



*em Grafos*



Daniel Lins da Silva



daniel.lins@gmail.com



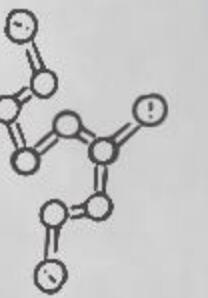
<https://github.com/daniellins/>

# Agenda



Banco de grafos

Neo4j



Demonstração



Discussão



# Quem sou eu?



Daniel Lins

## Me. em Engenharia da Computação

Doutorando em Computação pela POLI-USP.

LAA (Laboratório de Automação Agrícola da USP), BioComp (Núcleo de Pesquisa em Biodiversidade e Computação) e Grupo de Estudos, Pesquisa e Extensão em Big Data.

# O que são grafos?

$$G(V,E)$$

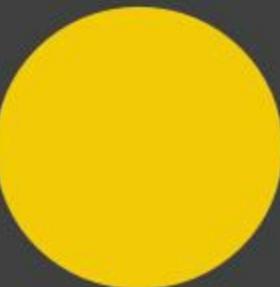
onde  $V$  é um conjunto não vazio de objetos denominados vértices e  $E$  é um subconjunto de pares não ordenados de  $V$ , chamados arestas.

# O que são grafos?



# ...e Banco de Dados de Grafos?

É um sistema que armazena dados em estruturas de grafos, permitindo também o armazenamento explícito do relacionamento entre eles.



“

MAS PORQUE UM  
BANCO DE GRAFOS?

”

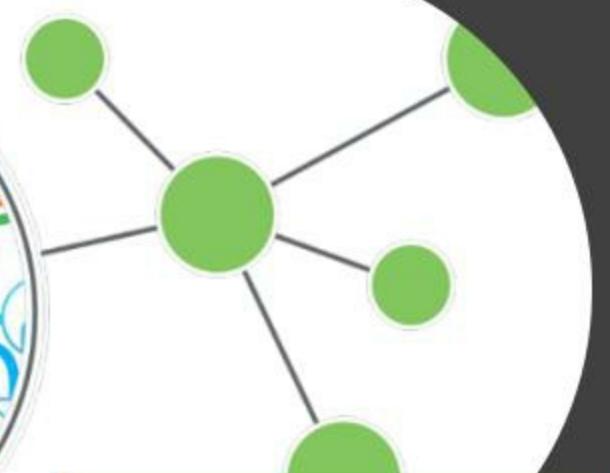




Tudo são  
relações!



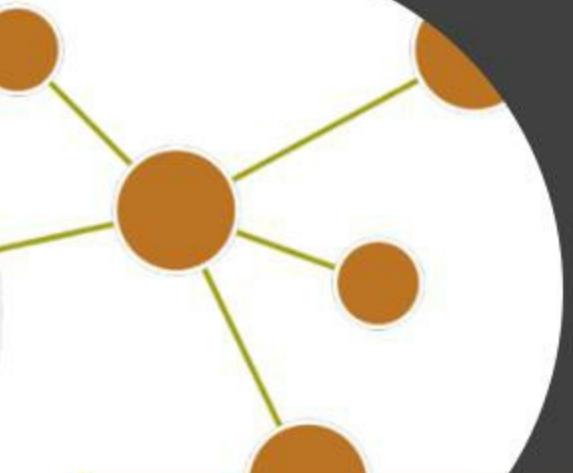
# Alguns Casos de Uso...



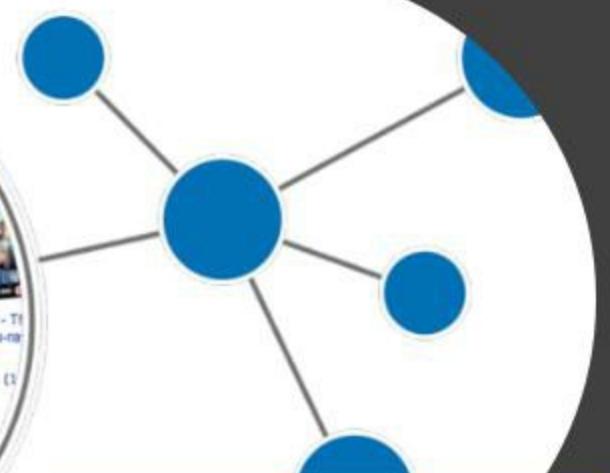
Redes Sociais



Análise de Redes



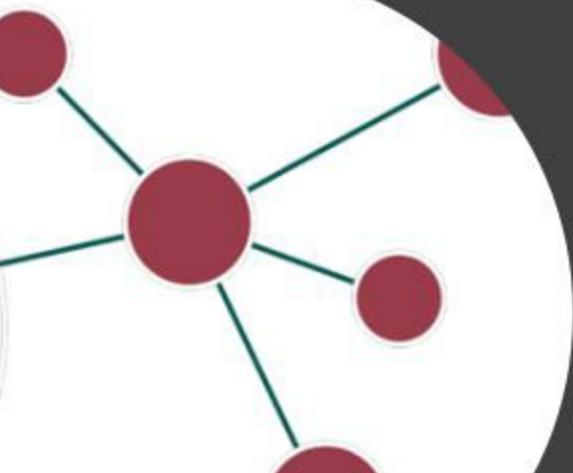
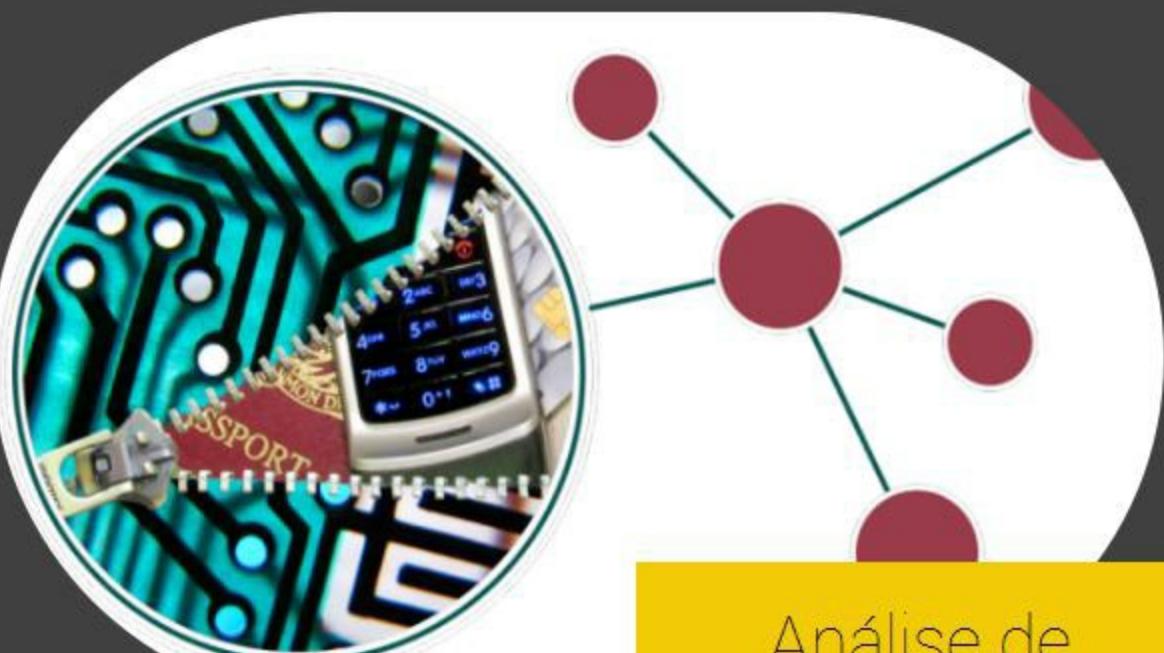
Definição de Rotas



