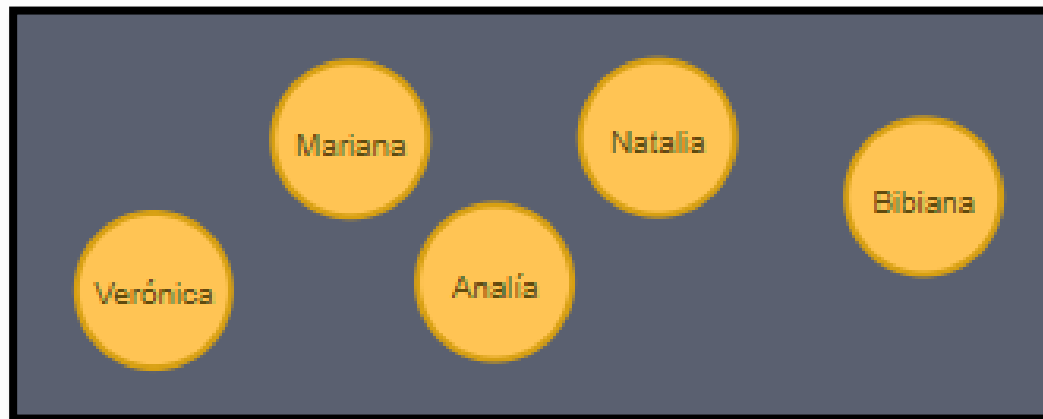


# Consultar Relaciones y Nodos

## Ejemplo 2:

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

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



# Consultar Relaciones y Nodos

## Operadores para Matcheo de Strings

**STARTS WITH - ENDS WITH - CONTAINS**

### Ejemplo1 - STARTS WITH

- Buscamos todos los nodos de **Conocimiento** cuyo nombre **comienza** con "Ges"

```
MATCH (c:Conocimiento)
WHERE c.nombre STARTS WITH "Ges"
RETURN c.nombre
```

c.nombre
"Gestión de proyectos"
"Gestión"
"Gestión de recursos"



# Consultar Relaciones y Nodos

## Operadores para Matcheo de Strings

**STARTS WITH - ENDS WITH - CONTAINS**

### Ejemplo 2 - ENDS WITH

- Buscamos todos los nodos de **Conocimiento** cuyo nombre **termina** con "os"

```
MATCH (c:Conocimiento)
WHERE c.nombre ENDS WITH "os"
RETURN c.nombre
```

c.nombre
"Gestión de proyectos"
"Revisión y elaboración de contratos"
"Elaboración de plan de negocios"
"Evaluación de proyectos"
"Gestión de recursos"



# Consultar Relaciones y Nodos

## Operadores para Matcheo de Strings

**STARTS WITH - ENDS WITH - CONTAINS**

### Ejemplo 3 – CONTAINS

- Buscamos todos los nodos de **Conocimiento** cuyo nombre **contenga** "de"

```
MATCH (c:Conocimiento)
WHERE c.nombre CONTAINS "de"
RETURN c.nombre
```

c.nombre
"Gestión de proyectos"
"Revisión y elaboración de contratos"
"Creación de sociedades"
"Contestación de demandas"
"Evaluación de proveedores"
"Elaboración de plan de negocios"
"Evaluación de proyectos"
"Elaboración de balances"
"Gestión de recursos"



# Consultar Relaciones y Nodos

## Expresiones Regulares Case Insensitive

- Buscamos todos los nodos de Persona cuyo nombre **comience** con "JOR" (sin tener en cuenta mayúsculas / minúsculas).

```
MATCH (n:Persona)
WHERE n.nombre =~ '(?i)JOR.*'
RETURN n.nombre
```

- Case Sensitive por omisión

```
MATCH (n:Persona)
WHERE n.nombre =~ 'JOR.*'
RETURN n.nombre
```

```
{
  "nombre": "Jorge",
  "email": "jlup@gmail.com",
  "fechanac": "27/09/1980",
  "apellido": "Lupis",
  "pais": "Argentina"
}
```

(no changes, no records)



