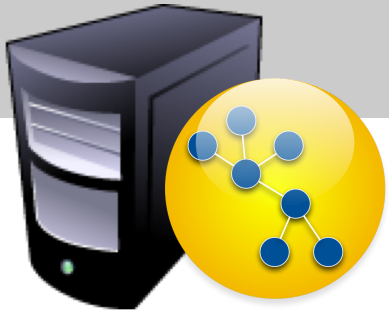
New Opportunities for Connected Data

@ianSrobinson
ian.robinson@neotechnology.com

#neo4j

Agenda



- Data complexity
- Graph databases – features and benefits
- Querying graph data
- Use cases

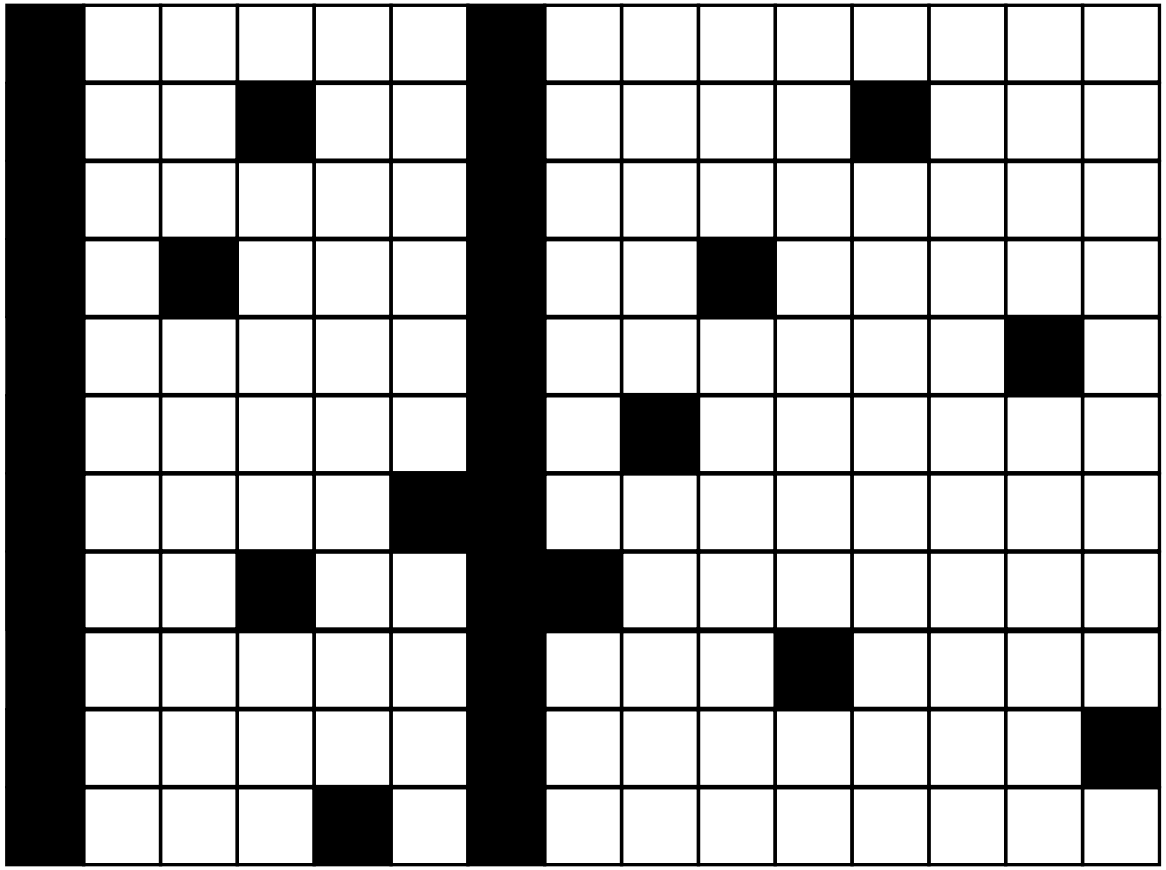
Data Complexity

complexity = f(size, semi-structure, connectedness)

