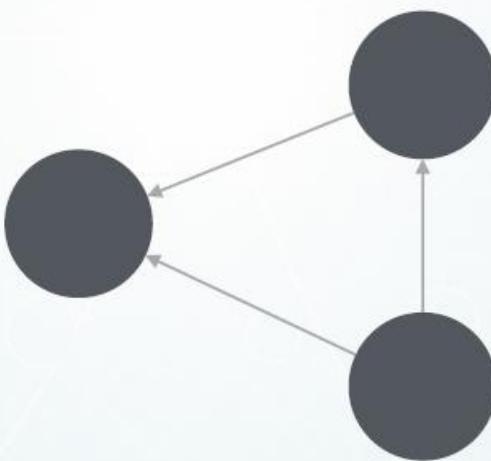


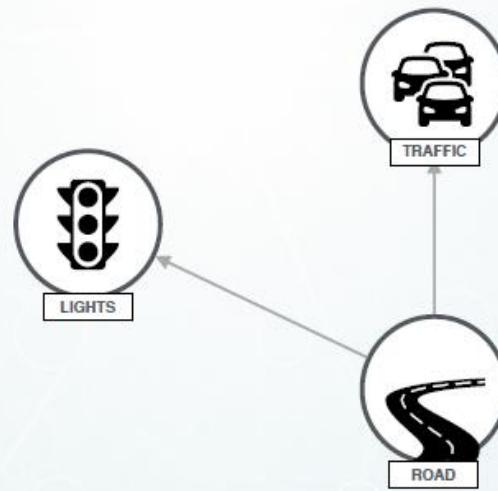
BASES DE DATOS BASADAS EN GRAFOS



Un grafo son datos conectados



Un grafo son datos conectados



Una manera de representar data

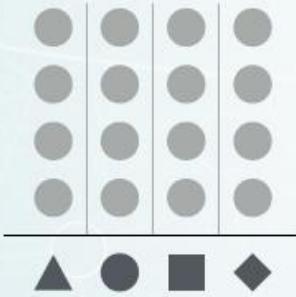


DATA

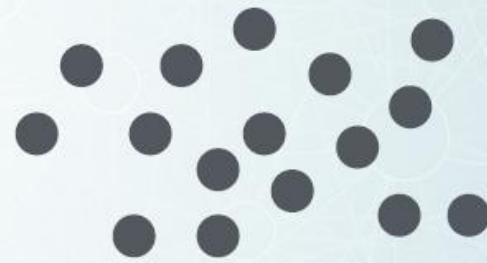


DATA

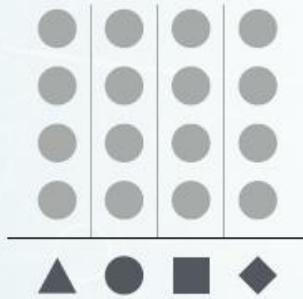
Una manera de representar data



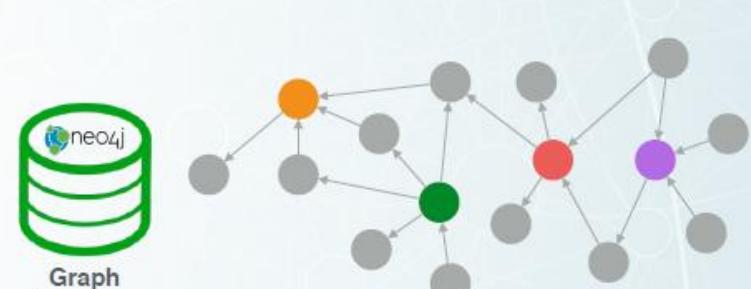
Relational Database



Una manera de representar data



Relational Database



Bueno para:

- Estructuras de datos conocidas que no cambian con frecuencia.
- Problemas conocidos que involucran datos discretos o con mínima conectividad.

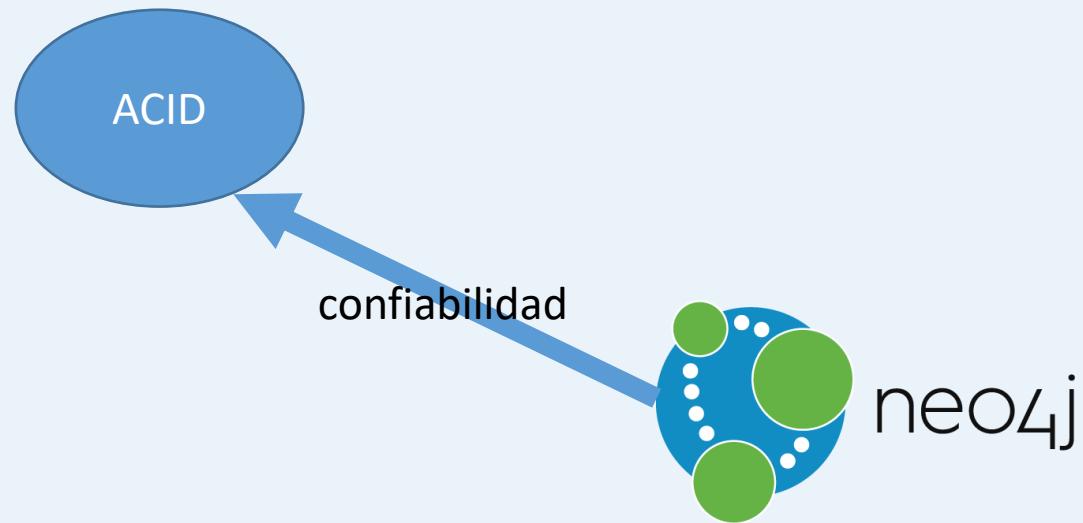
Bueno para:

- Sistemas dinámicos: donde la topología de los datos es difícil de predecir.
- Requerimientos dinámicos: relacionado con las reglas propias del negocio.
- Problemas donde las relaciones entre los datos contribuyen a dar significado y valor.