# Consultar Relaciones y Nodos

## Expresiones Regulares Carácter de Escape

Los caracteres "." y "\*" son Wildcards con un significado especial en las expresiones regulares. Si quiero buscar dicho caracter debo utilizar el caracter de escape "\\".

```
MATCH (n:Persona)  
WHERE n.email =~ '.*\\.com'  
RETURN n.nombre, n.apellido, n.email
```

n.nombre	n.apellido	n.email
"Gustavo"	"Corrado"	"gustavo.corrado@gmail.com"
"Analía"	"Díaz"	"adiaz@hotmail.com"
"Mariana"	"Dominguez"	"mariana@yahoo.com"
"Mario"	"López"	"mario.lopez@gmail.com"
"Natalia"	"Ferreira"	"nf@hotmail.com"
"Jorge"	"Lupis"	"jlup@gmail.com"





# Consultar Relaciones y Nodos

## Rango Simple <, <=, >, >=

- Buscamos todas las Personas con nombre Mayor o igual a "Claudio"

```
MATCH (n:Persona)
WHERE n.nombre >= "Claudio"
RETURN n.nombre
ORDER BY n.nombre
```

- Buscamos todas las Personas con nombre Menor a "Claudio"

```
MATCH (n:Persona)
WHERE n.nombre < "Claudio"
RETURN n.nombre
ORDER BY n.nombre
```

n.nombre
"Analía"
"Bibiana"
"Claudio"
"Eduardo"
"Gustavo"
"Jorge"
"Mariana"
"Mario"
"Natalia"
"Verónica"



# Consultar Relaciones y Nodos

## Rango Compuesto

- Buscamos las personas que conocen a otras personas desde **antes de 1994 inclusive** y **después de 1990**, informando apellido de ambos y fecha desde que se conocen.

```
MATCH (n:Persona) - [r:CONOCE_A] - (b:Persona)
WHERE r.fechad >= "1990" AND r.fechad <= "1994"
RETURN n.nombre, n.apellido, r.fechad, b.nombre, b.apellido
```

n.nombre	n.apellido	r.fechad	b.nombre	b.apellido
"Gustavo"	"Corrado"	"1990"	"Mariana"	"Dominguez"
"Gustavo"	"Corrado"	"1994"	"Jorge"	"Lupis"
"Mariana"	"Dominguez"	"1990"	"Gustavo"	"Corrado"
"Mario"	"López"	"1994"	"Jorge"	"Lupis"
"Jorge"	"Lupis"	"1994"	"Mario"	"López"
"Jorge"	"Lupis"	"1994"	"Gustavo"	"Corrado"



# Consultar Relaciones y Nodos

## Filtros en Propiedades con Nombre Dinámico de una Relación

- Buscamos las personas que conocen a otras personas desde antes del 1999 inclusive, informando apellido de ambos, motivo y fecha desde que se conocen.

```
WITH 'fechad' AS nombrePropiedad
MATCH (p:Persona) -[r:CONOCE_A]-> (o:Persona)
WHERE r[nombrePropiedad] <= "1999"
RETURN p.apellido, r.motivo, r.fechad, o.apellido
```

p.apellido	r.motivo	r.fechad	o.apellido
"Corrado"	"Amistad"	"1990"	"Dominguez"
"Corrado"	"Estudio"	"1994"	"Lupis"
"Lupis"	"Estudio"	"1994"	"López"



# Consultar Relaciones y Nodos

## Filtros en Propiedades de una Relación

- Buscamos las personas que conocen a otras personas **desde antes** del 1999 **inclusive**, informando apellido de ambos, motivo y fecha desde que se conocen.

```
MATCH (p:Persona) -[r:CONOCE_A] -> (o:Persona)
WHERE r.fechad <= "1999"
RETURN p.apellido, r.motivo, r.fechad, o.apellido
```

p.apellido	r.motivo	r.fechad	o.apellido
"Corrado"	"Amistad"	"1990"	"Dominguez"
"Corrado"	"Estudio"	"1994"	"Lupis"
"Lupis"	"Estudio"	"1994"	"López"