Recomendações



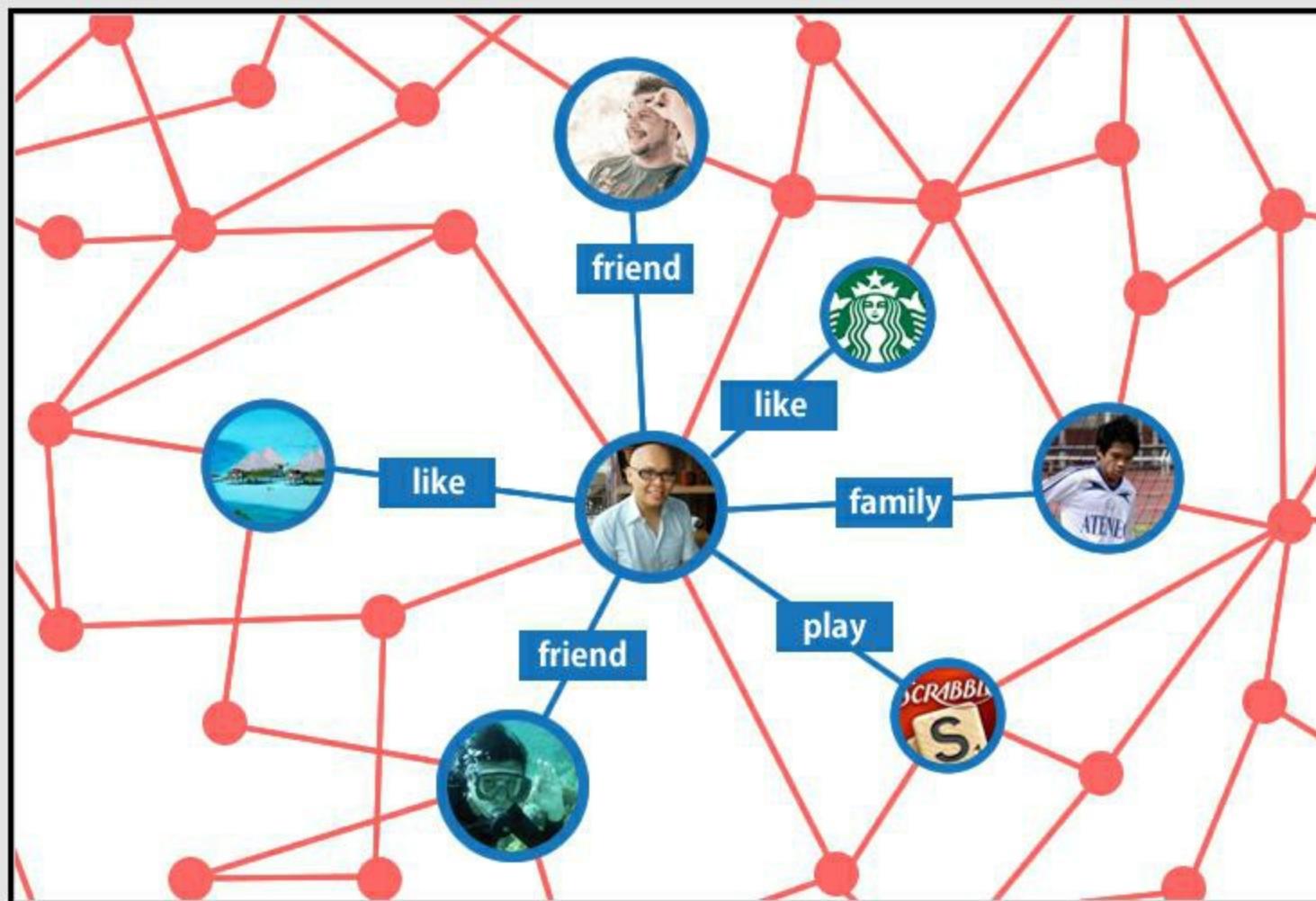
Análise Logística



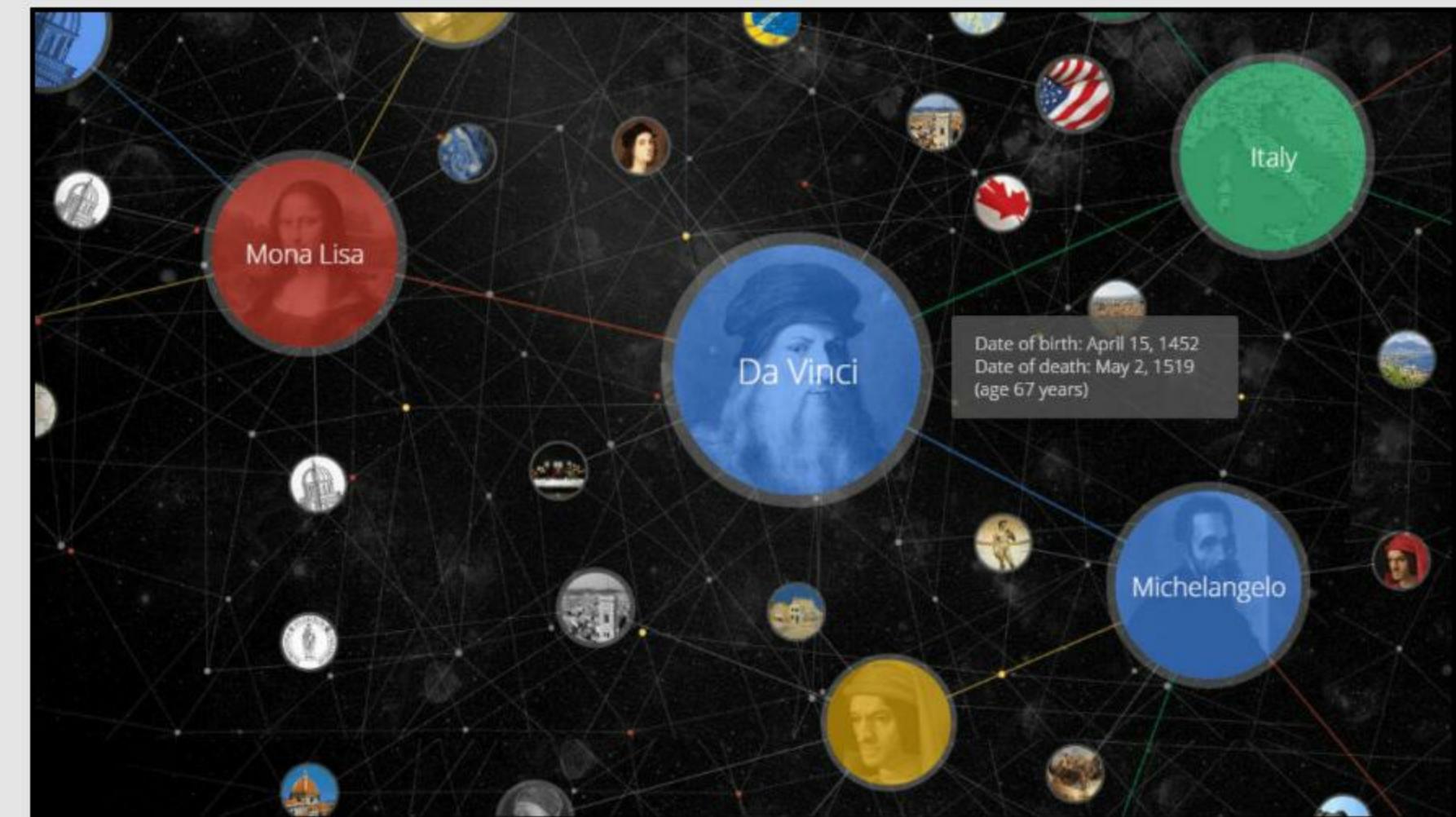
Análise de Fraudes