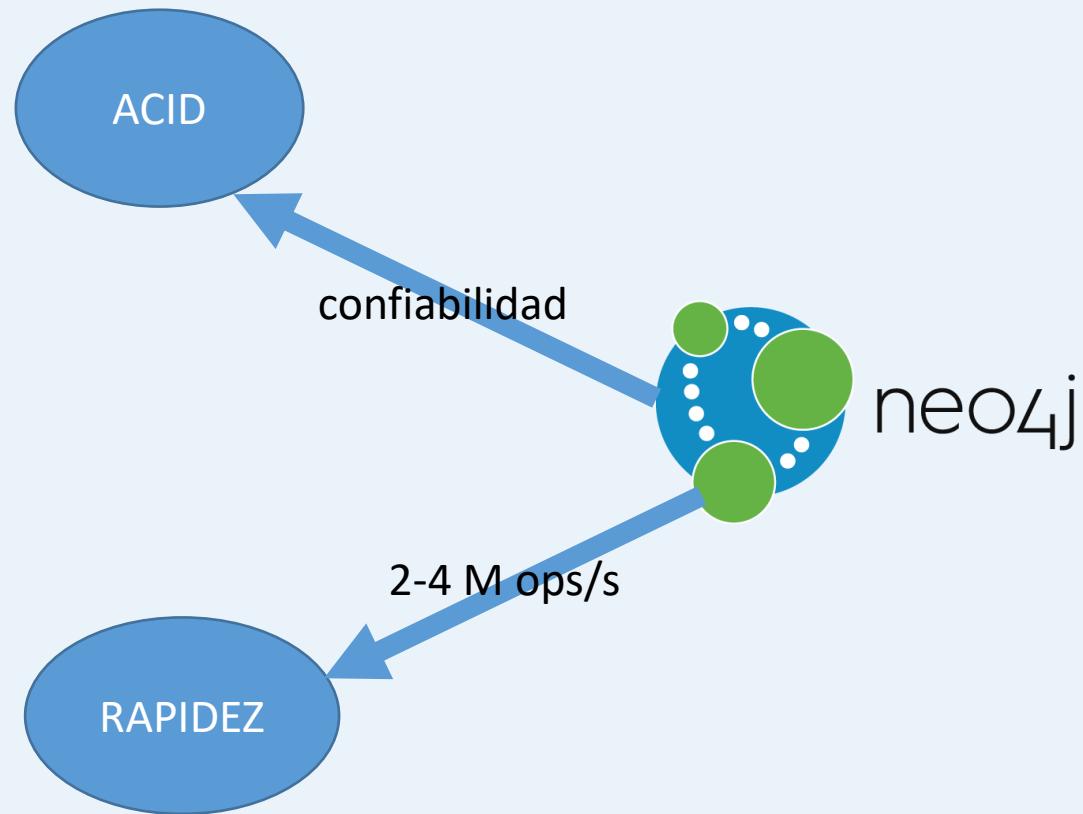
Noe4j es una base de datos



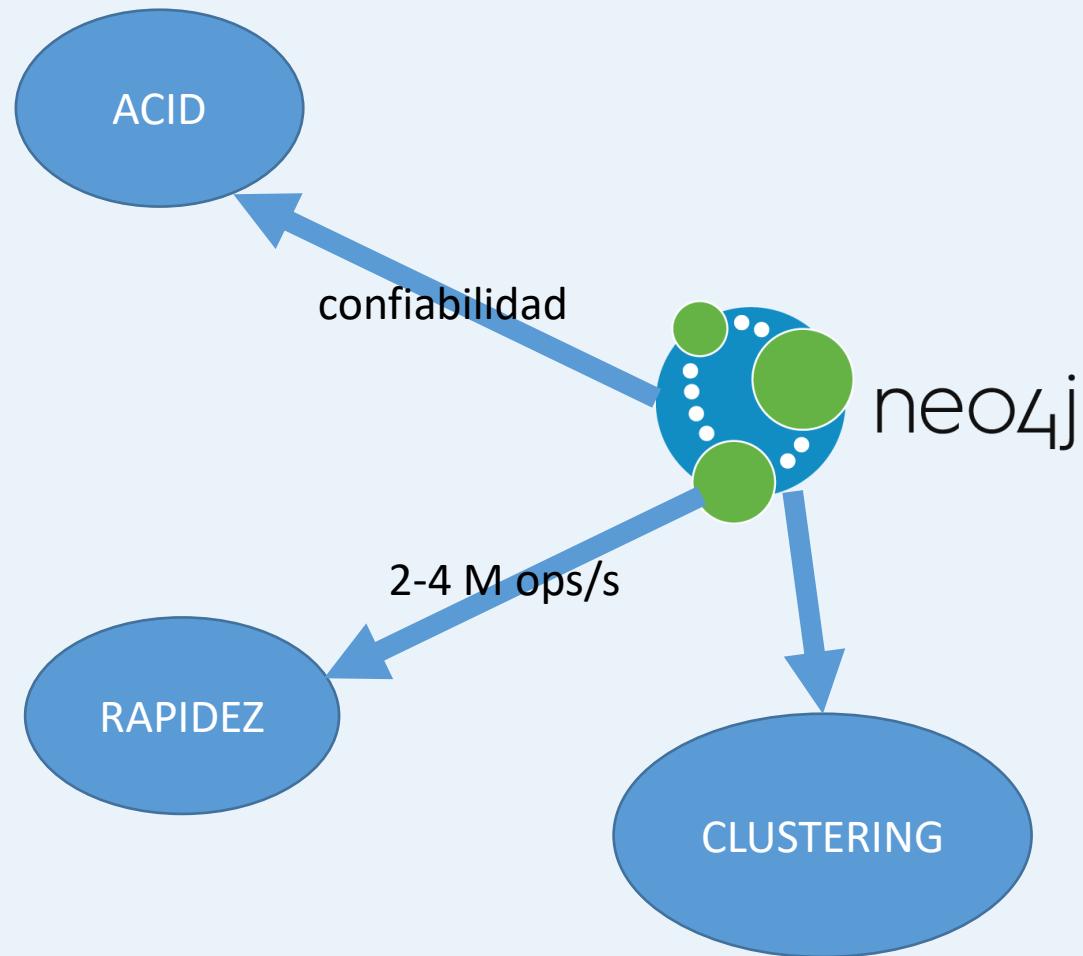
Noe4j es una base de datos



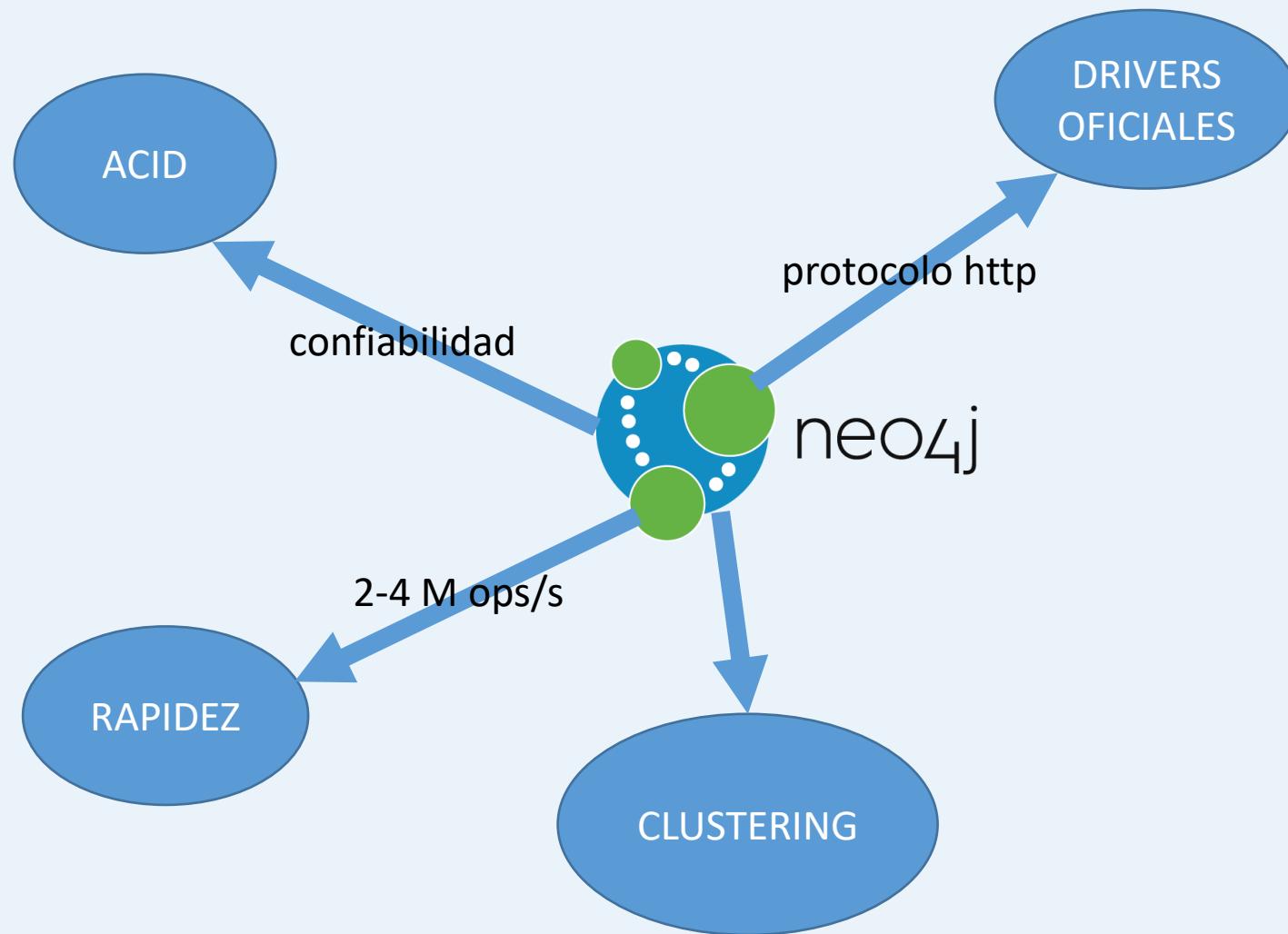
Noe4j es una base de datos



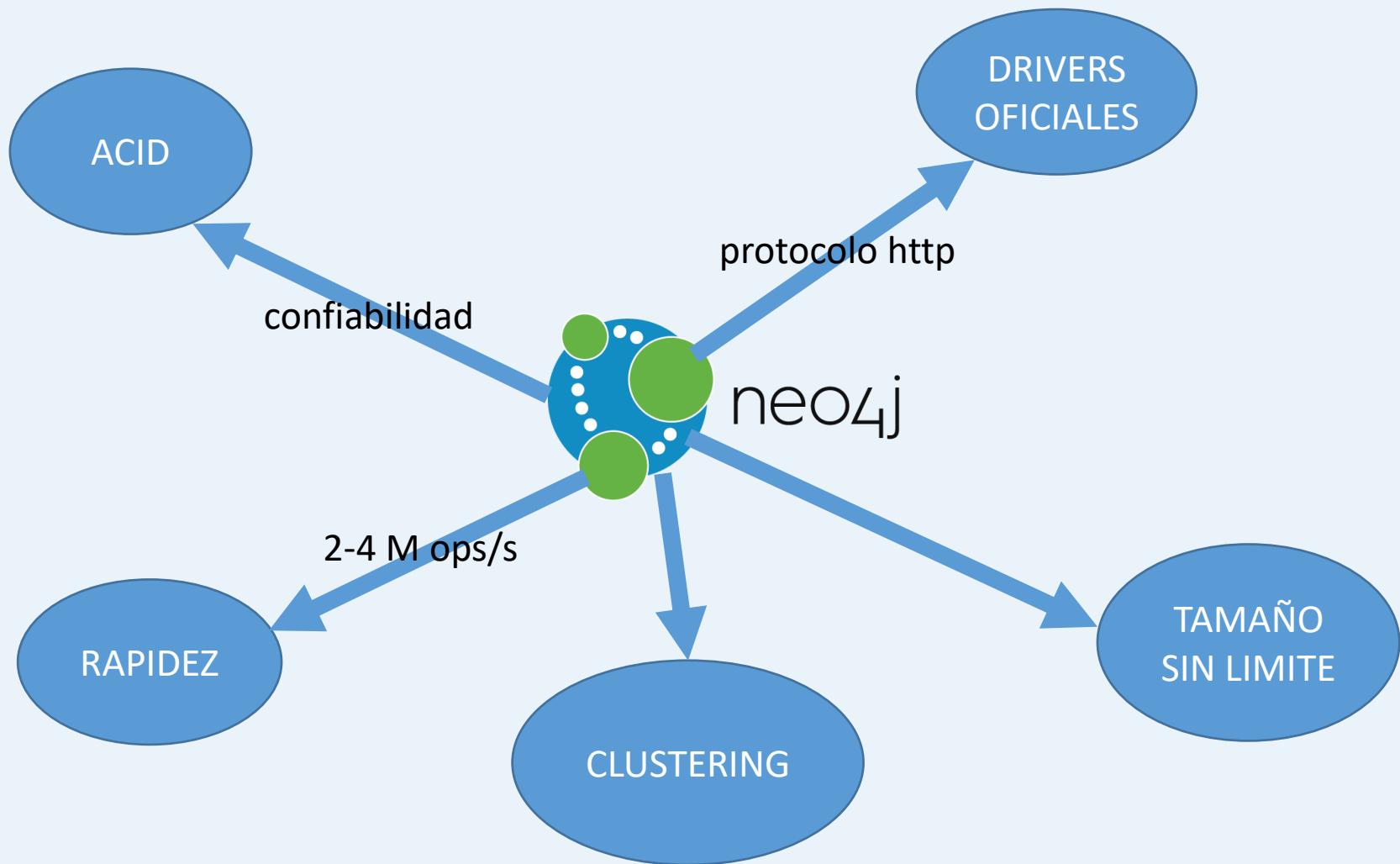
Noe4j es una base de datos



Noe4j es una base de datos



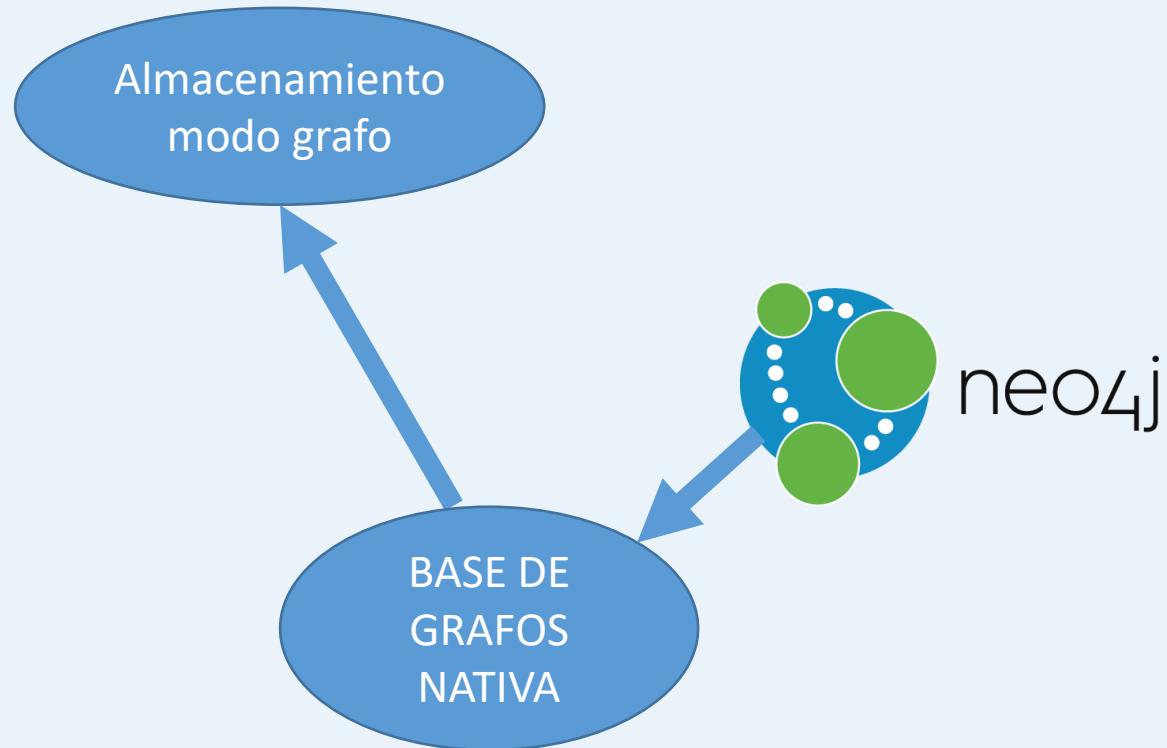
Noe4j es una base de datos



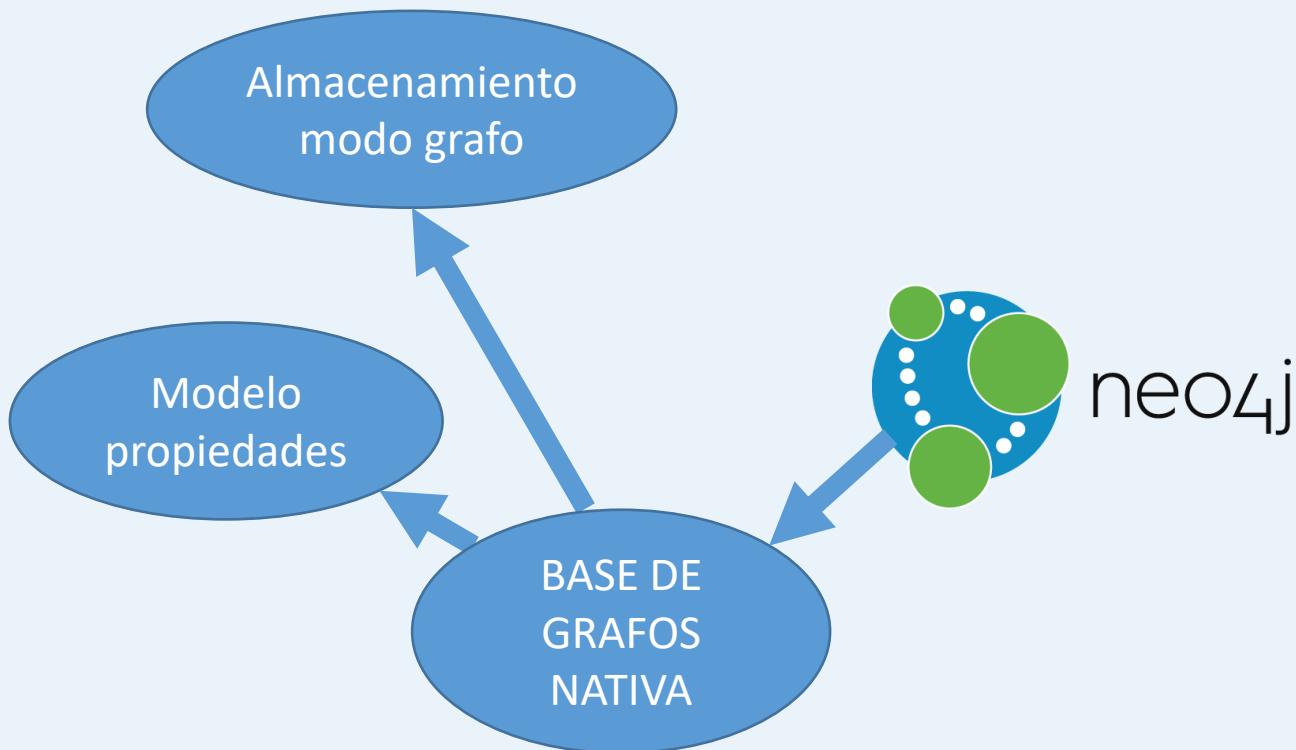
