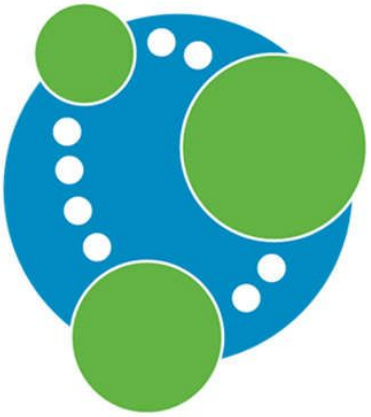
ESCUELA POLITÉCNICA SUPERIOR DE CÓRDOBA
Universidad de Córdoba

EPSC

UNIVERSIDAD DE CÓRDOBA

Neo4j



Caso Complejo nº2:

Manejo de datos de los profesores y alumnos de una Universidad

1.- Planteamiento del problema

Se quiere crear y manejar una base de datos en la que se almacenarán información de profesores y alumnos, cuyos atributos serán:

- ***nombre:*** *Nombre del profesor.*
- ***apellidos:*** *Apellidos del profesor.*
- ***dni:*** *dni del alumnos.*

Además, se trabajará con las asignaturas que imparten dichos profesores y a las cuales están matriculados los alumnos. Las asignaturas tendrán los atributos:

- ***nombre:*** *Nombre de la asignatura.*
- ***curso:*** *curso en el que se imparte.*
- ***cuatrimestre:*** *cuatrimestre en el que se imparte.*
- ***créditos:*** *créditos que tiene esa asignatura*

Por último, también trabajaremos con departamentos en los cuales , tanto profesores como alumnos pueden pertenecer a ellos.Sus atributos serán:

- ***nombre:*** *Nombre del departamento.*
- ***curso:*** *lugar en el que se encuentra el departamento.*

Entre los nodos existen las relaciones 3 tipos de relaciones:

La primera de ellas es la de Imparte , que va de los nodos profesores a los nodos Asignaturas.

La siguiente es la de Matriculado_en, que va de los Alumnos a las Asignaturas y cuyos atributos son:

- ***n_matriculas:*** *número de veces que el alumnos se ha matriculado en esa asignatura.*
- ***tiempo:*** *si está matriculado el alumno de la asignatura a tiempo parcial o completo*

Y por último, la relación Pertenece_a , referente a aquellas personas que pertenecen a un departamento. Sus atributos son:.

- ***años:*** *años que lleva esa persona en el departamento*

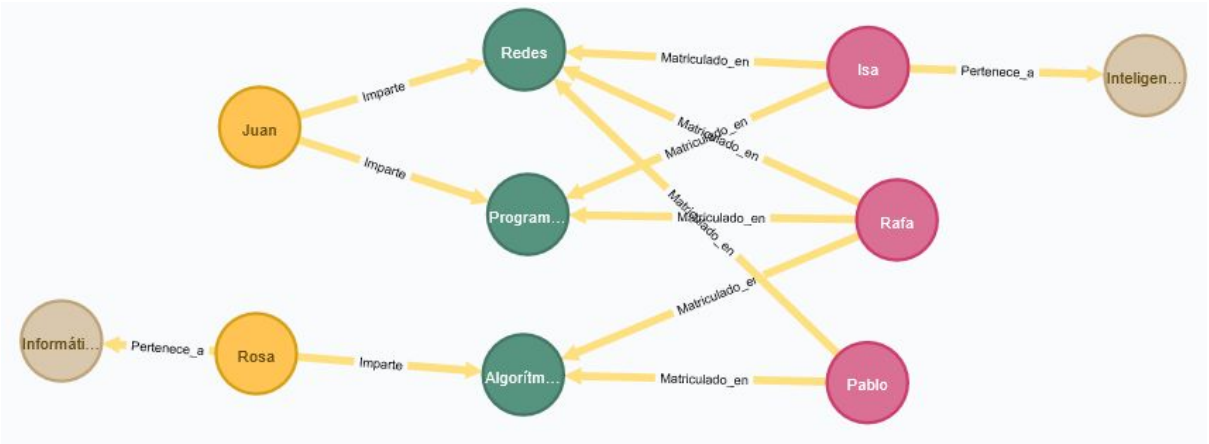
Con este planteamiento inicial, se prosigue en el desarrollo de la solución.