# Knowledge Graphs

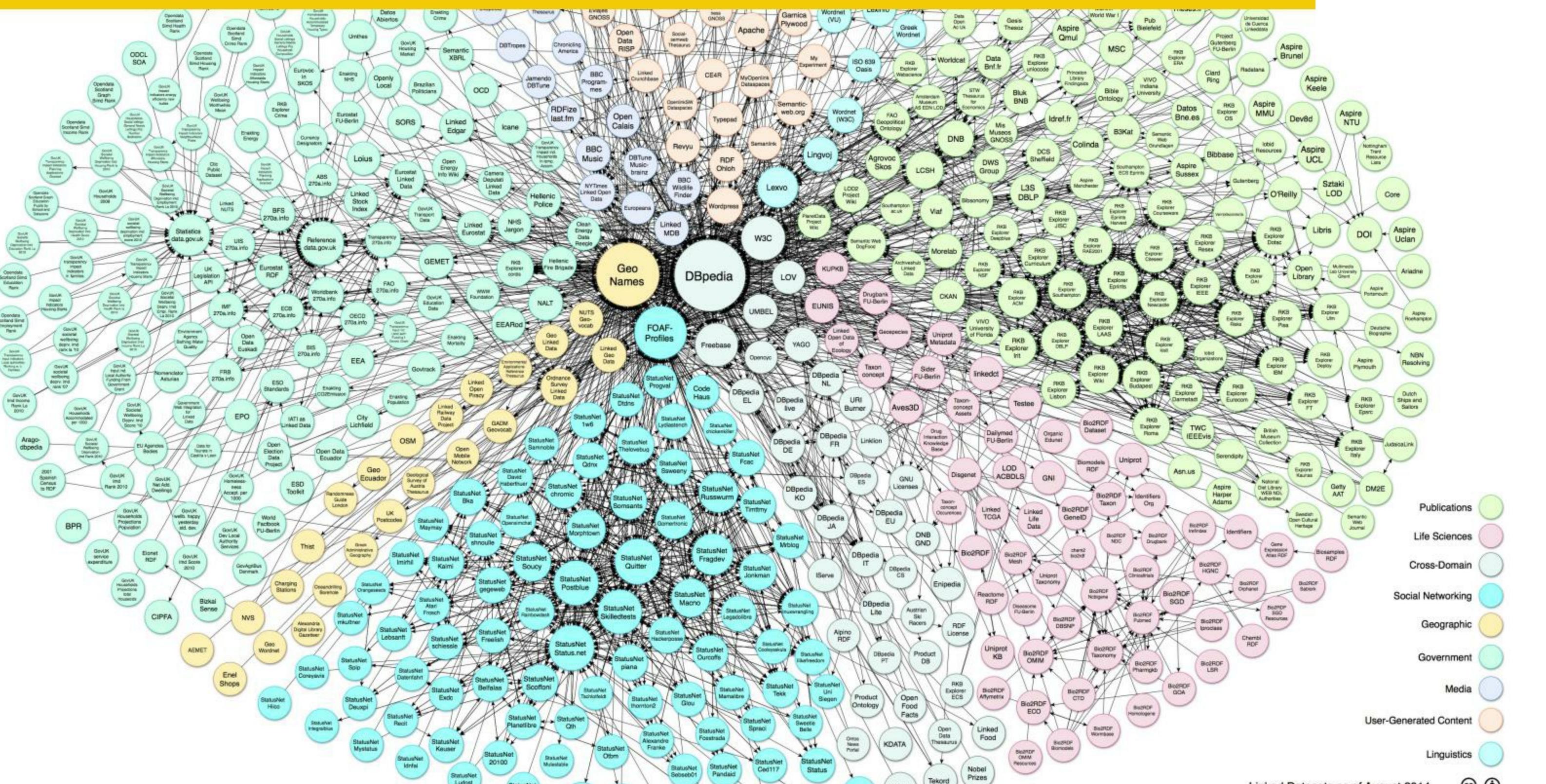
Facebook Graph



Google Graph



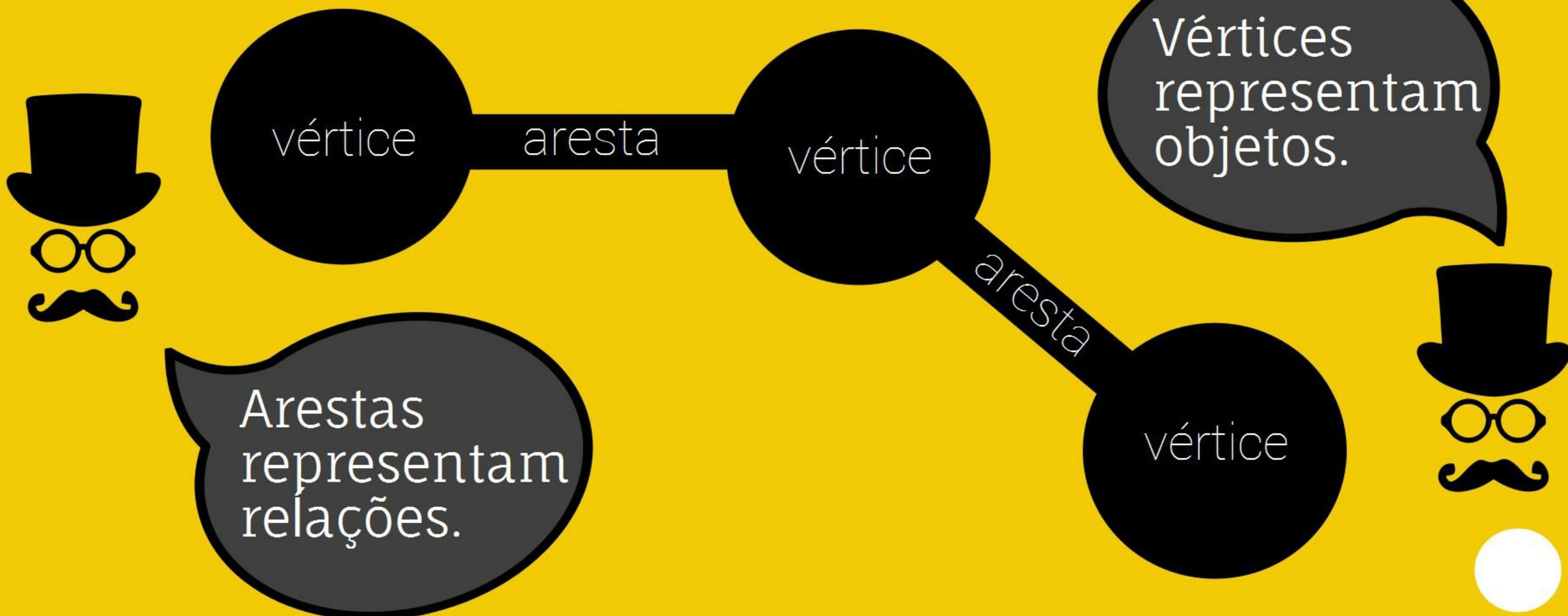
# Linked Open Data (LOD)