Noe4j es una base de datos



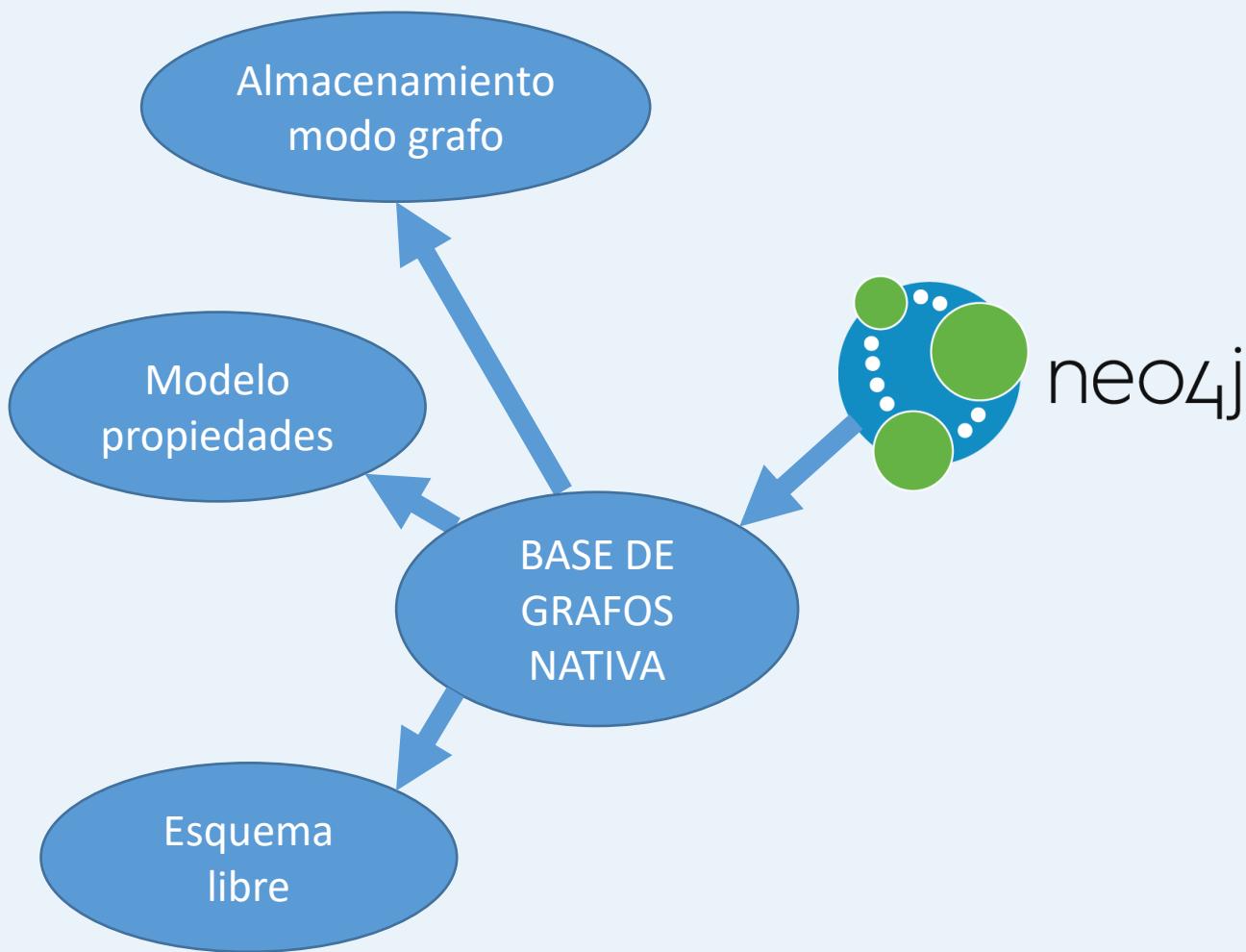
Noe4j es una base de datos



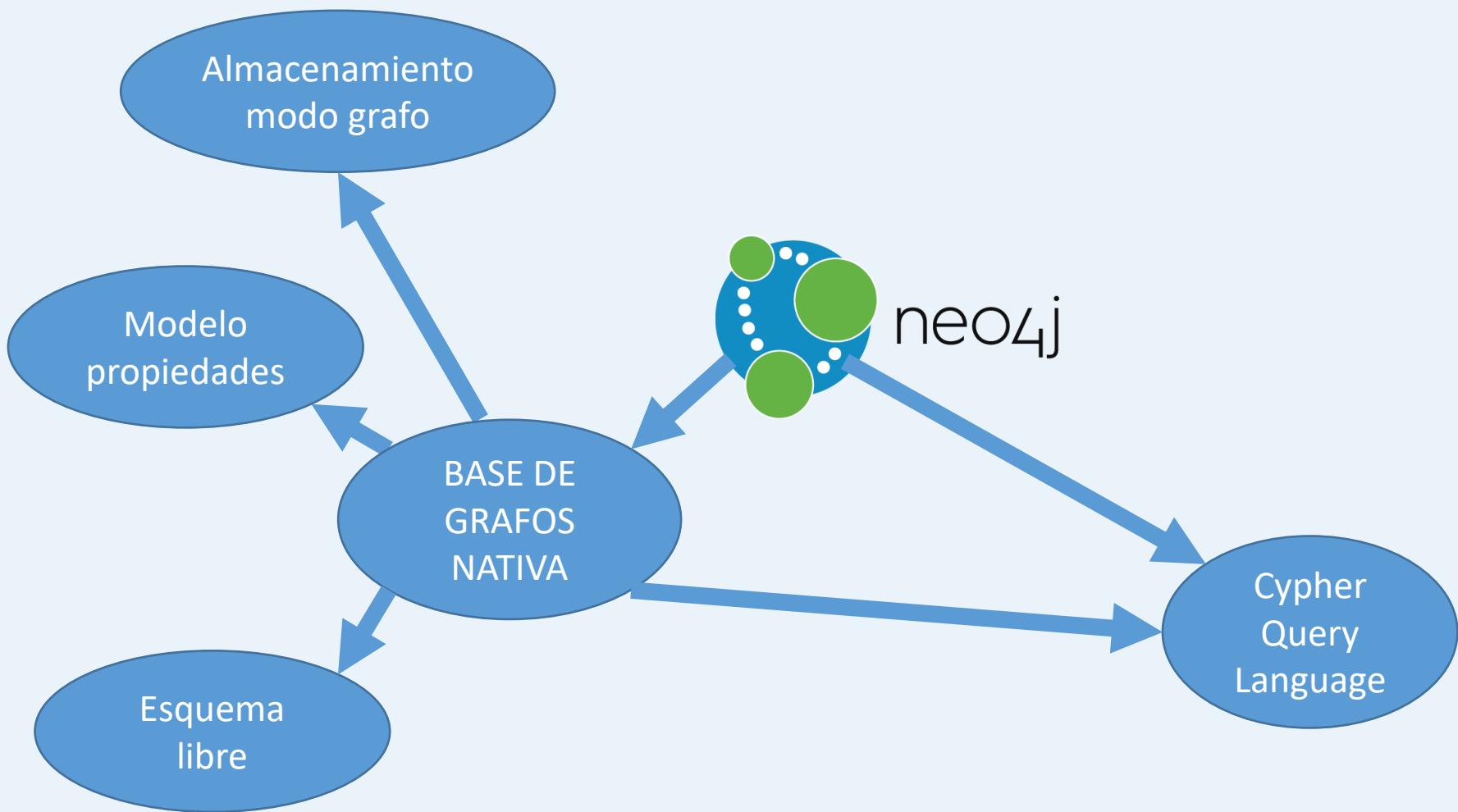
Noe4j es una base de datos



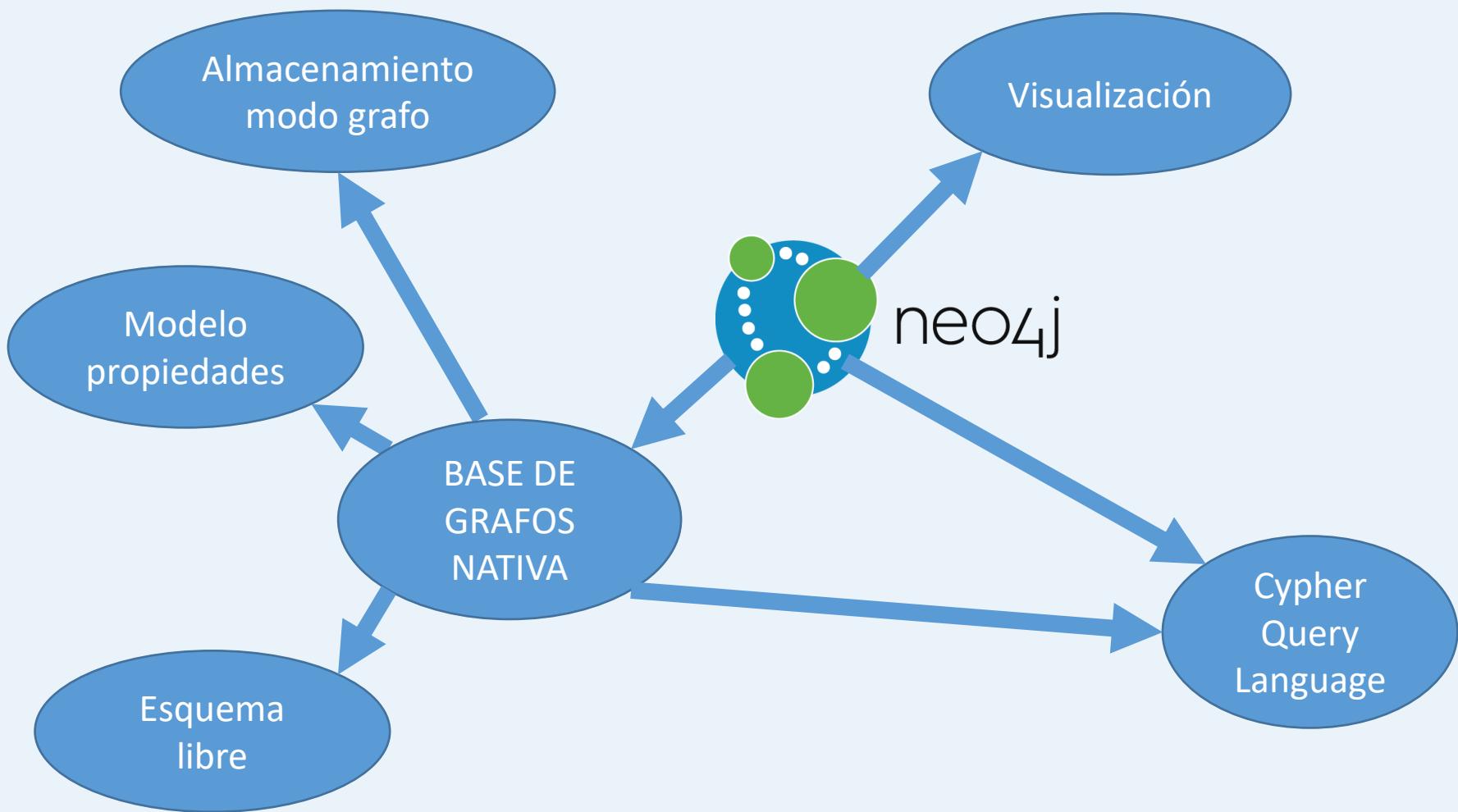
Noe4j es una base de datos



Noe4j es una base de datos



Noe4j es una base de datos



Aumento de conexiones de datos

Las conexiones de datos crecen tan rápido como el volumen de los mismos.