# Consultar Relaciones y Nodos

## Comentarios en una Consulta

```
1 WITH 'FECHAD' AS nombrePropiedad
2 // En la operación WITH se relaciona un valor con un alias y lo pasa al MATCH.
3 MATCH (p:Persona)-[r:CONOCE_A]->(o:Persona)
4 WHERE r[toLower(nombrePropiedad)] <= "1999"
5 // La función toLower pasa a minúsculas el valor de la variable nombrePropiedad
6 RETURN p, r, o
```

```
WITH 'FECHAD' AS nombrePropiedad MATCH (p:Persona)-[r:CONOCE_A]->(o:Persona) WHERE r[toLower(
```

\*(4)

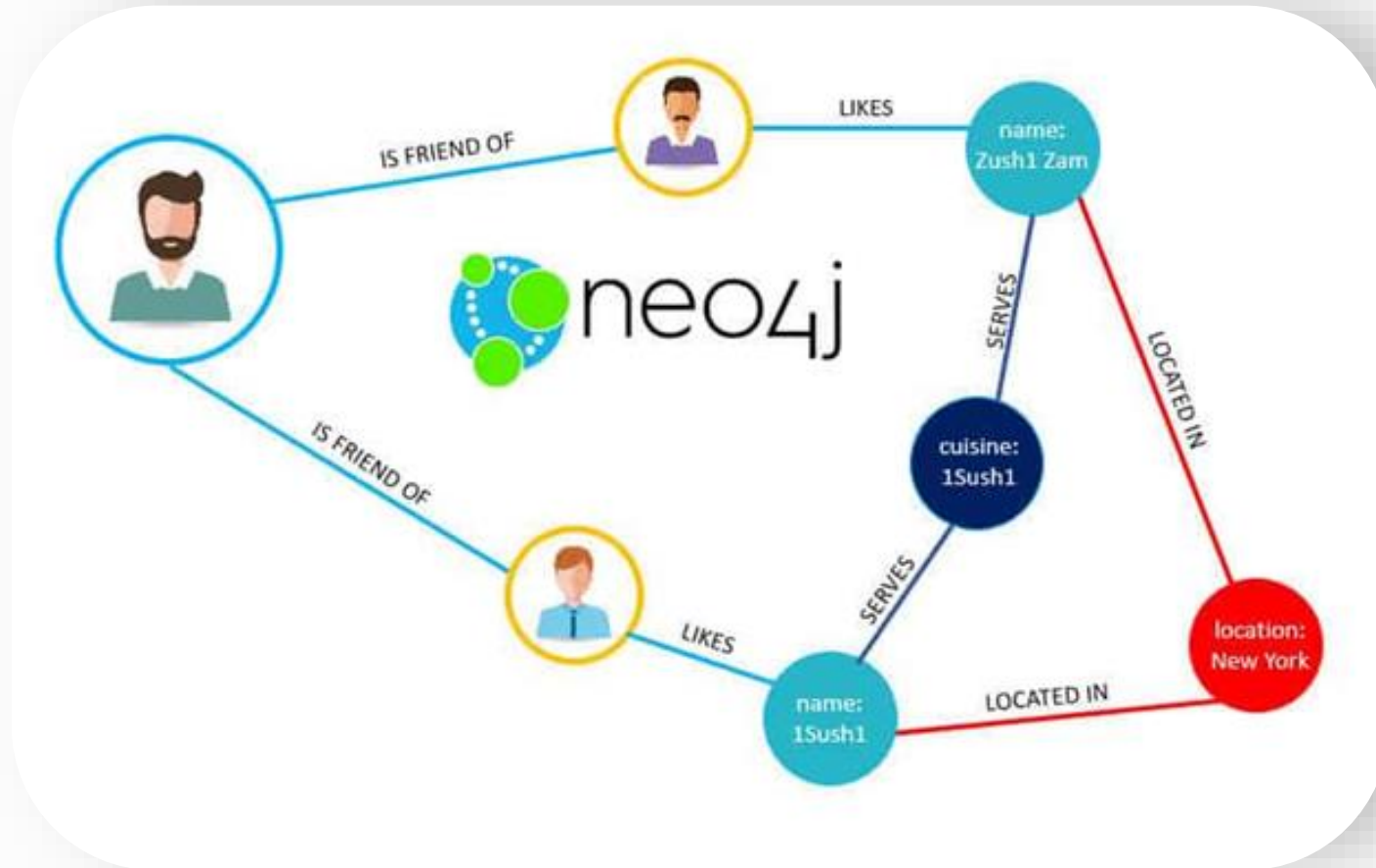
Persona(4)

