# *Trabajo Práctico Especial*

## Ejercicio 1

Indicar Verdadero o Falso

**1.1)** Si en Neo4J pregunto por una propiedad no existente obtengo un error.

Rta

Falso. Neo4j es flexible en su esquema. Si se consulta una propiedad que no existe en un nodo o relación, Neo4j devuelve null para esa propiedad, no un error.

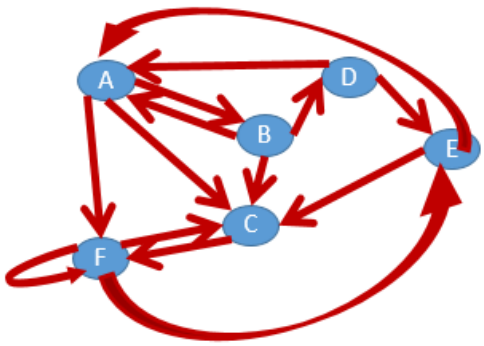
**1.2)** Neo4J permite que un mismo eje tenga asociados múltiples "types".

Rta

Falso. Cada relación (eje) en Neo4j debe tener exactamente un tipo de relación. Es posible tener múltiples relaciones entre dos nodos, cada una con su propio tipo, pero una única instancia de relación no puede tener múltiples tipos.

## Ejercicio 2

Dado el mismo grafo del TP 3, ejercicio 7 (los types no se colocaron en el dibujo porque son todos “follows”)




El cual, como ya sabemos, se puede crear con la siguiente sentencia:

```
create (a :URL {link: "A"}), (b :URL {link: "B"}),
(c :URL {link: "C"}), (d :URL {link: "D"}),
(e :URL {link: "E"}), (f :URL {link: "F"}),
(a)-[:follows]->(b),  (a)-[:follows]->(c),  (a)-[:follows]->(f),
(b)-[:follows]->(a),  (b)-[:follows]->(c),  (b)-[:follows]->(d),
(c)-[:follows]->(f),
(d)-[:follows]->(a),  (d)-[:follows]->(e),
(e)-[:follows]->(a),  (e)-[:follows]->(c),
(f)-[:follows]->(c),   (f)-[:follows]->(e),  (f)-[:follows]->(f)
RETURN a, b, c, d, e, f
```

Se pide:

2.1) Sean las siguientes 2 consultas sobre dicho grafo, indicar la letra que corresponde a la única opción correcta.

| Consulta Q1: |
|---|
| MATCH path= (p) <-[]- (p)<br>RETURN path<br>ORDER BY p.link |

| Consulta Q2: |
|---|
| MATCH path= (p) -[]-> (x)<br>WHERE p.link = x.link<br>RETURN path<br>ORDER BY p.link |

Opción A: Ambas consultas devuelven exactamente el mismo resultado.

Opción B: Q1 devuelve un grafo que contiene lo que devuelve Q2, es decir Q1 devuelve más información (nodos y/o ejes) que Q2.

Opción C: Q2 devuelve un grafo que contiene lo que devuelve Q1, es decir Q2 devuelve más información (nodos y ejes) que Q1.

| Rta OPCION A |
|---|
| |

2.2) Sean las siguientes 2 consultas sobre dicho grafo, indicar la letra que corresponde a la única opción correcta.

| Consulta Q1: |
| --- |
| MATCH path= (p) -[]-> (x) -[]-> (p)<br>RETURN path |

| Consulta Q2: |
| --- |
| MATCH path= (p) -[*2]-> (p)<br>RETURN path |

Opción A: Ambas consultas devuelven exactamente el mismo resultado.

Opción B: Q1 devuelve un grafo que contiene lo que devuelve Q2, es decir Q1 devuelve más información (nodos y/o ejes) que Q2.

Opción C: Q2 devuelve un grafo que contiene lo que devuelve Q1, es decir Q2 devuelve más información (nodos y ejes) que Q1.

Rta **OPCION A**

2.3) Sean las siguientes 2 consultas sobre dicho grafo, indicar la letra que corresponde a la única opción correcta.

| Consulta Q1: |
| --- |
| MATCH path= (p) -[*1..2]-> (p)<br>RETURN path |

| Consulta Q2: |
| --- |
| MATCH path= (p) -[*2]-> (p)<br>RETURN path |

Opción A: Ambas consultas devuelven exactamente el mismo resultado.

Opción B: Q1 devuelve un grafo que contiene lo que devuelve Q2, es decir Q1 devuelve más información (nodos y/o ejes) que Q2.

Opción C: Q2 devuelve un grafo que contiene lo que devuelve Q1, es decir Q2 devuelve más información (nodos y ejes) que Q1.