Redes de Personas

Empleados, Clientes,
Proveedores, Partners,
Influencers

Procesos de Negocios

Gestión de riesgos,
Cadena de proveedores,
Pagos, eComerce

Redes de conocimiento

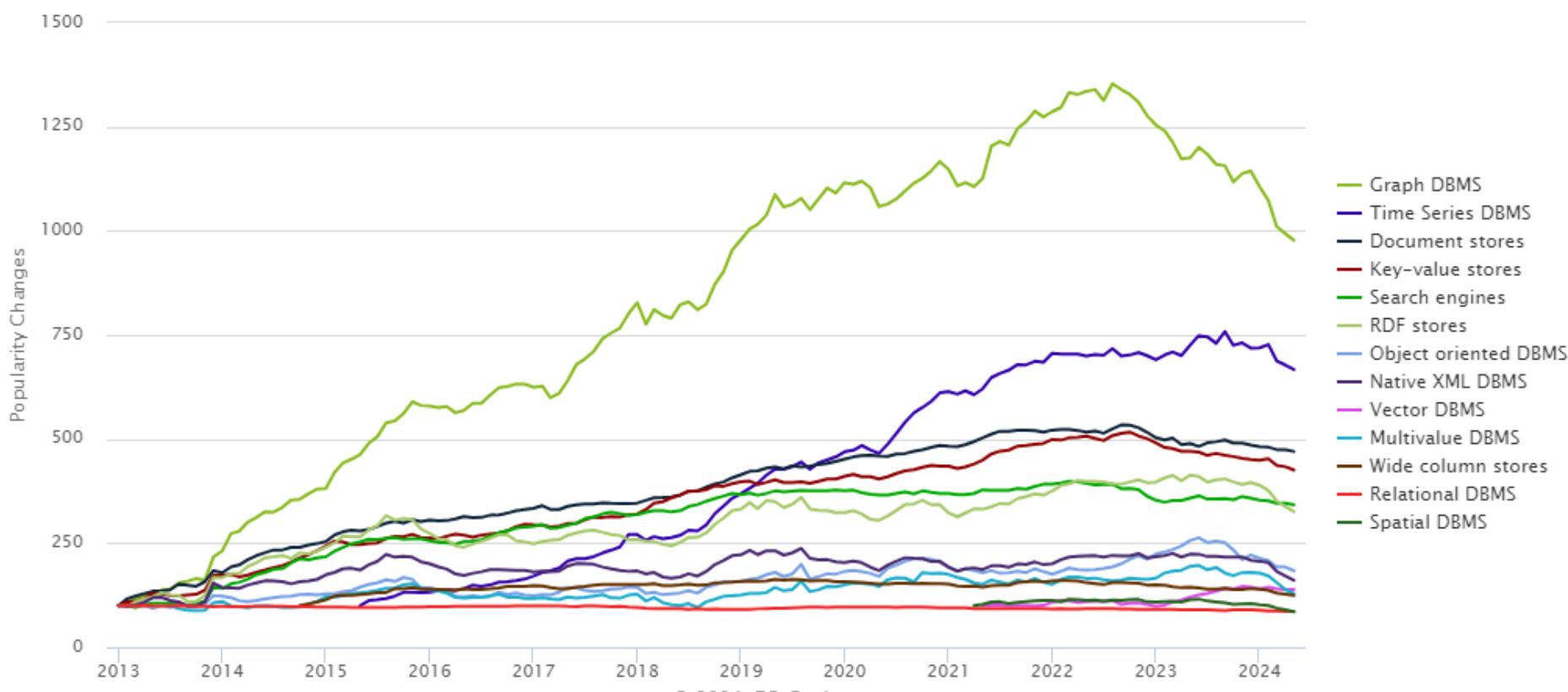
Datos de empresas,
Contenido de dominio
específico

Ejemplos de usuarios corporativos

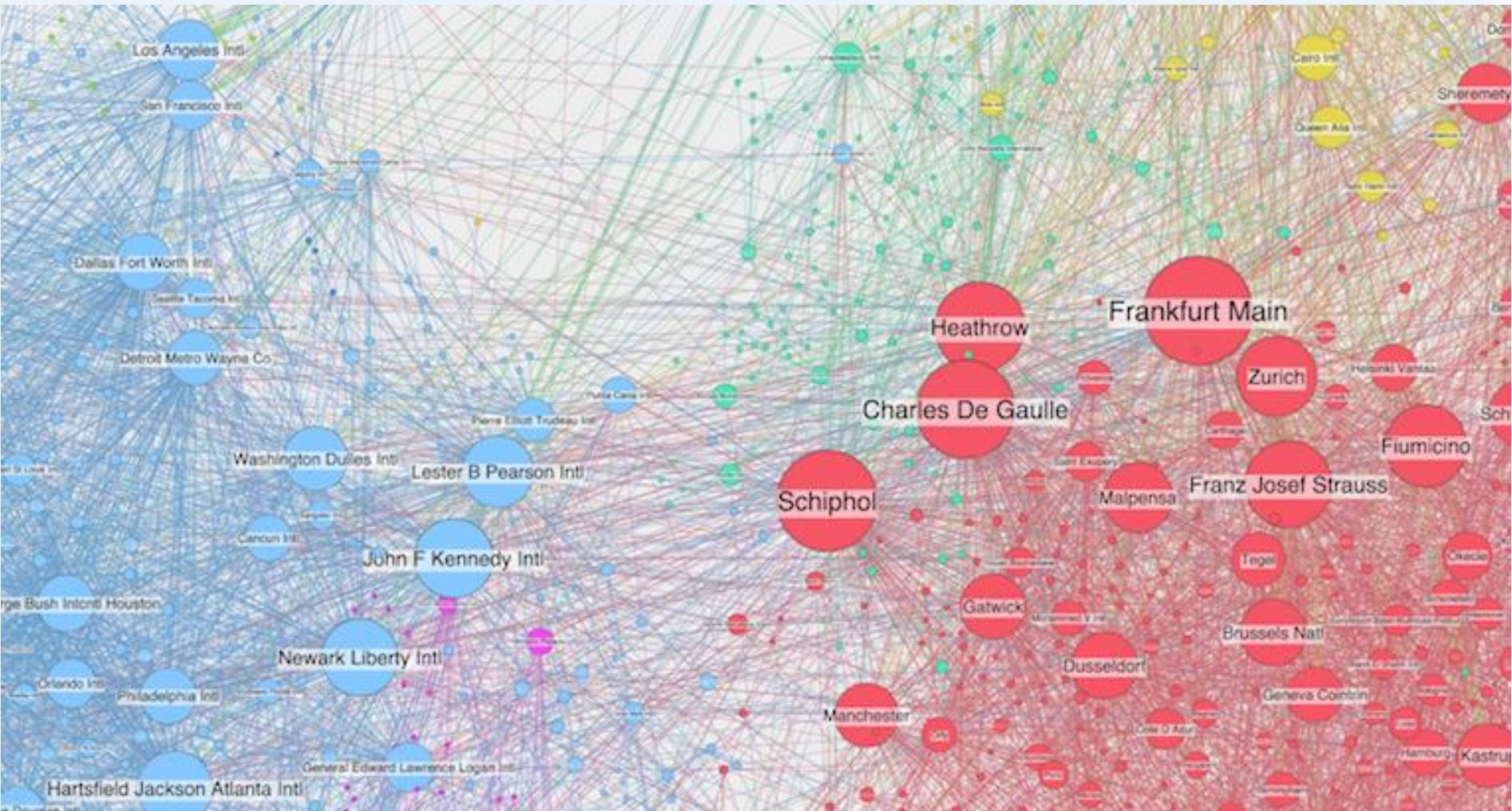
- [Citibank](#) - Mejorar la interacción y aumentar la satisfacción de los clientes con una visión 360
- [Zurich](#) - Investigación de Fraude con Neo4j
- [Ebay](#) - Generación de recomendaciones en tiempo real basadas en el comportamiento de los clientes
- [Cisco](#) - Ahorro de 4 millones de horas utilizando el Knowledge Graph (Grafo de Conocimiento) de Neo4j
- Adobe - Reducción de la huella de datos de 50TB a 40GB. Millones de dólares ahorrados en costos de AWS.
- [IBM](#) - Mejora del Watson Knowledge Catalog y Match 360 en IBM Cloud Pak for Data
- [UBS](#) - Mejora en la gestión de riesgos y cumplimiento
- [Panama Papers](#) - Detección de fraudes financieros
- Shopify - 10x ROI en la detección de fraudes
- Mapfre - Construcción de modelos de aprendizaje automático para mejorar la detección de fraudes en la apertura de reclamaciones
- Fidelity - Construcción de un Knowledge Graph (grafo de conocimiento) para los gestores de cartera y los activos que gestionan
- LATAM - Jornada del cliente y resolución de identidad

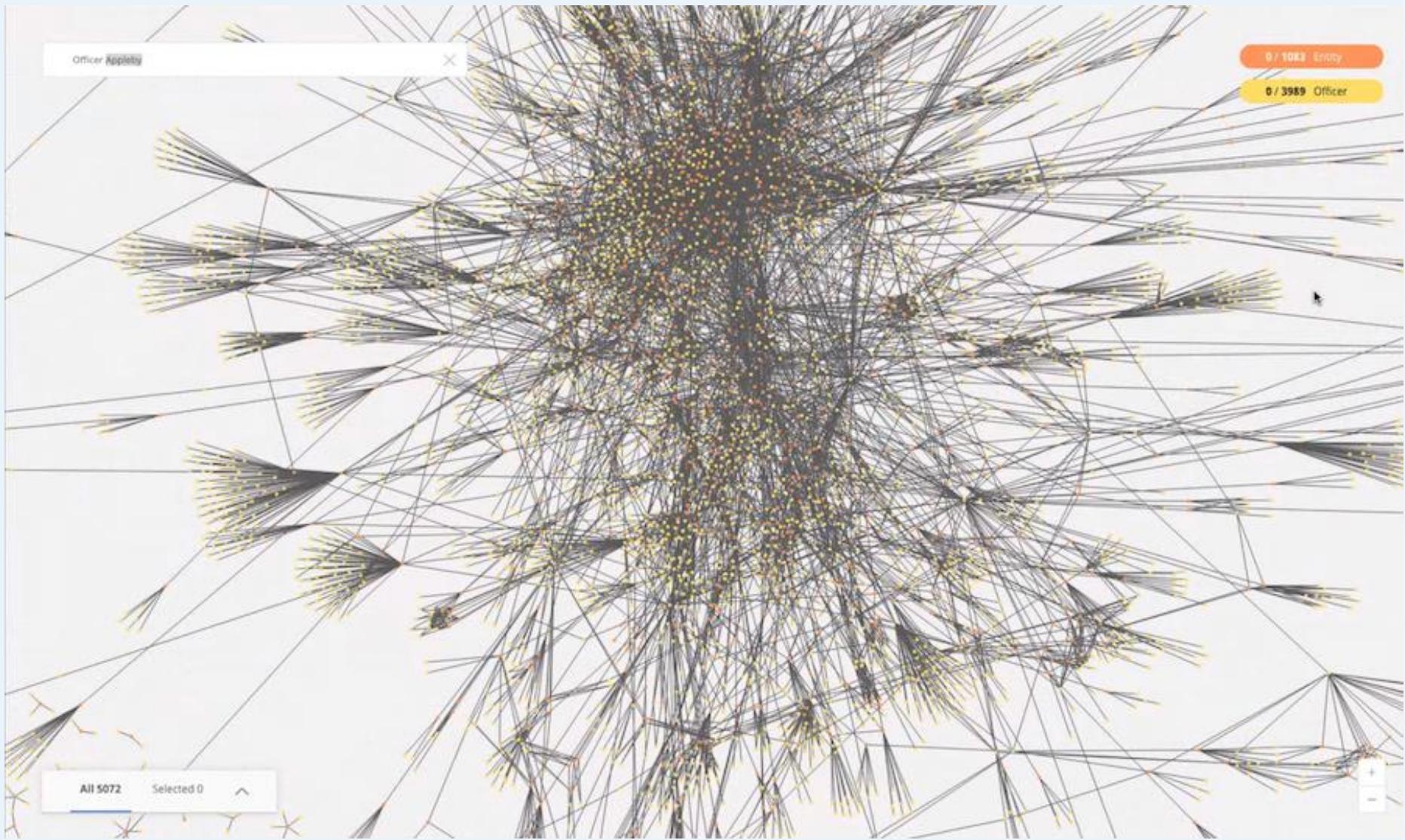
Interés por las bases de datos basadas en Grafos

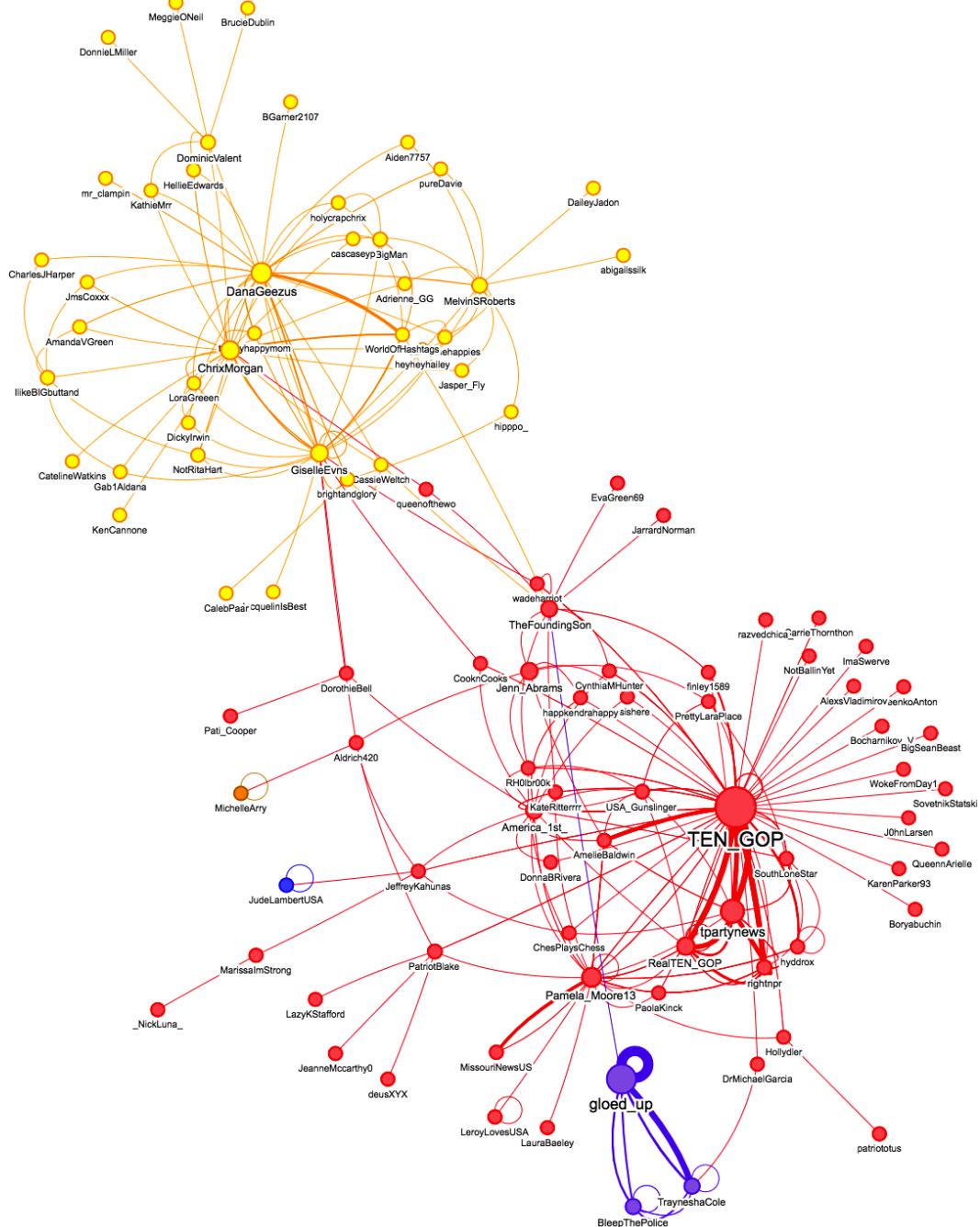
Complete trend, starting with January 2013