\*(3)

CONOCE\_A(3)



# Consultar Relaciones y Nodos



# Eliminar Nodos



# Eliminar Nodos

Eliminar nodos que responden a cierto criterio y no tienen relaciones


```
CREATE (a:Persona{nombre:"Javier", apellido:"Zeisfer"})
```

```
MATCH (a:Persona{nombre:"Javier", apellido:"Zeisfer"}) RETURN  
id(a)
```

id(a)
56



```
MATCH (a:Persona)  
WHERE id(a)=56  
DELETE a
```

\$ MATCH (a:Persona) WHERE id(a)=56 DELETE a	
	Deleted 1 node, completed after 2 ms.





# Eliminar Nodos

---

**Eliminar nodos que responden a cierto criterio y no tienen relaciones**

```
CREATE (a:Persona{nombre:"Javier", apellido:"Zeisfer"})
```

```
MATCH (a:Persona{nombre:"Javier", apellido:"Zeisfer"})
```

```
DELETE a
```



DBlandIT



# Eliminar Nodos

Eliminar nodos que responden a cierto criterio y tienen relaciones

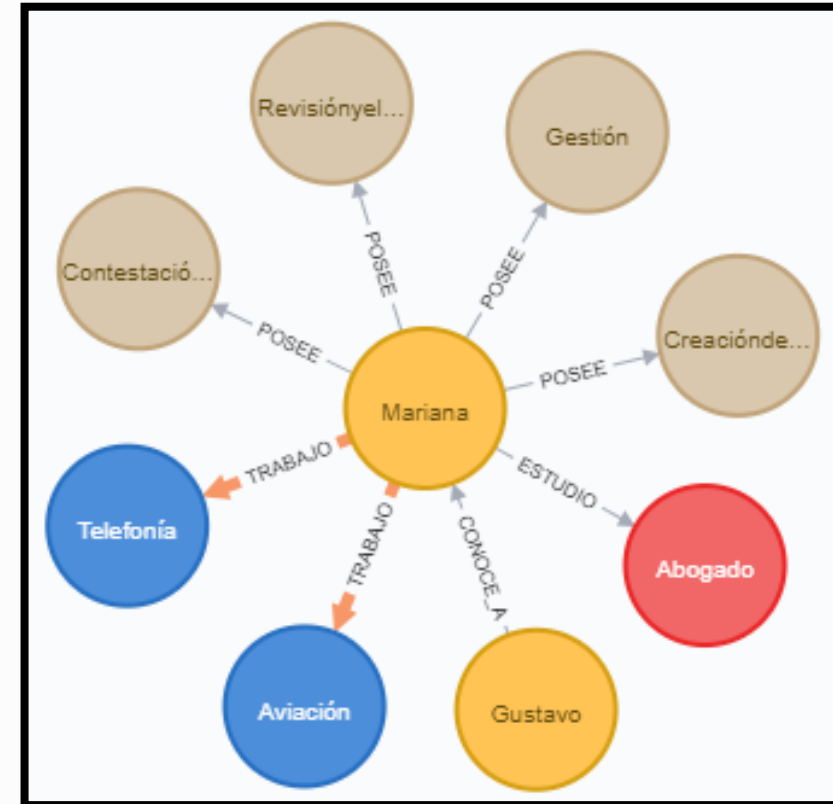
```
MATCH (a:Persona{nombre:"Mariana"})  
RETURN a.nombre, a.apellido, id(a)
```

a.nombre	a.apellido	id(a)
"Mariana"	"Dominguez"	2

```
MATCH (a:Persona) WHERE id(a)=2 DELETE a
```