Rta **OPCION B**

**Q1 devuelve un grafo que contiene lo que devuelve Q2, es decir Q1 devuelve más información (nodos y/o ejes) que Q2.**

## Ejercicio 3

Sabemos que según el DE-9IM para que una la relación entre un linestring y un polygon del tipo **st_touches(linestring, polygon)** devuelva **true** debe matchear alguna de estas 3 matrices

Matriz 1

| | I (g2) | B(g2) | E(g2) |
|---|---|---|---|
| I(g1) | False | | |
| B(g1) | | true | |
| E(g1) | | | |

Matriz 2

| | I (g2) | B(g2) | E(g2) |
|---|---|---|---|
| I(g1) | False | true | |
| B(g1) | | | |
| E(g1) | | | |

Matriz 3

| | I (g2) | B(g2) | E(g2) |
|---|---|---|---|
| I(g1) | False | | |
| B(g1) | true | | |
| E(g1) | | | |

Sea el polígono **'Polygon( ( -88 35, -88 20, 22 20, 22 35, -88 35) )'::Geometry**

**3.1)** Proponer un linestring tal que devuelva TRUE al evaluar si la misma ST_TOUCHES dicho polígono. Es decir, completar en el rectángulo azul, cómo debería definirse dicha linestring para que al ejecutar esta consulta se obtenga como resultado TRUE.

```
SELECT
ST_TOUCHES('Polygon( ( -88 35, -88 20, 22 20, 22 35, -88 35) )'::Geometry,
                'LineString(    -88 35, -88 20    '::Geometry )
```

Rta

En el rectángulo azul debería colocarse: -88 35, -88 20

**3.2)** Sea geom1 el polígono y **geom2 el linestring** del item 3.1. Completar el encoding de ambos.

Rta:

| | **Interior geom2** | **Border geom2** | **Exterior geom2** |
|---|---|---|---|
| Interior geom1 | F | F | T |
| Borde geom1 | T | T | T |
| Exterior geom1 | F | F | T |

**3.3)** Indicar con cuál de las matrices que puede corresponder a un ST_TOUCHES dio matching dicho encoding: con la matriz 1, la matriz 2 o la matriz 3?

Rta

Nuestro encoding dio como resultado que el **Borde de geom1** intersecta con el **Interior de geom2**

**Matriz 3** es la única que tiene la condición B(g1) con I(g2) en **true**, lo cual corresponde al caso donde la línea descansa sobre el borde del polígono.

**Ejercicio 4**

Tomando la base de datos grafo utilizada en los TP 3 y TP 4 (fixeada según TP 3 ejercicio 0, y TP4 ejercicio 1 y 21. Es decir, los idiomas son arreglos de strings, la duración de las llamadas son integer, la longitud de los sms son integer y las coordenadas lat / lon de las ciudades son floats )

**Se pide:**

**4.1)**

Ejecutar la siguiente sentencia que genera una ciudad adicional

```
MATCH (p: Pais)
WHERE p.Nombre='Espana'
CREATE (b :Ciudad { Nombre:"Barcelona", lon: 2.15899, lat: 41.388, ID: "9999"}),
(b)-[:perteneceA]->(p)
RETURN p, b
```

Proponer una consulta para calcular **solo para aquellos países que tienen TODAS sus ciudades al este del meridiano de Greenwich**: las ciudades que poseen. En el resultado debe aparecer el país y una colección con las ciudades que están al este. Es decir, aparecen 2 columnas.

Rta

```
MATCH (p:Pais)<-[:perteneceA]-(c:Ciudad)
WITH p, collect(c) AS ciudades
WHERE ALL(x IN ciudades WHERE x.lon > 0)
RETURN p.Nombre AS Pais,
[ciudad IN ciudades | {Nombre: ciudad.Nombre, Longitud: ciudad.lon}] AS ColeccionCiudadesAlEste
```

**4.2)** Mostrar el resultado que resulta al ejecutar la query propuesta en 4.1, en formato tabla.

| Nodo Pais (con todas sus properties) | ColeccionCiudadesAlEste |
|---|---|
| Alemania | [ {Nombre: "Berlin", lon: 13.40}, {Nombre: "Frankfurt", lon: 8.68} ] |
| Austria | [ {Nombre: "Viena", lon: 16.36} ] |
| Belgica | [ {Nombre: "Amberes", lon: 4.40}, {Nombre: "Bruselas", lon: 4.34}, {Nombre: "Brujas", lon: 3.22} ] |
| Francia | [ {Nombre: "Paris", lon: 2.34} ] |
| Italia | [ {Nombre: "Roma", lon: 12.49}, {Nombre: "Milan", lon: 9.18}, {Nombre: "Napoles", lon: 14.30}, {Nombre: "Venecia", lon: 12.32} ] |

| | |
|---|---|
| Monaco | [ {Nombre: "Monaco", lon: 7.41}, {Nombre: "Montecarlo", lon: 7.41} ] |
| Suiza | [ {Nombre: "Berna", lon: 7.44}, {Nombre: "Ginebra", lon: 6.14} ] |

**Ejercicio 5**

Utilizando la base de datos que hemos creado en Postgis de quiere calcular la cantidad de estaciones de subte que caen dentro de la zona delimitada por la calle circular 'Columbus Cir'

La query propuesta es:

```
SELECT circular.geom, s.name, s.gid
FROM nyc_streets circular, nyc_subway_stations s
WHERE circular.name = 'Columbus Cir' and st_contains(circular.geom, s.geom)
```

En QGis, veo claramente que tiene 2 estaciones de subte

Sin embargo, al ejecutarla se obtiene resultado vacío. Pero visualmente se puede ver que hay 2 estaciones que caen dentro de la zona delimitada por dicha calle.

Explicar claramente por qué no se obtiene en el resultado esas 2 estaciones de subte.

Rta:

El resultado está vacío porque la geometría de la calle (circular.geom) es un LineString (una línea), no un Polígono. La función ST_Contains verifica si el punto está en el interior de la otra geometría. Una línea (1D) no tiene área interior; solo tiene longitud. Por lo tanto, una línea nunca puede tener un punto a menos que el punto esté exactamente encima del trazo negro de la calle.