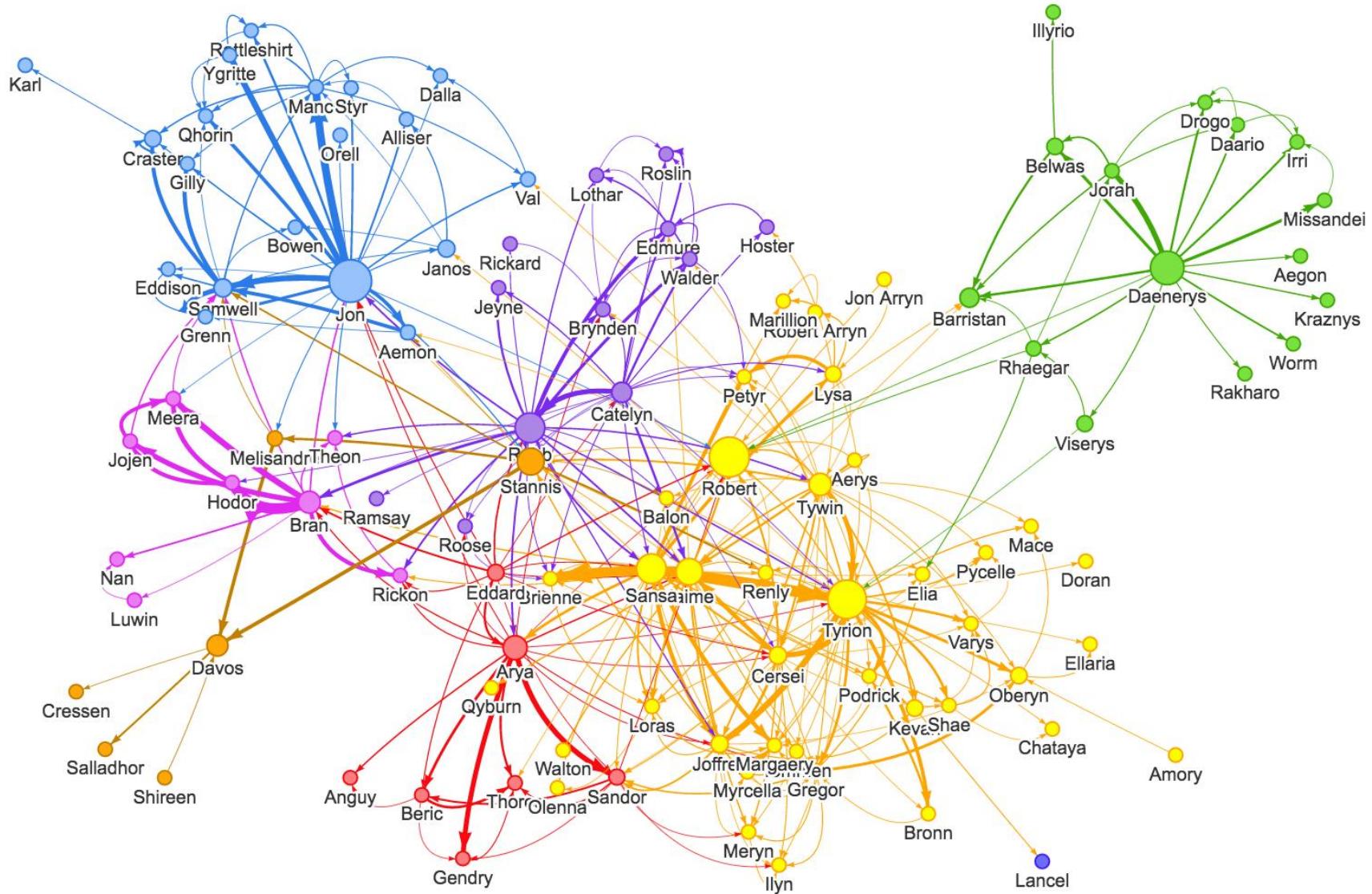


Bases de datos de Grafos: conceptos









Neo4j: Modelo de grafo etiqueta-propiedad

Nodos

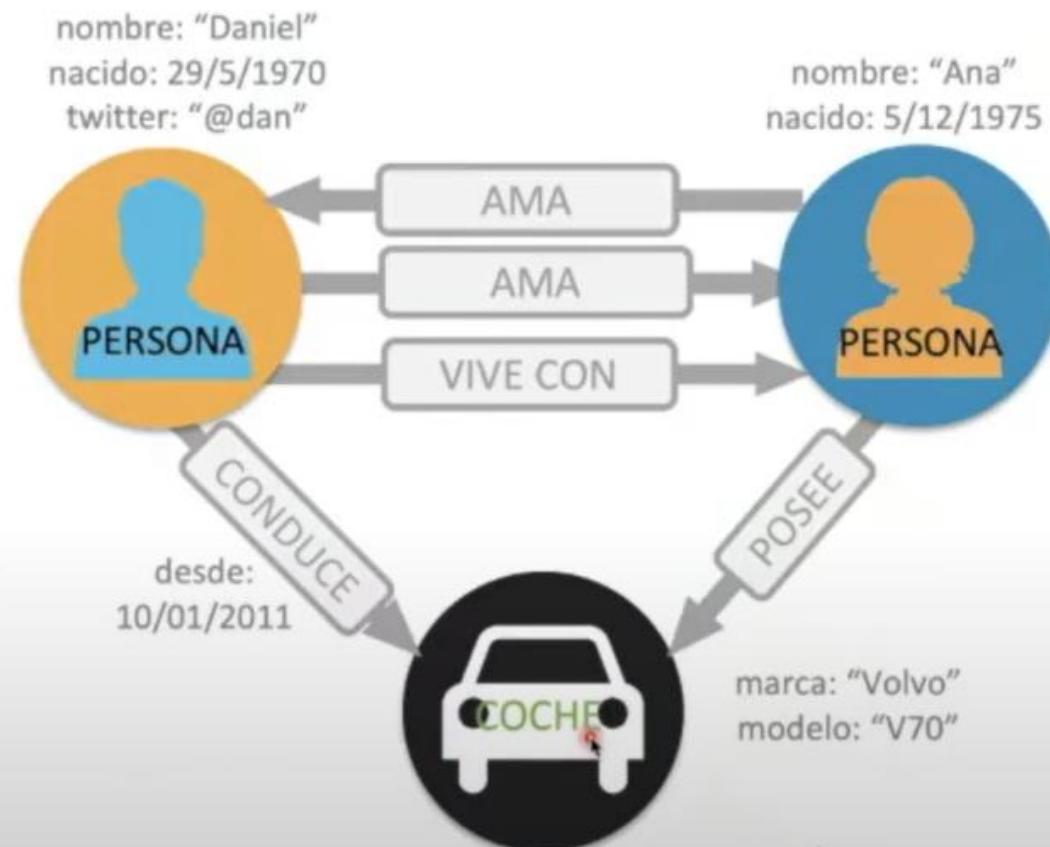
- Pueden tener etiquetas para su clasificación
- Las etiquetas tienen índices nativos

Relaciones

- Relacionan nodos por tipo y dirección

Propiedades

- Atributos de nodos y relaciones
- Almacenados como pares Clave/Valor
- Pueden tener índices e índices compuestos



Cypher: Lenguaje de consulta poderoso y expresivo.



```
MATCH (:Persona { nombre:"Daniel" }) -[:CASADO_CON]-> (esposa)
```

LABEL

PROPERTY

VARIABLE

ALGORITMOS DE GRAFOS



Pathfinding & Search

- Parallel Breadth First Search & DFS
- Shortest Path
- Single-Source Shortest Path
- All Pairs Shortest Path
- Minimum Spanning Tree
- A* Shortest Path
- Yen's K Shortest Path
- K-Spanning Tree (MST)
- Random Walk



Centrality / Importance

- Degree Centrality
- Closeness Centrality
- CC Variations: Harmonic, Dangalchev, Wasserman & Faust
- Betweenness Centrality
- Approximate Betweenness Centrality
- PageRank
- Personalized PageRank
- ArticleRank
- Eigenvector Centrality



Community Detection

- Triangle Count
- Clustering Coefficients
- Connected Components (Union Find)
- Strongly Connected Components
- Label Propagation
- Louvain Modularity – 1 Step & Multi-Step
- Balanced Triad (identification)



Similarity

- Euclidean Distance
- Cosine Similarity
- Jaccard Similarity
- Overlap Similarity
- Pearson Similarity



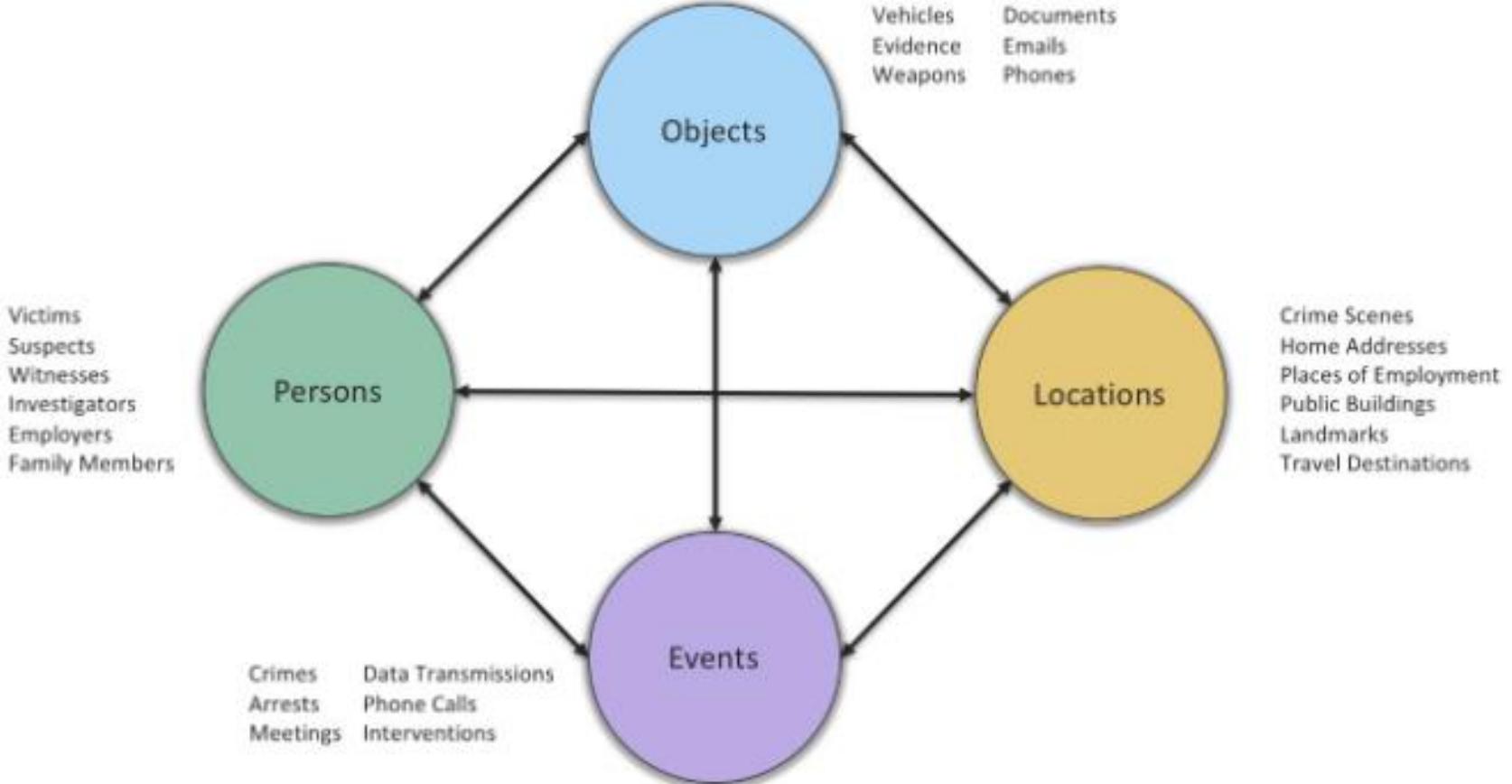
Link Prediction

- Adamic Adar
- Common Neighbors
- Preferential Attachment
- Resource Allocations
- Same Community
- Total Neighbors