2.- Posible solución para el problema

Primeramente creamos la base de datos como sigue. Para ello, copiar y pegar todos los comandos de los recuadros de a continuación en una misma línea de ejecución de Neo4j. De esta forma se consigue ahorrar tiempo, ya que, en caso de hacerlo por separado se deberían crear matches antes de poder crear cada relación como es debido. Una vez ejecutados todos los comandos, podemos continuar.

```
CREATE (Juan:Persona:Profesor{nombre:'Juan',apellidos:'Torres',dni:'12345678V'})
CREATE (Rosa:Persona:Profesor{nombre:'Rosa',apellidos:'García',dni:'32145678A'})
CREATE (Isa:Persona:Alumno{nombre:'Isa',apellidos:'Jiménez',dni:'45925678Q'})
CREATE (Rafa:Persona:Alumno{nombre:'Rafa',apellidos:'Morales',dni:'21345678S'})
CREATE (Pablo:Persona:Alumno{nombre:'Pablo',apellidos:'Ruiz',dni:'13245678G'})
CREATE(PW:Asignatura{nombre:'Programación Web',curso:3,cuatri:1,créditos:6})
CREATE(Redes:Asignatura{nombre:'Redes',curso:3,cuatri:1,créditos:6})
CREATE(Algoritmica:Asignatura{nombre:'Algorítmica',curso:3,cuatri:1,créditos:6})
CREATE(AI:Departamento{nombre:'Inteligencia Artificial',lugar:'Edificio Ramón y Cajal'})
CREATE(IAN:Departamento{nombre:'Informática y Análisis Numérico',lugar:'Edificio Da  Vinci'})
CREATE (Juan)-[:Imparte]->(PW)
CREATE (Juan)-[:Imparte]->(Redes)
CREATE (Rosa)-[:Imparte]->(Algoritmica)
CREATE (Isa)-[:Matriculado_en{n_matriculas:2,tiempo:'Completo'}]->(PW)
CREATE (Isa)-[:Matriculado_en{n_matriculas:2,tiempo:'Parcial'}]->(Redes)
CREATE (Rafa)-[:Matriculado_en{n_matriculas:1,tiempo:'Parcial'}]->(PW)
CREATE (Rafa)-[:Matriculado_en{n_matriculas:2,tiempo:'Completo'}]->(Redes)
CREATE (Rafa)-[:Matriculado_en{n_matriculas:1,tiempo:'Completo'}]->(Algoritmica)
CREATE (Pablo)-[:Matriculado_en{n_matriculas:1,tiempo:'Completo'}]->(Redes)
CREATE (Pablo)-[:Matriculado_en{n_matriculas:1,tiempo:'Completo'}]->(Algoritmica)
CREATE (Rosa)-[:Pertenece_a{años:3}]->(IAN)
CREATE (Isa)-[:Pertenece_a{años:1}]->(AI)
```

Una vez creada nuestra base de datos, y si la mostramos de forma completa, obtendremos algo como sigue:



Ahora, se mostrarán una serie de consultas y modificaciones básicas de los datos, para poder manejarla de forma correcta:

1) ***Mostrar el nombre y apellidos de los alumnos matriculados de nuestra Base de Datos cuyo apellido comience por M:***

Comando a usar:

```
MATCH (p:Alumno) WHERE p.apellidos STARTS WITH 'M' return p.nombre, p.apellidos
```

Resultado esperado:

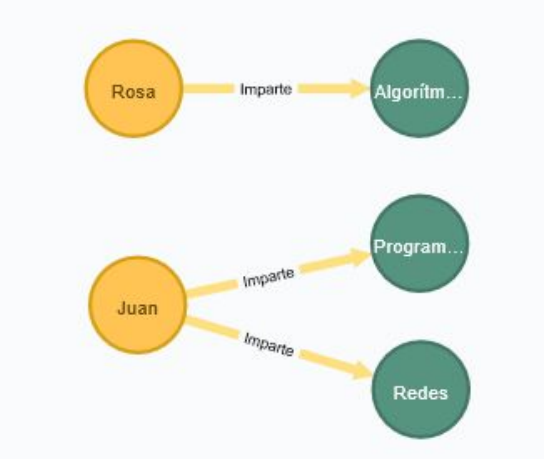
p.nombre	p.apellidos
"Rafa"	"Morales"

2) ***Mostrar primero en forma de nodos y luego en forma de tabla con sus correspondientes alias (Asignatura y Profesor) las asignaturas que hay y el nombre de los profesores que la imparten :***

Comando a usar para formato nodo:

```
MATCH relation=(p:Profesor)-[:Imparte]->(a:Asignatura) RETURN p,a
```

Resultado esperado:



Comando a usar formato tabla:

```
MATCH relation=(p:Profesor)-[:Imparte]->(a:Asignatura)  RETURN p.nombre as Profesor ,a.nombre  as Asignatura
```

Resultado esperado:

Profesor	Asignatura
"Juan"	"Redes"
"Juan"	"Programación Web"
"Rosa"	"Algorítmica"

3) Mostrar el número de asignaturas en los que el Alumno Rafa está matriculado:

Comando a usar:

```
MATCH relation=(p:Alumno)-[:Matriculado_en]->(a:Asignatura) WHERE p.nombre='Rafa' RETURN count(a)
```

Resultado esperado:

count(a)
3

4) ***Mostrar las asignaturas y el nombre de los profesores que las imparten en las que el alumno Isa está matriculado por segunda vez a tiempo parcial:***

Comando a usar:

```
MATCH relation  = (p:Profesor)-[:Imparte]->(a:Asignatura) <- [r:Matriculado_en ]
-(p2:Alumno) WHERE p2.nombre='Isa'AND r.n_matriculas=2 AND
r.tiempo='Parcial' RETURN p.nombre as Profesor , a.nombre as Asignatura
```

Resultado esperado:

Profesor	Asignatura
"Juan"	"Redes"

5) ***Mostrar el número de repetidores que hay por asignatura, es decir , los alumnos que tienen segunda matrícula y mostrar resultado con alias:***

Comando a usar:

```
MATCH relation  = (p:Alumno)- [r:Matriculado_en ]-> (a:Asignatura) WHERE
r.n_matriculas=2  RETURN a.nombre as Asignatura , count(r) as Repetidores
```

Resultado esperado:

Asignatura	Repetidores
"Programación Web"	1
"Redes"	2

6) ***Mostrar aquellas personas , ya sea alumno o profesor, que pertenezcan a un departamento desde hace más de un año y el nombre de dicho departamento***

Comando a usar:

```
MATCH relation  = (p:Persona)- [r:Pertenece_a ]-> (d:Departamento) WHERE
r.años>1  RETURN p.nombre, d.nombre
```

Resultado esperado:

p.nombre	d.nombre
"Rosa"	"Informática y Análisis Numérico"

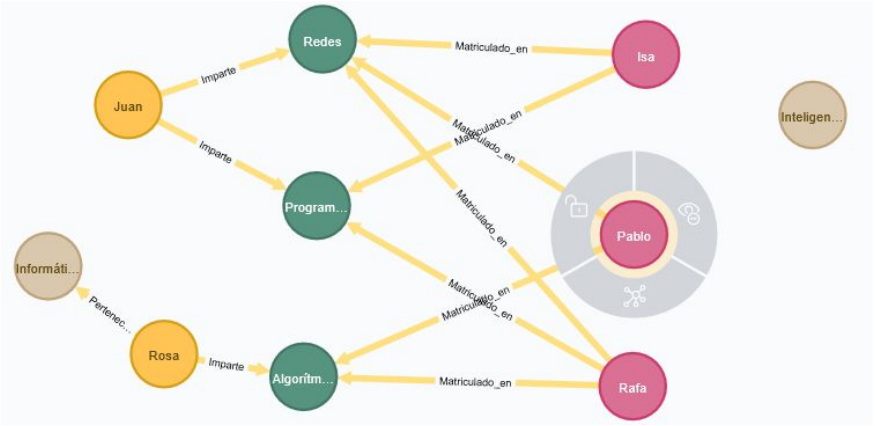
7) ***Quitar de un Departamento a aquellas personas que lleven un año o menos perteneciendo a él:***

Comando a usar:

```
MATCH relation  = (p:Persona)- [r:Pertenece_a ]-> (d:Departamento) WHERE r.años<=1  delete r
```

```
MATCH (n) return n
```

Resultado esperado:



8) ***Mostrar por cada asignatura el profesor que la imparte y el número de alumnos matriculados, ordenados por orden descendiente de matrículas:***

Comando a usar:

```
MATCH relation  = (p:Profesor)-[:Imparte]->(a:Asignatura) <- [r:Matriculado_en ] -(p2:Alumno)  RETURN p.nombre , a.nombre, count(p2) ORDER BY count(p2) DESC
```

Resultado esperado:

p.nombre	a.nombre	count(p2)
"Juan"	"Redes"	3
"Juan"	"Programación Web"	2
"Rosa"	"Algorítmica"	2

9) Mostrar por cada asignatura si tiene más de dos alumnos matriculados o no :

Comandos a usar:

```
MATCH relation = (p:Alumno)-[r:Matriculado_en]->(a:Asignatura) RETURN
a.nombre, count(p)>2 as matriculados_mayor_2
```

Resultado esperado:

a.nombre	matriculados_mayor_2
"Programación Web"	false
"Redes"	true
"Algorítmica"	false