Linked Datasets as of August 2014

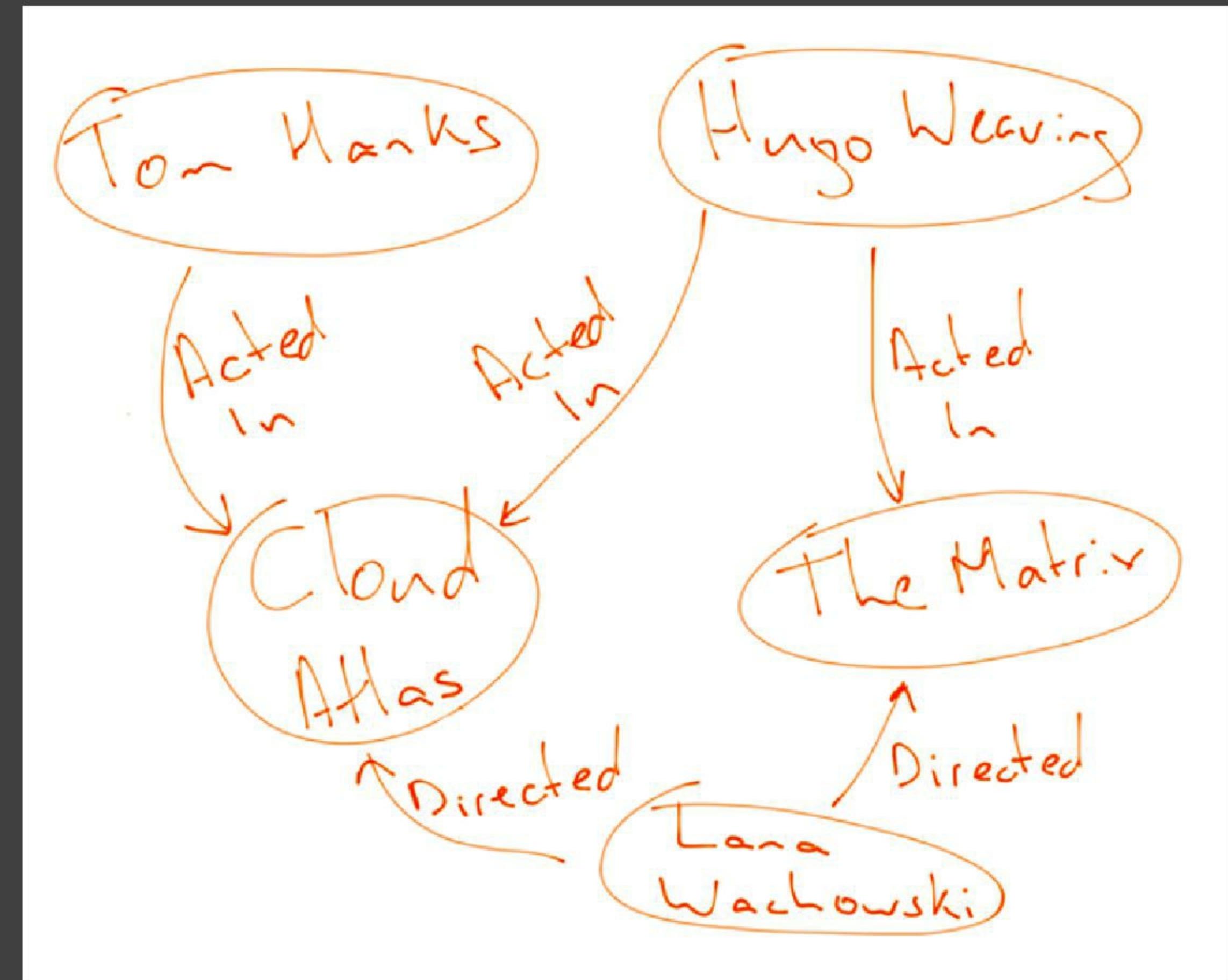


# Como organizar as informações?



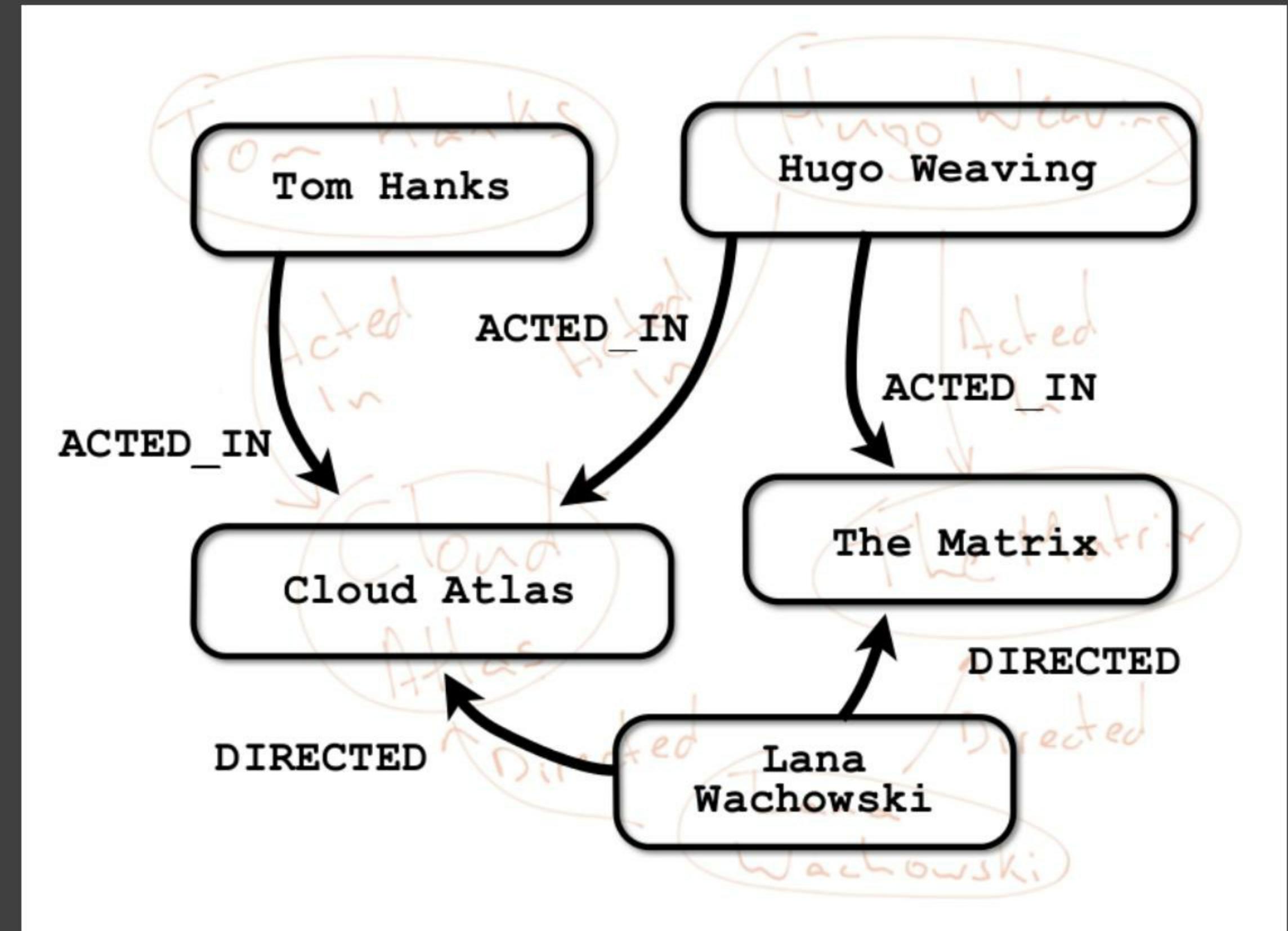


Desenhando  
um modelo...



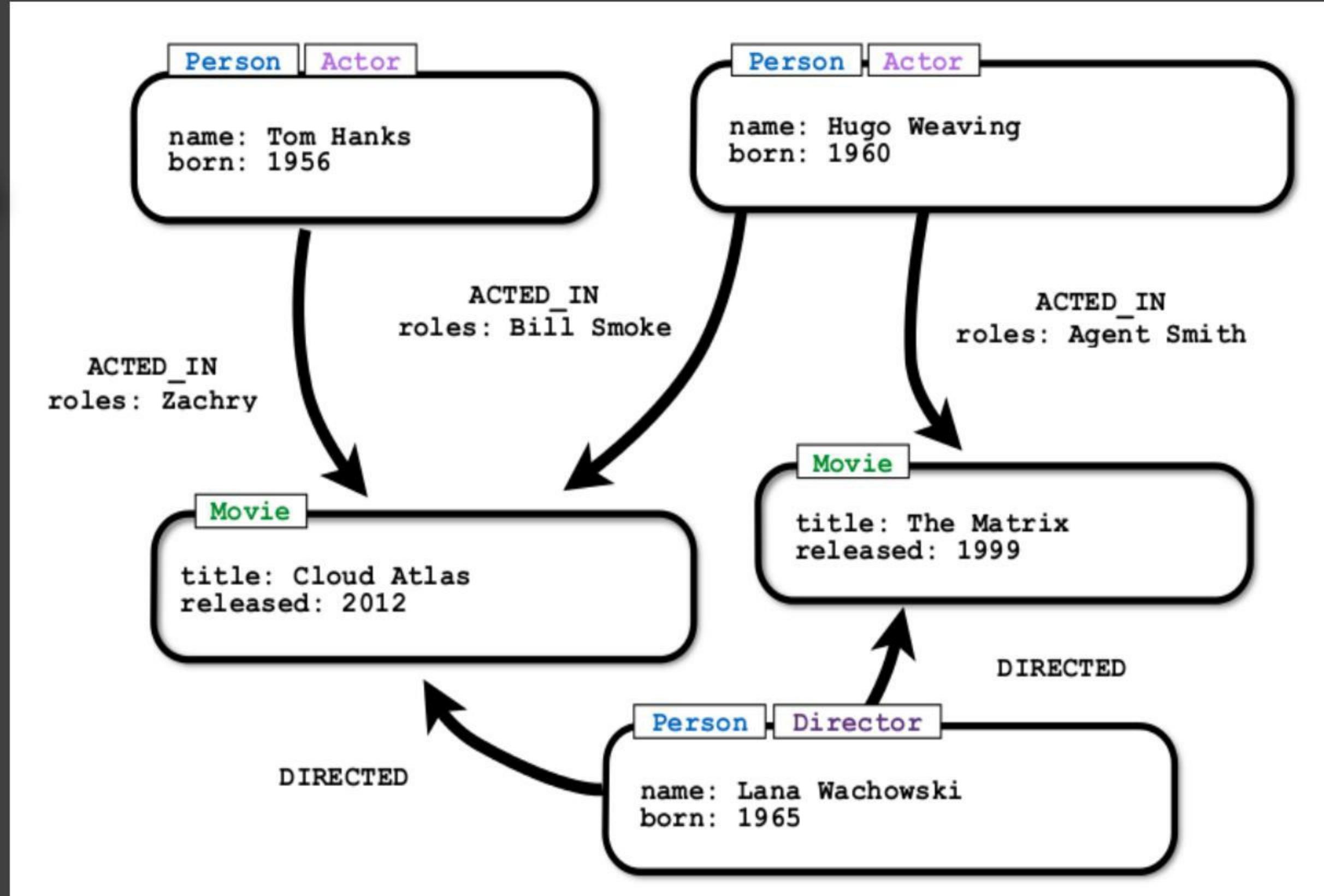


Desenhando  
um modelo...





Desenhando  
um modelo...





# Demonstração



neo4j <http://neo4j.com/>

Banco de Dados em Grafo nativo mais popular,  
de acordo com o DB Engines.

Fonte: <http://db-engines.com/en/ranking>

Implementado em Java.

Linguagem de consulta própria: Cypher.

Dados podem ser acessados a partir de uma  
API REST ou Drivers disponíveis em várias  
línguagens.