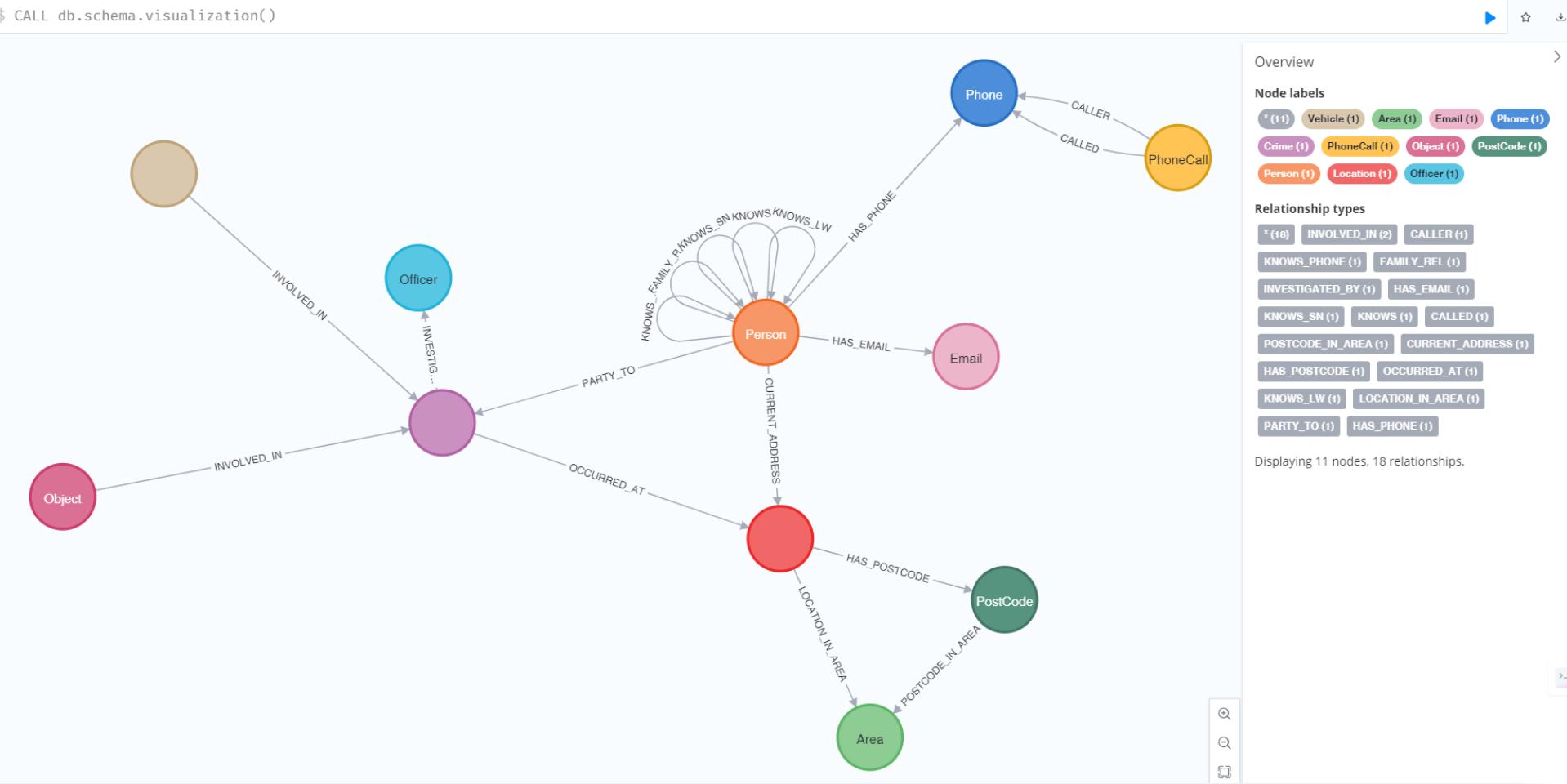
Base de datos Crime, Manchester U.K.



Modelo POLE: Person, Object, Location, Event



Estructura de Crimen

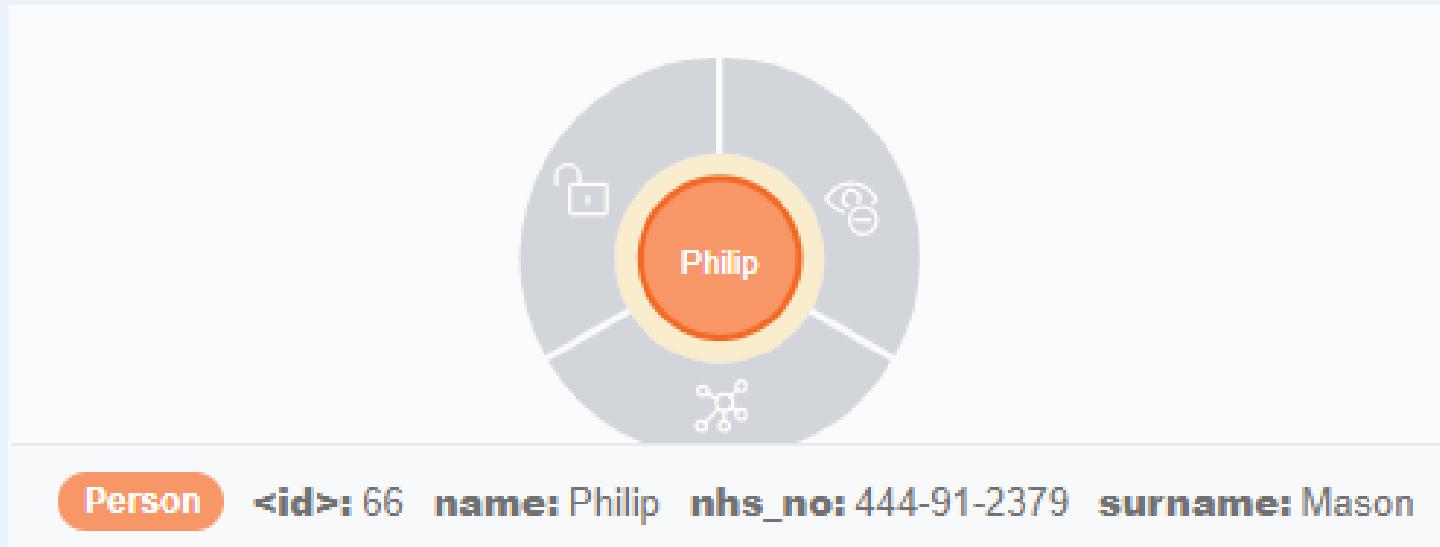


Procedimientos para conocer la base

- **CALL db.schema.visualization()**
- **CALL db.labels**
- **CALL db.relationshipTypes**
- **CALL db.propertyKeys**
- **SHOW PROCEDURES yield name, description, signature**

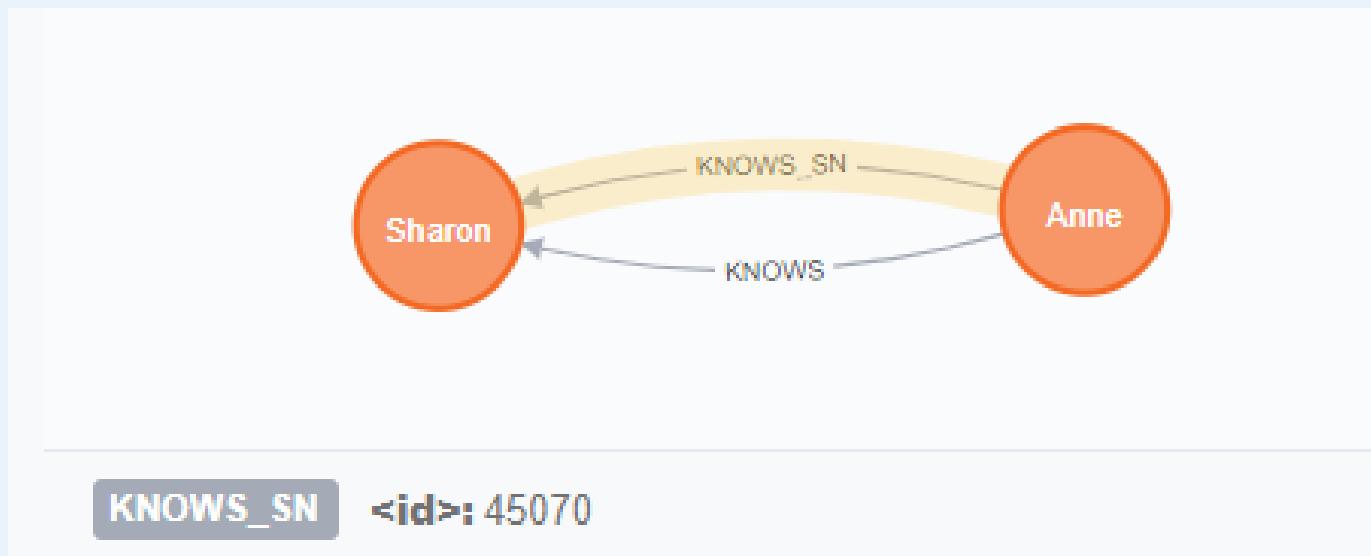
Nodos

- Los nodos tienen valores para distintas propiedades (id: interno)
- Además pueden tener varias etiquetas



Arcos

- Tienen un único tipo
- Pueden también tener propiedades
- Tienen una dirección
- Vinculan dos nodos, o un único nodo si se vincula consigo mismo



MATCH – Estructura general

MATCH *Patrón*

[**WHERE** *Filtros*]

RETURN [**DISTINCT**] *Respuestas*

[**ORDER BY** *Expresiones*]

[**SKIP** *cant.* **LIMIT** *cant.*]

MATCH – Estructura general

- **Patrón:**
 - Estructura del subgrafo a revisar
- **Filtros (Opcional):**
 - Condiciones que debe cumplir el subgrafo para ser devuelto
- **Respuestas:**
 - Se puede devolver nodos y arcos completos o el valor de algunas propiedades
 - Separar con comas
 - Opción DISTINCT
- **Orden y paginado (Opcional):**
 - Similar a SQL

Patrón de nodos

Lo único obligatorio:
estar entre paréntesis

(alias:etiqueta {filtros})

Útil para
referenciar en
filtros/orden o
para devolver

Si debe tener varias
etiquetas, usar
:et1:et2
(Es un AND)
Ojo: Es case
sensitive

Expresados como
{nombre:valor
, nombre2:valor2}
, similar a JSON.
Sólo filtros de
igualdad

EJ: (ce:Person {name:"Gregory"})

Filtros en WHERE vs en patrón

MATCH (ce:Person)

WHERE ce.name = "Gregory"

RETURN ce

Es equivalente a

MATCH (ce:Person {name :"Gregory" })

RETURN ce

Manejo de strings

Valores case sensitive

Además de los operadores clásicos está el de expresiones regulares =~

Ejemplo: a.name =~ "A.*"

También cuenta con:

STARTS WITH,

ENDS WITH y

CONTAINS

Manejo de labels

- Distinto a otras propiedades del nodo

Devuelva el nombre y etiquetas de personas que no tienen “e” en sus nombres:

MATCH (n:Person)

WHERE NOT n.name **CONTAINS** "e "

RETURN n.name, labels(n)

LIMIT 20

Se devuelven con la
función labels

Primeras consultas

1. *Muestre en orden alfabético, los nombres de las primeras 10 personas apellidadas ‘Smith’.*
2. *Muestre la marca y modelos de los vehículos de año 2013.*
3. *Muestre el nombre, apellido y rango de los oficiales cuyos apellidos comiencen con ‘Mc’, ordenados por rango (rank).*