ERROR

Neo.ClientError.Schema.ConstraintValidationFailed



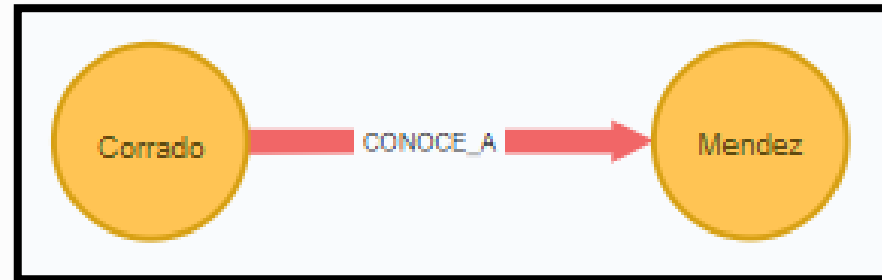
# Eliminar Relaciones



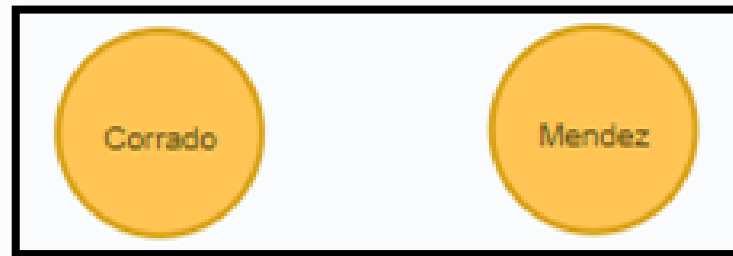
# Eliminar Relaciones

Eliminar determinadas relaciones de un nodo que cumplan con un criterio dado

```
MATCH (a:Persona{apellido:"Corrado"})-[r:CONOCE_A{motivo:"Trabajo"}]->(b)  
RETURN a,r,b
```



```
MATCH (a:Persona{apellido:"Corrado"})-[r:CONOCE_A]-()  
DELETE r
```



# Eliminar Nodos y sus Relaciones



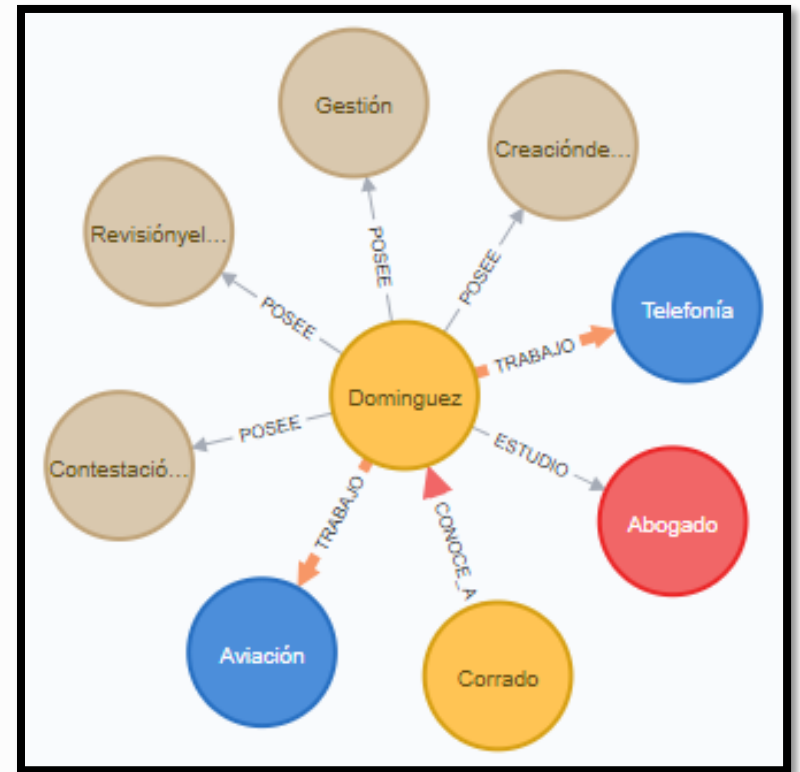
# Eliminar Nodos y sus Relaciones

**Eliminar Nodos que responden a cierto criterio y tienen relaciones entrantes o salientes**

```
MATCH (a:Persona{nombre:"Mariana"})-[r]-(b)  
RETURN a,r,b
```

```
MATCH (a:Persona{nombre:"Mariana"})-[r]-()  
DELETE a,r
```

Deleted 1 node, deleted 8 relationships, completed after 3 ms.