# Iniciando o Neo4j

> bin/neo4j start

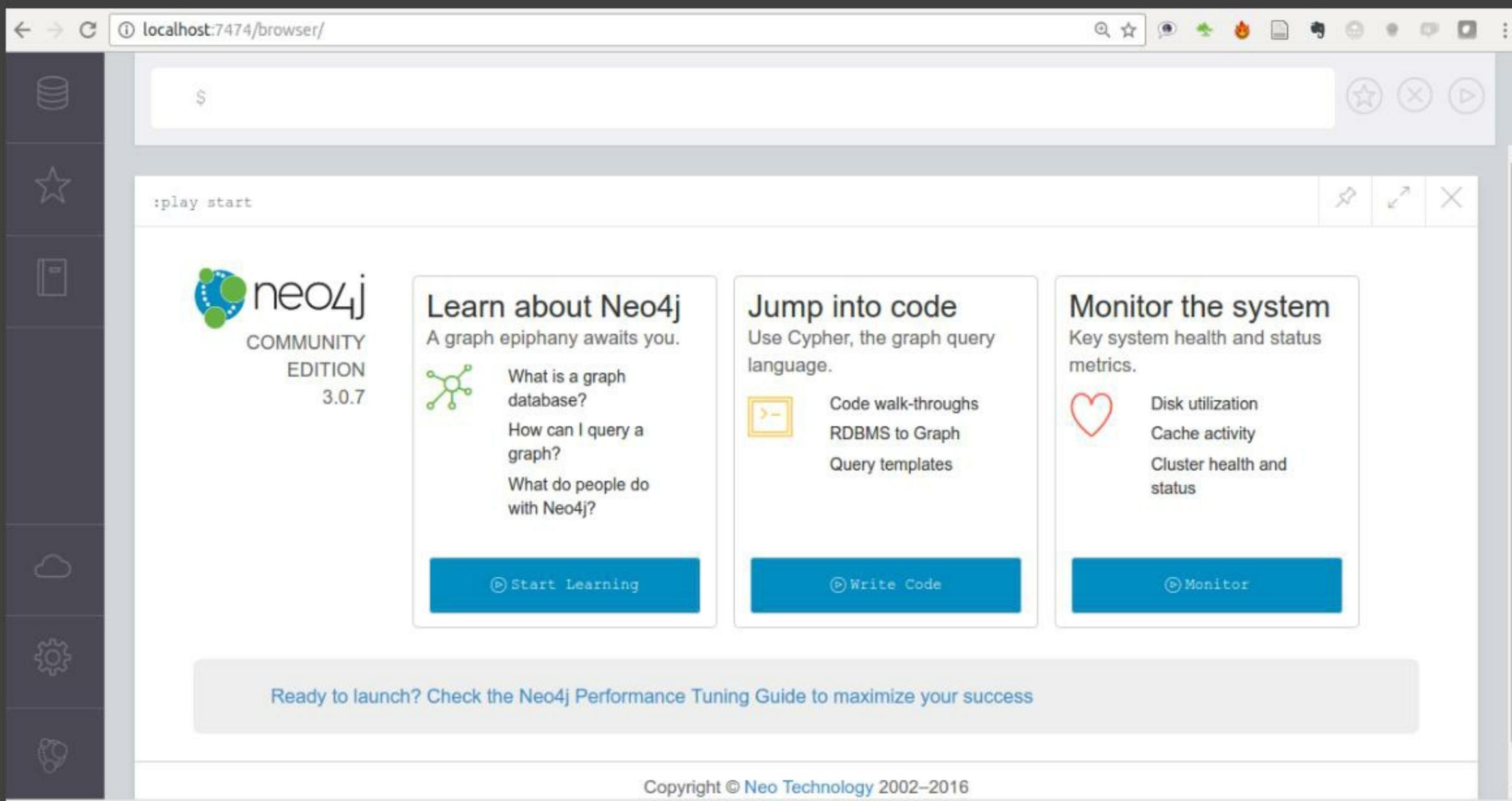
> bin/neo4j stop

> bin/neo4j status



# Acessando o Cliente Web

Acessar o endereço <http://localhost:7474/>



The screenshot shows the Neo4j Browser interface running in a web browser. The address bar at the top displays "localhost:7474/browser/". The interface has a dark theme with a sidebar on the left containing icons for database, star, file, cloud, and settings. The main area features a search bar with placeholder "\$" and a command input field showing ":play start". Below this is a section for "neo4j COMMUNITY EDITION 3.0.7" with the Neo4j logo. Three main cards are displayed: "Learn about Neo4j" (with sub-links for "What is a graph database?", "How can I query a graph?", and "What do people do with Neo4j?"), "Jump into code" (with sub-links for "Code walk-throughs", "RDBMS to Graph", and "Query templates"), and "Monitor the system" (with sub-links for "Disk utilization", "Cache activity", and "Cluster health and status"). At the bottom, a message says "Ready to launch? Check the Neo4j Performance Tuning Guide to maximize your success" and the footer includes "Copyright © Neo Technology 2002–2016".



# neo4j Caso não tenha o Neo4j...

Acessar o endereço <http://console.neo4j.org/>

Graph Setup:

```
create (Neo:Crew {name:'Neo'}), (Morpheus:Crew {name: 'Morpheus'}), (Trinity:Crew {name: 'Trinity'}), (Cypher:Crew:Matrix {name: 'Cypher'}), (Smith:Matrix {name: 'Agent Smith'}), (Architect:Matrix {name:'The Architect'}),  
(Neo)-[:KNOWS]->(Morpheus), (Neo)-[:LOVES]->(Trinity), (Morpheus)-[:KNOWS]->(Trinity),  
(Morpheus)-[:KNOWS]->(Cypher), (Cypher)-[:KNOWS]->(Smith), (Smith)-[:CODED_BY]->(Architect)
```

Query:

```
match (n:Crew) -[r:KNOWS*]-(m) where n.name='Neo' return n as Neo,r,m
```

Neo	r	m
(0:Crew {name:"Neo"})	[(0)-[0:KNOWS]->(1)]	(1:Crew {name:"Morpheus"})
(0:Crew {name:"Neo"})	[(0)-[0:KNOWS]->(1), (1)-[2:KNOWS]->(2)]	(2:Crew {name:"Trinity"})
(0:Crew {name:"Neo"})	[(0)-[0:KNOWS]->(1), (1)-[3:KNOWS]->(3)]	(3:Crew:Matrix {name:"Cypher"})
(0:Crew {name:"Neo"})	[(0)-[0:KNOWS]->(1), (1)-[3:KNOWS]->(3), (3)-[4:KNOWS]->(4)]	(4:Matrix {name:"Agent Smith"})

Query took 14 ms and returned 4 rows. [Result Details](#)

```
graph TD; Neo -- KNOWS --> Morpheus; Neo -- LOVES --> Trinity; Neo -- KNOWS --> Cypher; Morpheus -- KNOWS --> Trinity; Morpheus -- KNOWS --> Cypher; Cypher -- KNOWS --> AgentSmith[Agent Smith]; AgentSmith -- CODED_BY --> Architect[The Architect]
```

You can modify and query this graph by entering statements in the input field at the bottom.  
 For some syntax help hit the [Help](#) button.  
 If you want to share your graph, just do it with [Share](#)

[Match \(n:Crew\) -\[r:KNOWS\\*\]-\(m\) where n.name='Neo' return n as Neo,r,m](#)



## Criando um nó

```
CREATE (p:Person { name: "Daniel", from:  
"Brasil", age: 34 })
```

```
CREATE (p:Person { name: "Suelane", from:  
"Brasil", age: 30 })
```



# Criando um relacionamento

```
MATCH (daniel:Person), (suelane:Person)
WHERE daniel.name="Daniel"
AND suelane.name="Suelane"
CREATE (daniel)-[f:KNOWS]->(suelane)
```