Arcos en el patrón

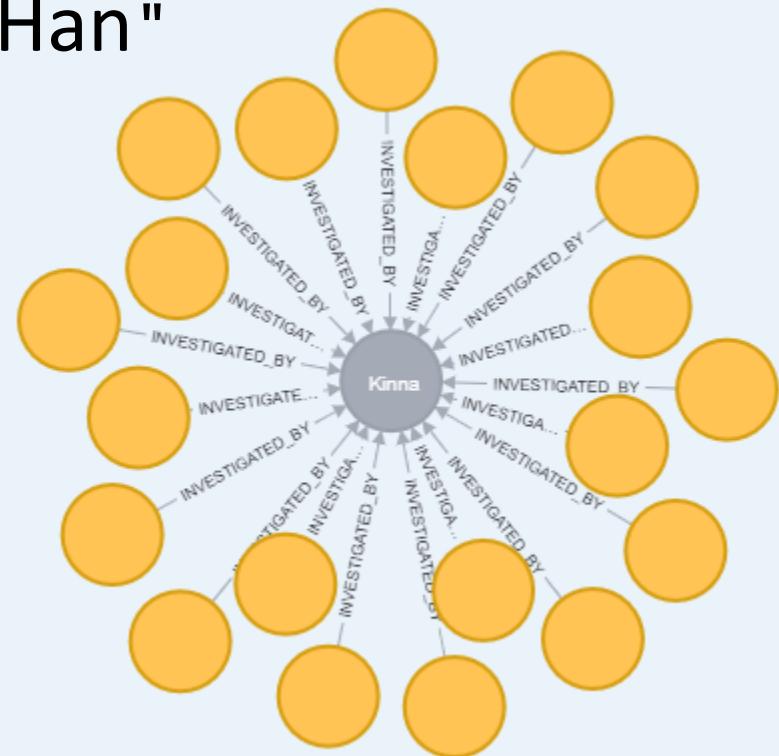
Crímenes relacionados con el oficial McHan

MATCH (n:Officer)--(m:Crime)

WHERE n.surname = "McHan"

RETURN n, m

LIMIT 20



Filtros de arcos

Permiten alias, etiquetas y propiedades

Pero van entre corchetes

Ej: Personas que viven con Craig Gordon

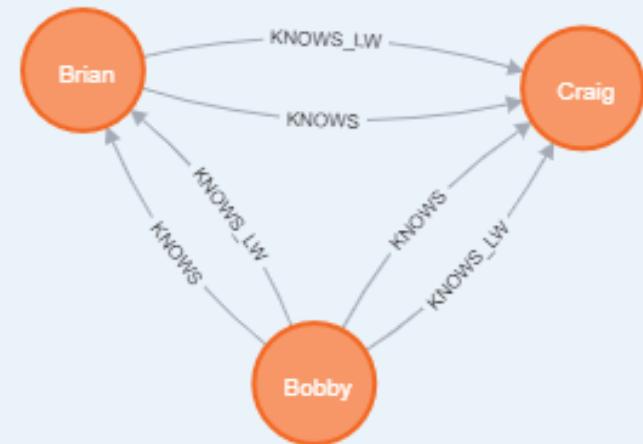
MATCH

(n:Person)-[d:**KNOWS_LW**]->(m:Person)

WHERE m.surname = "Gordon" AND

m.name="Craig"

RETURN n, m



Filtros de Arcos

Obtener la información de personas que sean familiares entre sí (FAMILY_REL) y sean hermanos ('SIBLING')

MATCH

```
(a:Person)-[r:FAMILY_REL{rel_type:"SIBLING"}]-  
(b:Person)
```

RETURN a,b

limit 20

Filtros de arcos - Extra

- Permiten definir dirección del arco

-[]-

-[]->

<-[]-

- Permiten definir cantidades de arcos

-[*]-

-[*cant]-

-[*min..]-

-[*..max]-

-[*min..max]-

**una teoría que se
llama los seis grados
de separación, que
no sé si**

Más consultas

4. Muestre el grafo de las locations en el área M30. Cuantos nodos hay?
5. Muestre el grafo de todos que conocen a alguien que conoce a Gordon Craig.
6. Muestre las personas que están a distancia 3 de Gordon Craig.

Aclaración Gordon es el surname.

Alias de Subgrafos

- Se puede definir un alias para trabajar

```
MATCH s=((n:Person)-[HAS_PHONE]->(p:Phone)<-[ ]-(o:PhoneCall))
```

```
WHERE n.surname="Gordon" and n.name="Craig"
```

```
RETURN s
```

- Funciones sobre subgrafos:

- Length(s)
- Relationships(s)
- Nodes(s)

Subgrafos en filtros

Se pueden usar como si fueran booleanos
Reusando alias, pero sin crear alias nuevos!

Personas que conocen a Craig Gordon y no a Bonnie Robinson:

```
MATCH (n:Person{surname:"Gordon", name:"Craig"})-[]-(p:Person)
```

```
WHERE NOT (p)-[]-(:Person{surname:"Robinson",  
name:"Bonnie"})
```

```
RETURN DISTINCT p.name, p.surname
```

Varios subgrafos

Se pueden poner varios subgrafos en el MATCH

Llamadas de Gordon a Moore

```
MATCH (n:Person{surname:"Gordon", name:"Craig"})--  
(p:Phone)--(o:PhoneCall),(m:Person{surname:"Moore",  
name:"Judith"})--(p1:Phone)--(o:PhoneCall)  
RETURN o
```

Caminos mas cortos

Es común querer el camino más corto entre dos nodos

```
MATCH s=shortestPath( (a)-[*]-(b) )  
RETURN s
```

También allShortestPaths

Si se especifica dirección de arco, todos la deben cumplir

Más consultas

7. Muestre las personas conocidas de Roger Brooks que no participaron en ningún crimen.
8. Muestre el camino más corto de Judith Moore a Richard Green.
9. Encuentre los oficiales que investigaron los crímenes cometidos en 165 Laurel Street.

Funciones de agregación

Al incluir funciones de agregación en el
RETURN funciona similar a SQL

Mismas funciones!

Devolver el año del auto más nuevo.

```
MATCH (n:Vehicle) RETURN max(n.year)
```

Agrupamiento

Si se devuelven propiedades con función de agregación, se agrupa por ellas:

Devolver cuantos vehículos marca Ford hay en la base

MATCH (a:Vehicle)

WHERE a.make **CONTAINS** "Ford"

RETURN a.make, **COUNT(*)**

Agrupamiento

Para los 10 marcas de autos con más unidades
mostrar el promedio de año y cantidad.

MATCH (a:Vehicle)

RETURN a.make, AVG(toInteger(a.year)), COUNT(a)

ORDER BY COUNT(a) **DESC**

LIMIT 10

Agrupamiento

"a.make"	"AVG(apoc.convert.toInteger(a.year))"	"COUNT(a)"
"Ford"	1998.564705882353	85
"Chevrolet"	2001.911764705882	68
"Dodge"	2001.072727272728	55
"Toyota"	2003.244897959184	49
"Pontiac"	1989.9772727272725	44
"Nissan"	2002.9302325581396	43
"Mazda"	1999.7837837837837	37
"GMC"	2000.054054054054	37
"Volkswagen"	1998.861111111111	36
"Mercedes-Benz"	1999.7222222222222	36

WITH

Reemplaza al RETURN

Permite manipular resultados antes de pasar a siguiente parte de la consulta

Se puede usar para filtrar agregación

Hacer la misma consulta anterior, pero mostrando sólo las marcas que tienen mas de 45 vehículos en la base.

MATCH (a:Vehicle)

WITH a.make as marca, **AVG**(toInteger(a.year)) as prom_year
, **COUNT**(a) **AS** cant

WHERE cant > 45

RETURN marca, prom_year,cant

WITH

"marca"	"prom_year"	"cant"
"Toyota"	2003.244897959184	49
"Ford"	1998.564705882353	85
"Dodge"	2001.072727272728	55
"Chevrolet"	2001.911764705882	68

WITH – multiples MATCH

Se pueden hacer nuevos MATCH en base al resultado del primero

Devolver los autos más nuevos de la base.

MATCH (a:Vehicle)

WITH MAX(a.year) AS maxyear

MATCH (v:Vehicle) **WHERE** v.year = maxyear

RETURN v.make,v.model,v.year

WITH – multiples MATCH

"v.make"	"v.model"	"v.year"
"Chevrolet"	"Tahoe"	"2013"
"Nissan"	"Altima"	"2013"
"Volvo"	"C70"	"2013"
"BMW"	"X6"	"2013"

Últimas consultas

10. Obtenga el modelo, marca y año del auto más viejo de la base.
11. ¿A qué distancia se encuentra el auto más viejo de Roger Brooks?
12. Devuelva el nombre y apellido de personas que conozcan más de 10 personas.
13. Cuantas personas hay en la base? Cuantos tiene teléfono? Cuantos mail?

Consultas interesantes



Totales por tipo de crimen

```
MATCH (c:Crime)  
RETURN c.type AS crime_type, count(c) AS total  
ORDER BY count(c) DESC
```

"crime_type"	"total"
"Violence and sexual offences"	8765
"Public order"	4839
"Criminal damage and arson"	3587
"Burglary"	2807
"Vehicle crime"	2598
"Other theft"	2140
"Shoplifting"	1427
"Other crime"	651
"Robbery"	541
"Theft from the person"	423
"Bicycle theft"	414
"Drugs"	333
"Possession of weapons"	236
null	1

Crimenes por locacion

```
MATCH (l:Location)<-[ :OCCURRED_AT ]-(:Crime)
RETURN l.address AS address, l.postcode
AS postcode, count(l) AS total
ORDER BY count(l) DESC
LIMIT 15
```

"address"	"postcode"	"total"
"Piccadilly"	"M1 1LU"	166
"Shopping Area"	"M60 1TA"	111
"Prison"	"M60 9AH"	48
"Shopping Area"	"M4 3AL"	46
"Nightclub"	"M1 3LZ"	41
"Parking Area"	"M90 2AY"	38
"Supermarket"	"WN7 5SJ"	38
"Nightclub"	"M4 2BS"	36
"182 Waterson Avenue"	"M40 9BY"	35
"43 Walker's Croft"	"M3 1DA"	35
"Nightclub"	"M1 3WB"	34
"Parking Area"	"M1 3HF"	33
"Bus/Coach Station"	"OL1 1QN"	32
"136 A5185"	"M6 8FQ"	30
"Hospital"	"M13 9WL"	30

Crimenes cerca de una dirección particular

```
MATCH (l:Location {address: '1 Coronation Street', postcode: 'M5 3RW'})  
WITH point(l) AS corrie  
MATCH (x:Location)-[:HAS_POSTCODE]->(p:PostCode),  
(x)<-[ :OCCURRED_AT ]-(c:Crime)  
WITH x, p, c, point.distance(point(x), corrie) AS distance  
WHERE distance < 500  
RETURN x.address AS address, p.code AS postcode, count(c) AS crime_total,  
collect(distinct(c.type)) AS crime_type, distance  
ORDER BY distance  
LIMIT 10
```

Crimenes cerca de una dirección particular

"address"	"postcode"	"crime_total"	"crime_type"	"distance"
"1 Coronation Street"	"M5 3RW"	3	["Bicycle theft", "Violence and sexual offences", "Public order"]	0.0
"158 Gloucester Street"	"M5 3SG"	1	["Public order"]	104.1354489517625
"147 West Crown Avenue"	"M5 3WT"	3	["Public order", "Shoplifting"]	162.54121621528688
"170 Chancel Avenue"	"M5 3SJ"	1	["Violence and sexual offences"]	165.7102054854755
"157 Parish View"	"M5 3PA"	1	["Burglary"]	175.47453728972013
"65 West Bank Street"	"M5 3GY"	1	["Public order"]	237.15914155050027
"160 Carmel Close"	"M5 3LR"	1	["Burglary"]	242.82251946889645
"66 Brassington Avenue"	"M5 3JX"	1	["Violence and sexual offences"]	279.8874106950031
"141 Garden Wall Close"	"M5 3GQ"	1	["Vehicle crime"]	289.88766629407894
"153 Robert Hall Street"	"M5 3PT"	2	["Public order", "Criminal damage and arson"]	296.7257174132251