# Borrado de Nodos y Todas sus Relaciones

## Sintaxis

```
MATCH ...  
DETACH DELETE ...
```

## Ejemplo 1:

- Elimina todos los nodos de la Base y sus relaciones

```
MATCH (a)  
DETACH DELETE a
```

Deleted 59 nodes, deleted 69 relationships, completed after 10 ms.

**ADVERTENCIA !!!!**



**WARNING !!!!**



**AVISO !!!!**



**WARNUNG !!!!**



**AVERTISSEMENT !!!!**



**警告 !!!!**



**चेतावनी !!!!**



DBlandIT



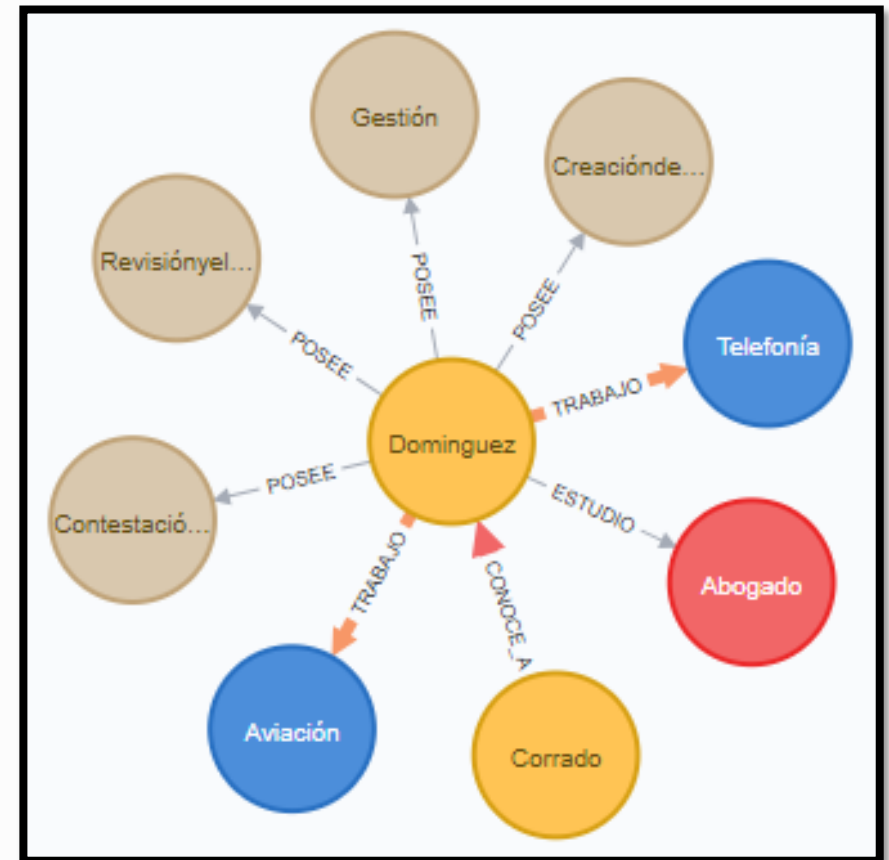
# Eliminar Nodos y todas sus Relaciones

**Eliminar Nodos que responden a cierto criterio y tienen relaciones**

```
MATCH (a:Persona{nombre:"Mariana"})  
RETURN a.nombre, a.apellido, id(a)
```

```
MATCH (a:Persona)  
WHERE id(a)=473  
DETACH DELETE a
```

Deleted 1 node, deleted 8 relationships, completed after 3 ms.







# DBlandIT

info@dblandit.com  
+54 11 3889-4009  
www.dblandit.com/