# Consultando o grafo

```
MATCH (p:Person) WHERE p.name =  
"Daniel" RETURN p;
```



# neo4j Consultando todo o grafo

```
MATCH (n) RETURN n
```



# Incluindo uma nova propriedade

```
MATCH (p:Person{name:"Daniel"})-[r]-  
(s:Person{name:"Suelane"})  
SET r.since = 2013
```



neo4j

# Consultando padrões



Nó

Relação

Nó



# Consultando padrões

```
MATCH (p:Person)-[:KNOWS]-(friends)
WHERE p.name = "Daniel"
RETURN p, friends
```



# neo4j Apagando um Relacionamento

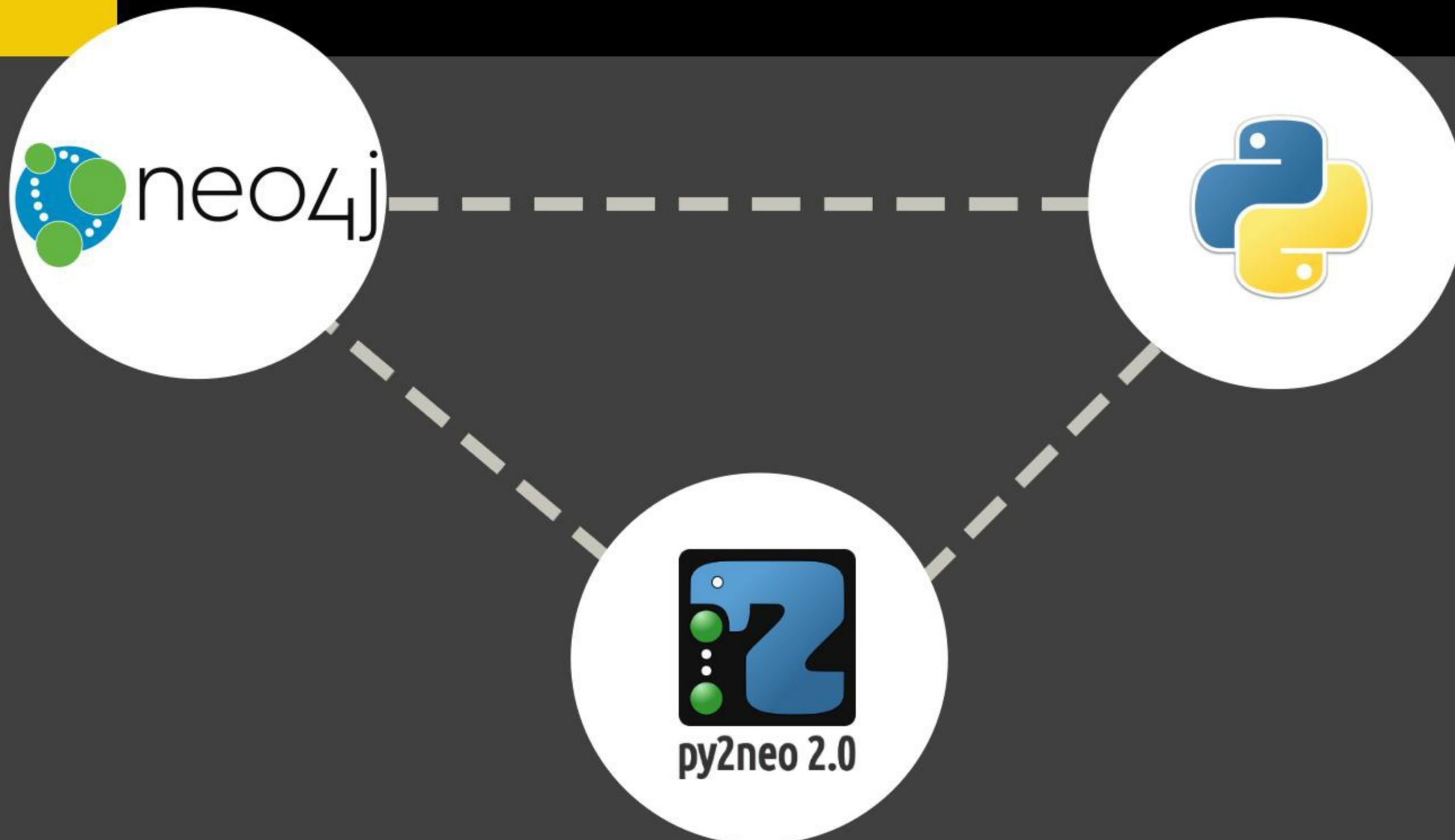
```
MATCH (d)-[rel:KNOWS]-(s) WHERE  
d.name='Daniel' AND s.name='Suelane'  
DELETE rel
```