Personas Vulnerables

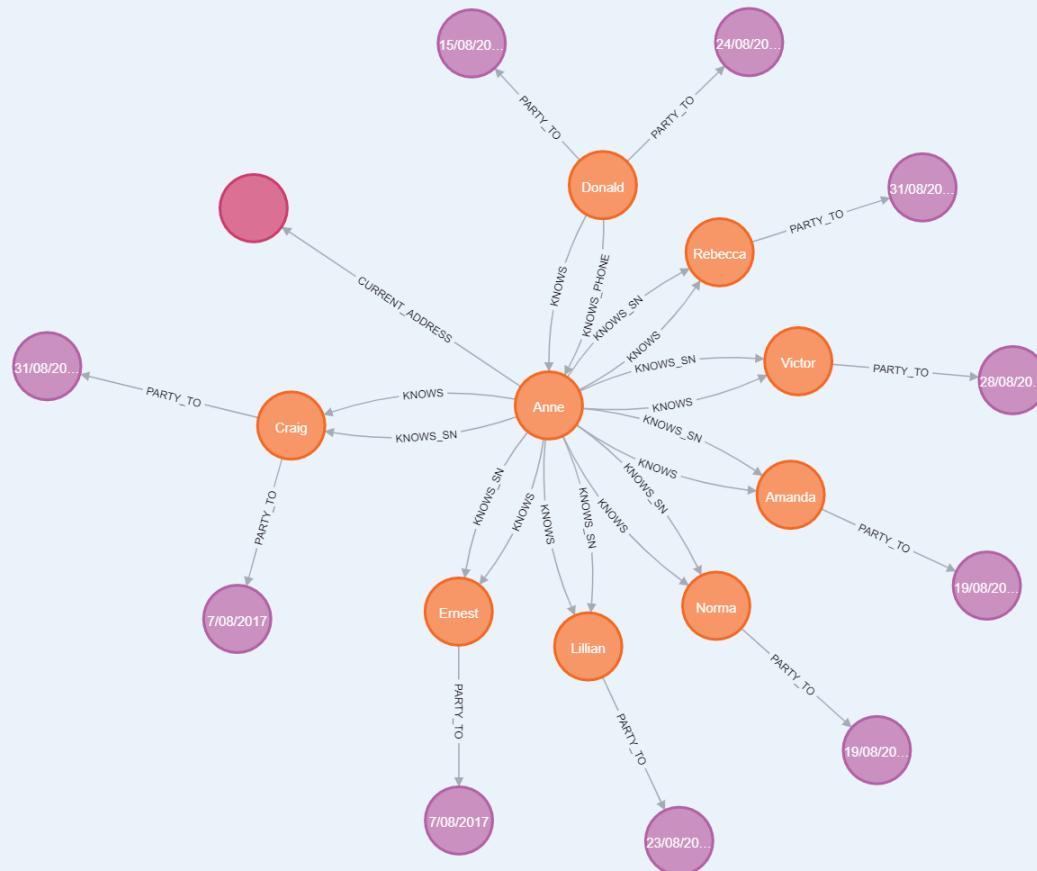
```
MATCH (p:Person)-[:KNOWS]-(friend)-[:PARTY_TO]->(:Crime)
WHERE NOT (p:Person)-[:PARTY_TO]->(:Crime)
RETURN p.name AS name, p.surname AS surname, p.nhs_no AS id,
count(distinct friend) AS dangerousFriends
ORDER BY dangerousFriends DESC
LIMIT 5
```

"name"	"surname"	"id"	"dangerousFriends"
"Anne"	"Freeman"	"804-54-6976"	8
"Bonnie"	"Gilbert"	"622-53-3302"	7
"Ashley"	"Robertson"	"554-93-4466"	5
"Kathy"	"Wheeler"	"218-31-0921"	3
"Pamela"	"Gibson"	"838-11-7607"	2

Los amiguitos de Anne

MATCH path = (:Location)<-[{:CURRENT_ADDRESS}]-(:Person {nhs_no: '804-54-6976', surname: 'Freeman'})-[:KNOWS]-(:Person)-[:PARTY_TO]->(:Crime)

RETURN path



Centralidad

```
CALL gds.betweenness.stream('social')
YIELD nodeId, score AS centrality
WITH gds.util.asNode(nodeId) AS node, centrality
RETURN node.name AS name, node.surname AS surname,
node.nhs_no AS id, toInteger(centrality) AS score
ORDER BY centrality DESC
LIMIT 10;
```

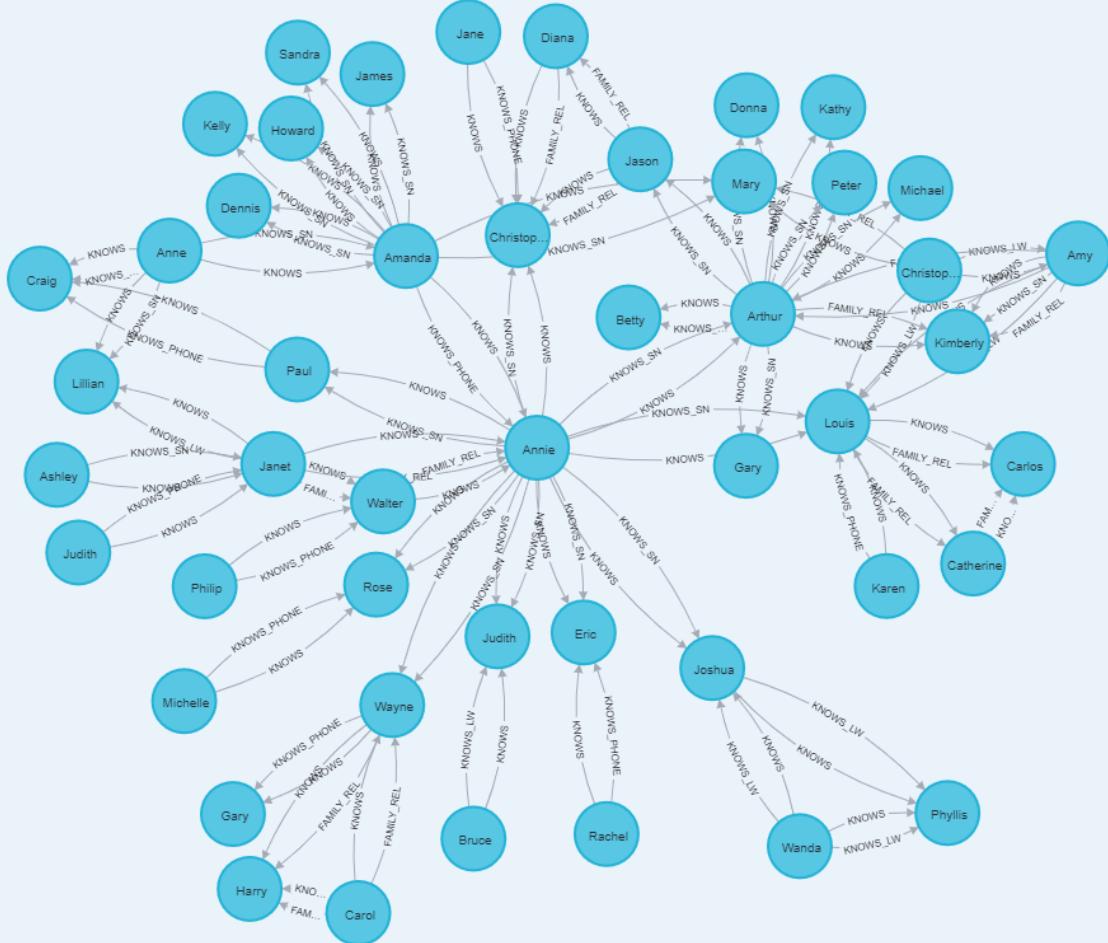
Centralidad

"name"	"surname"	"id"	"score"
"Annie"	"Duncan"	"863-96-9468"	5275
"Ann"	"Fox"	"576-99-9244"	5116
"Amanda"	"Alexander"	"893-63-6176"	4599
"Bruce"	"Baker"	"576-82-7408"	4193
"Andrew"	"Foster"	"214-77-6416"	3693
"Anne"	"Rice"	"612-83-6356"	3418
"Alan"	"Hicks"	"852-52-0933"	3347
"Amy"	"Murphy"	"367-54-3328"	3282
"Adam"	"Bradley"	"237-02-1263"	3275
"Arthur"	"Willis"	"271-78-8919"	3259

Centralidad

```
MATCH path = (:Person {nhs_no: '863-96-9468', surname: 'Duncan'})-[:KNOWS*..2]-(:Person)
```

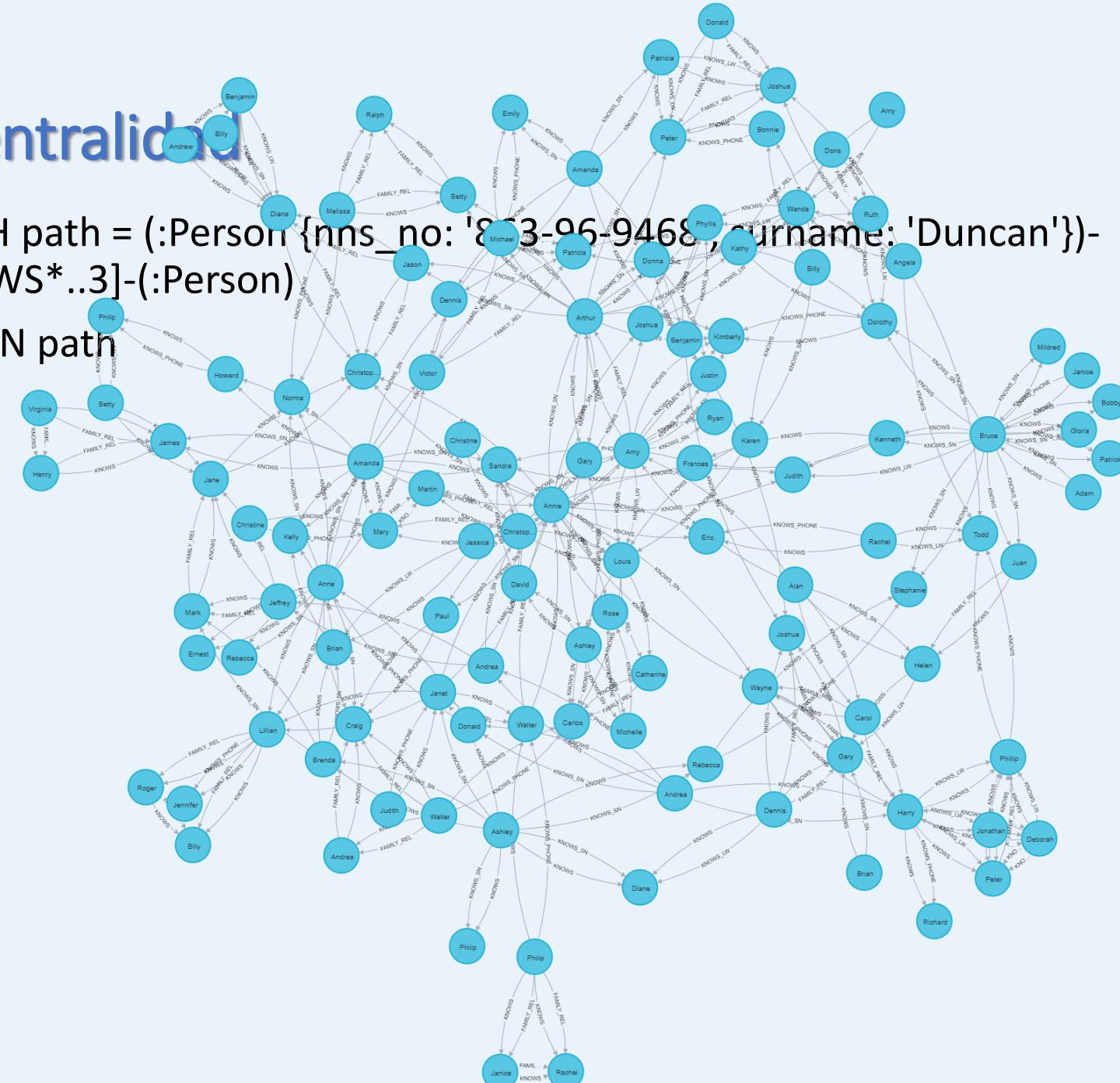
RETURN path



Centralide

```
MATCH path = (:Person {nns_no: '863-96-9468', surname: 'Duncan'})-[:KNOWS*..3]-(:Person)
```

RETURN path



ABM



ABMs - Altas

Diferente para nodos y arcos

El arco debe tener una dirección

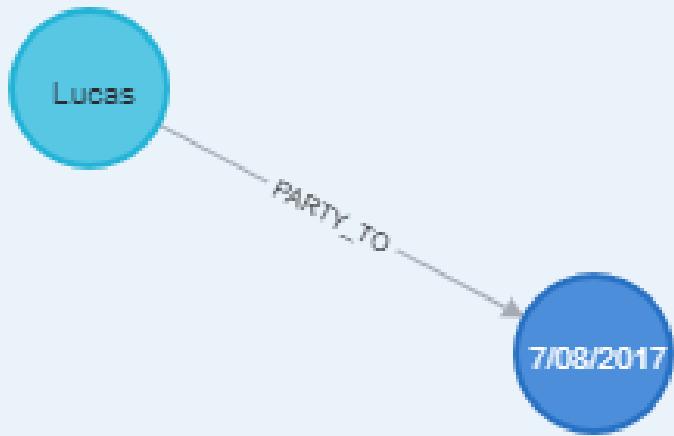
CREATE (a:Person {name: "Lucas",surname:"Roma"})

MATCH

(a:Person{name:"Lucas",surname:"Roma"}),(m:Crime
{id:"df4e954114d147be364d17d580cda1214ff694b61
4bdf76baa0a640d5bda74d3"})

CREATE (a)-[:PARTY_TO]->(m)

ABMs - Altas



ABMs - Modificaciones

Aplica a propiedades y etiquetas

```
MATCH (a:Person{name:"Lucas",surname:"Roma"})  
SET a.biography = "Reconocido delincuente ...."
```

```
MATCH (a:Person{name:"Lucas",surname:"Roma"})  
REMOVE a.biography
```

ABMs - Bajas

Borramos los arcos de un nodo

```
MATCH (a:Person{name:"Lucas",surname:"Roma"})-  
[r]-()  
DELETE r
```

Borramos al nodo

```
MATCH (a:Person{name:"Lucas",surname:"Roma"})  
DELETE a
```

Gracias por su atención