# neo4j Apagando o Banco de Dados

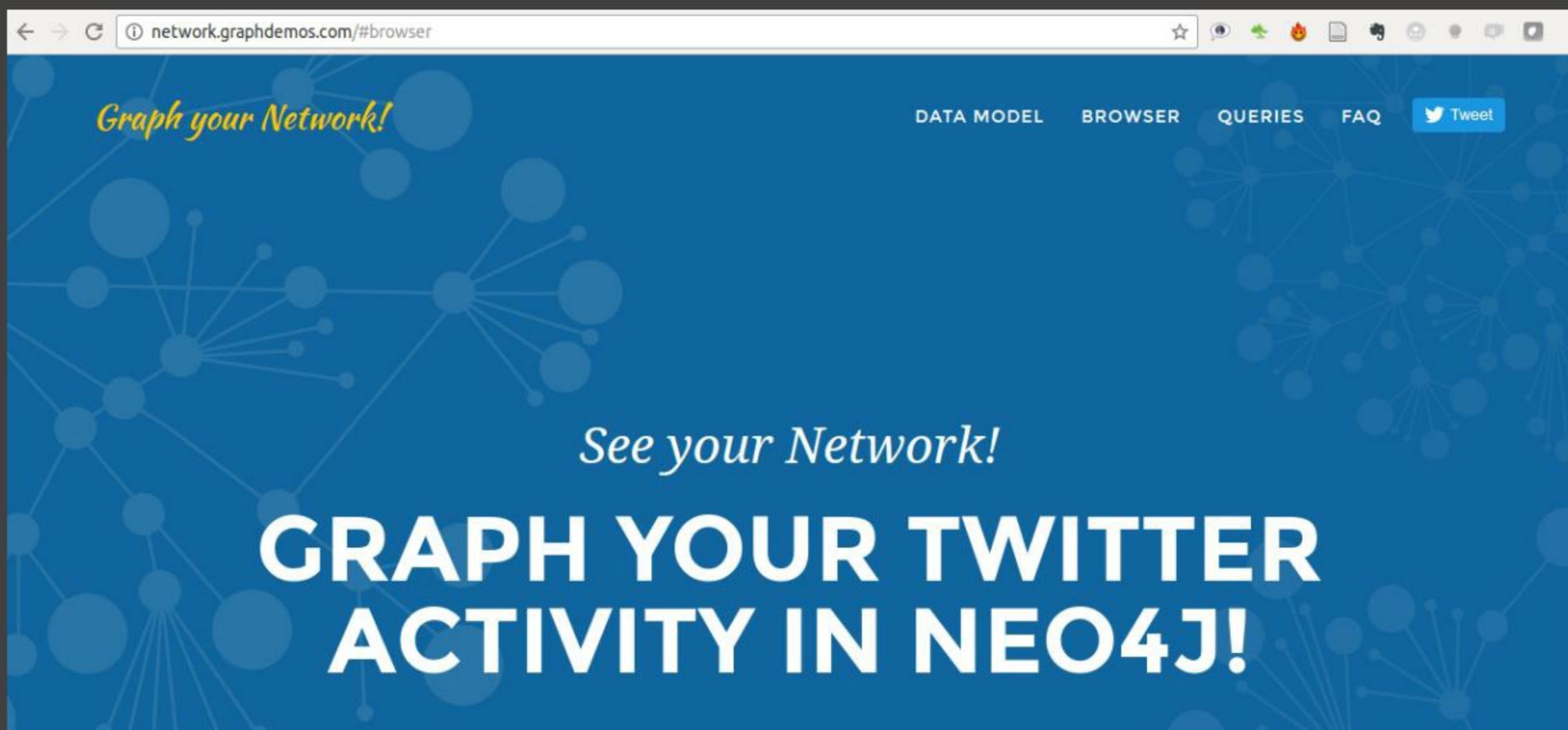
```
MATCH (n) DETACH DELETE n
```

# Demonstração 2



# Demonstração 2

<http://network.graphdemos.com/>



# Demonstração 3

<http://console.neo4j.org/?id=StarWars>

Graph Setup:

```
create (Neo:Crew {name:'Neo'}), (Morpheus:Crew {name: 'Morpheus'}), (Trinity:Crew {name: 'Trinity'}), (Cypher:Crew:Matrix {name: 'Cypher'}), (Smith:Matrix {name: 'Agent Smith'}), (Architect:Matrix {name:'The Architect'}), (Neo)-[:KNOWS]->(Morpheus), (Neo)-[:LOVES]->(Trinity), (Morpheus)-[:KNOWS]->(Trinity), (Morpheus)-[:KNOWS]->(Cypher), (Cypher)-[:KNOWS]->(Smith), (Smith)-[:CODED_BY]->(Architect)
```

Query:

```
match (n:Crew)-[r:KNOWS*]-(m) where n.name='Neo' return n as Neo,r,m
```

Neo	r	m
(0:Crew {name:"Neo"})	[(0)-[0:KNOWS]->(1)]	(1:Crew {name:"Morpheus"})
(0:Crew {name:"Neo"})	[(0)-[0:KNOWS]->(1), (1)-[2:KNOWS]->(2)]	(2:Crew {name:"Trinity"})
(0:Crew {name:"Neo"})	[(0)-[0:KNOWS]->(1), (1)-[3:KNOWS]->(3)]	(3:Crew:Matrix {name:"Cypher"})
(0:Crew {name:"Neo"})	[(0)-[0:KNOWS]->(1), (1)-[3:KNOWS]->(3), (3)-[4:KNOWS]->(4)]	(4:Matrix {name:"Agent Smith"})

Query took 14 ms and returned 4 rows. [Result Details](#)

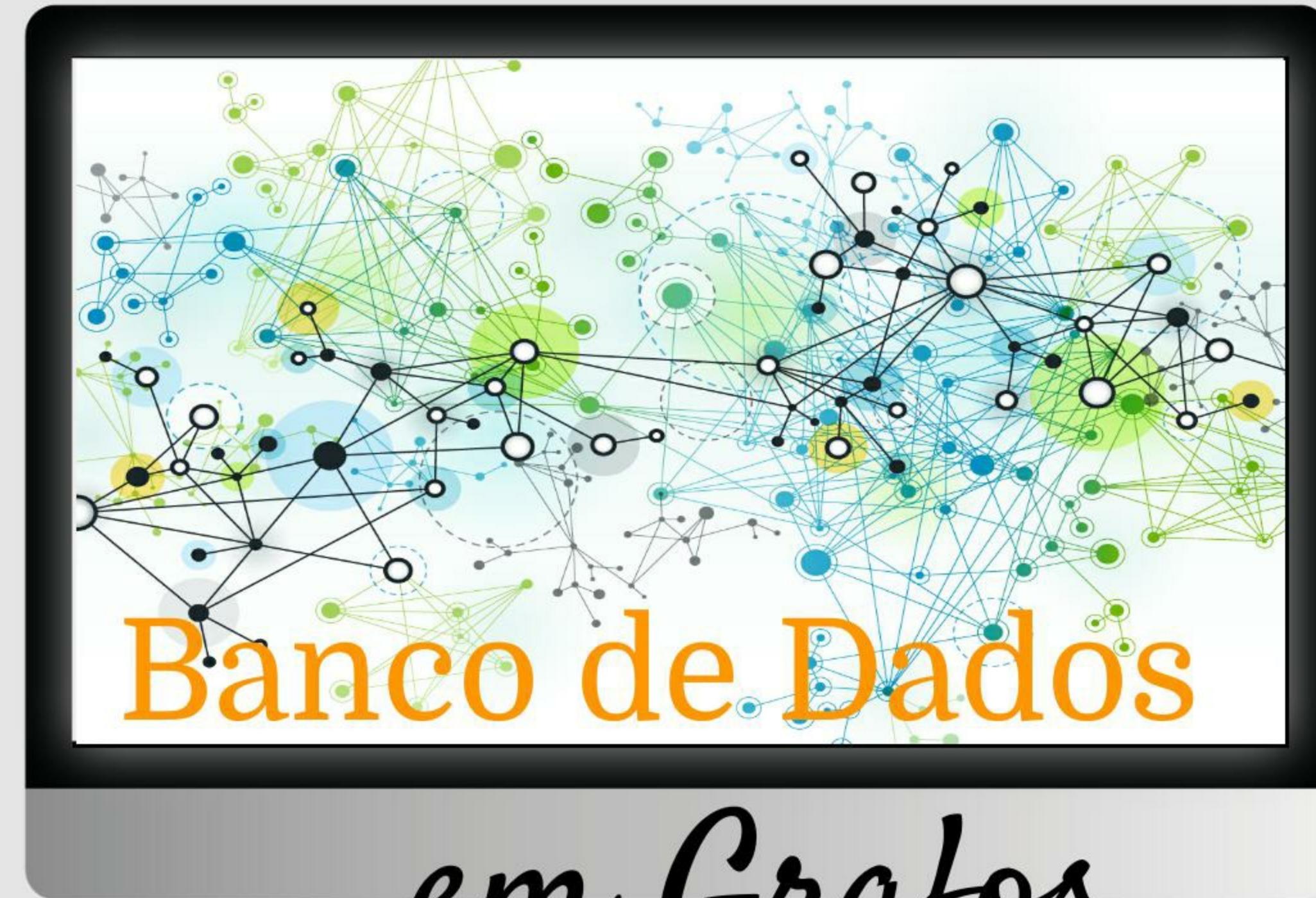
You can modify and query this graph by entering statements in the input field at the bottom.  
For some syntax help hit the [Help](#) button.  
If you want to share your graph, just do it with [Share](#)

[match \(n:Crew\)-\[r:KNOWS\\*\]-\(m\) where n.name='Neo' return n as Neo,r,m](#)

The screenshot shows a Neo4j browser window with a yellow header bar. The main area contains a graph visualization and a table of results. The graph has nodes for Neo, Morpheus, Trinity, Cypher, Agent Smith, and The Architect. Relationships include KNOWS and LOVES. The table lists the relationships from Neo to each of these nodes. A message at the bottom encourages modifying and querying the graph. A footer bar at the bottom has a play button icon.

# Para se divertir em casa

<https://neo4j.com/developer/guide-build-a-recommendation-engine/>



Daniel Lins da Silva  
 daniel.lins@gmail.com  
 <https://github.com/daniellins/>

*em Grafos*