Data Complexity

complexity = f(size, semi-structure, ***connectedness****)*

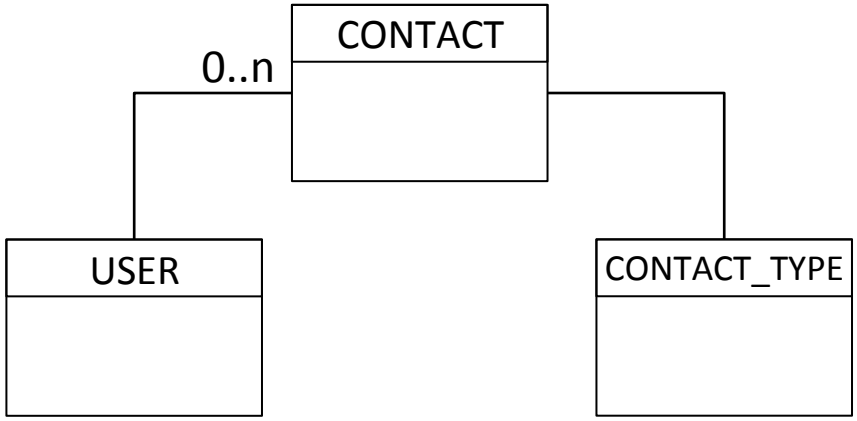
Semi-Structure



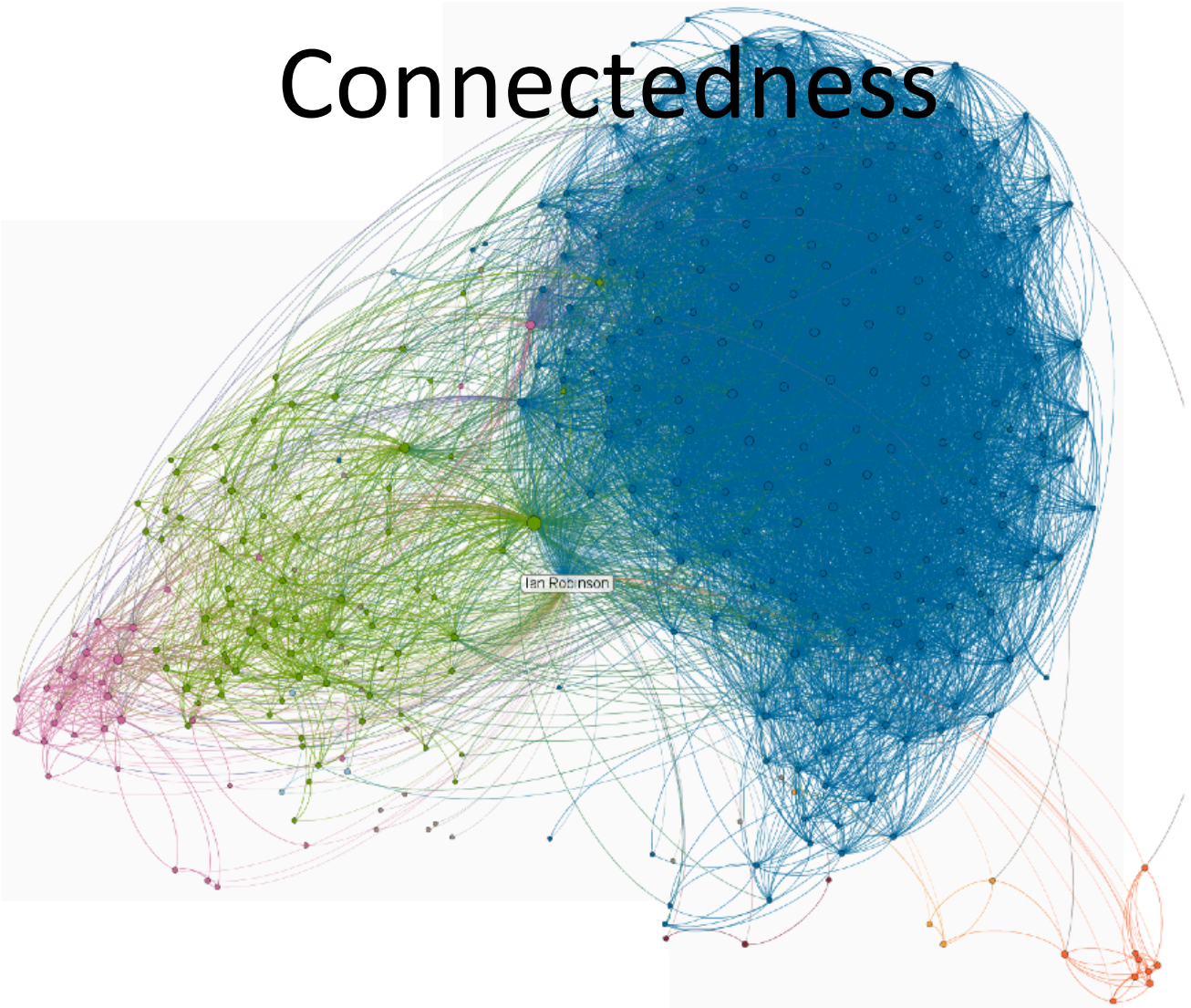
Semi-Structure

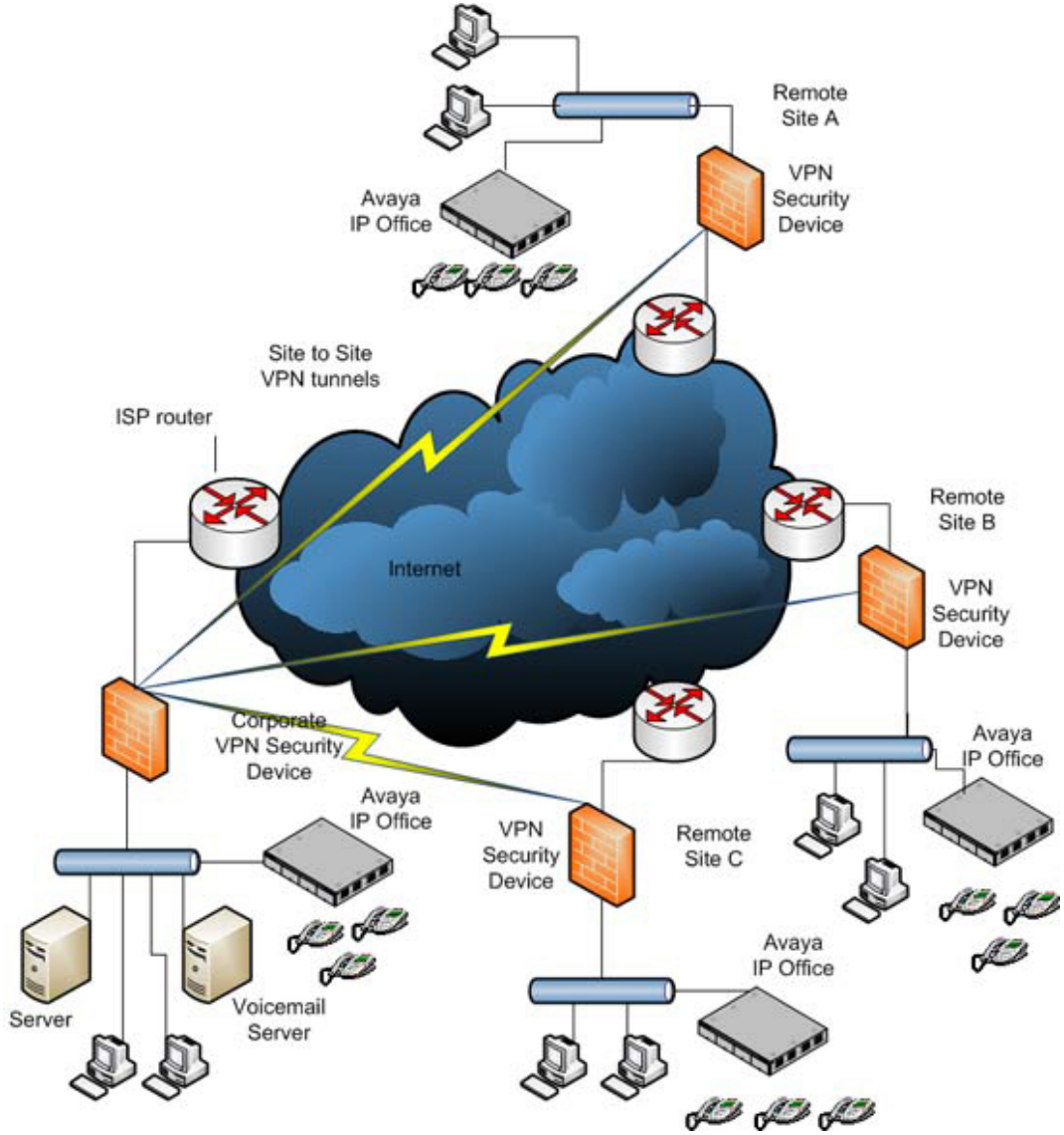
USER_ID	FIRST_NAME	LAST_NAME	EMAIL_1	EMAIL_2	FACEBOOK	TWITTER	SKYPE
315	Ian	Robinson	ian@neotechnology.com	iansrobinson@gmail.com	NULL	@iansrobinson	iansrobinson

Email: ian@neotechnology.com
Email: iansrobinson@gmail.com
Twitter: @iansrobinson
Skype: iansrobinson

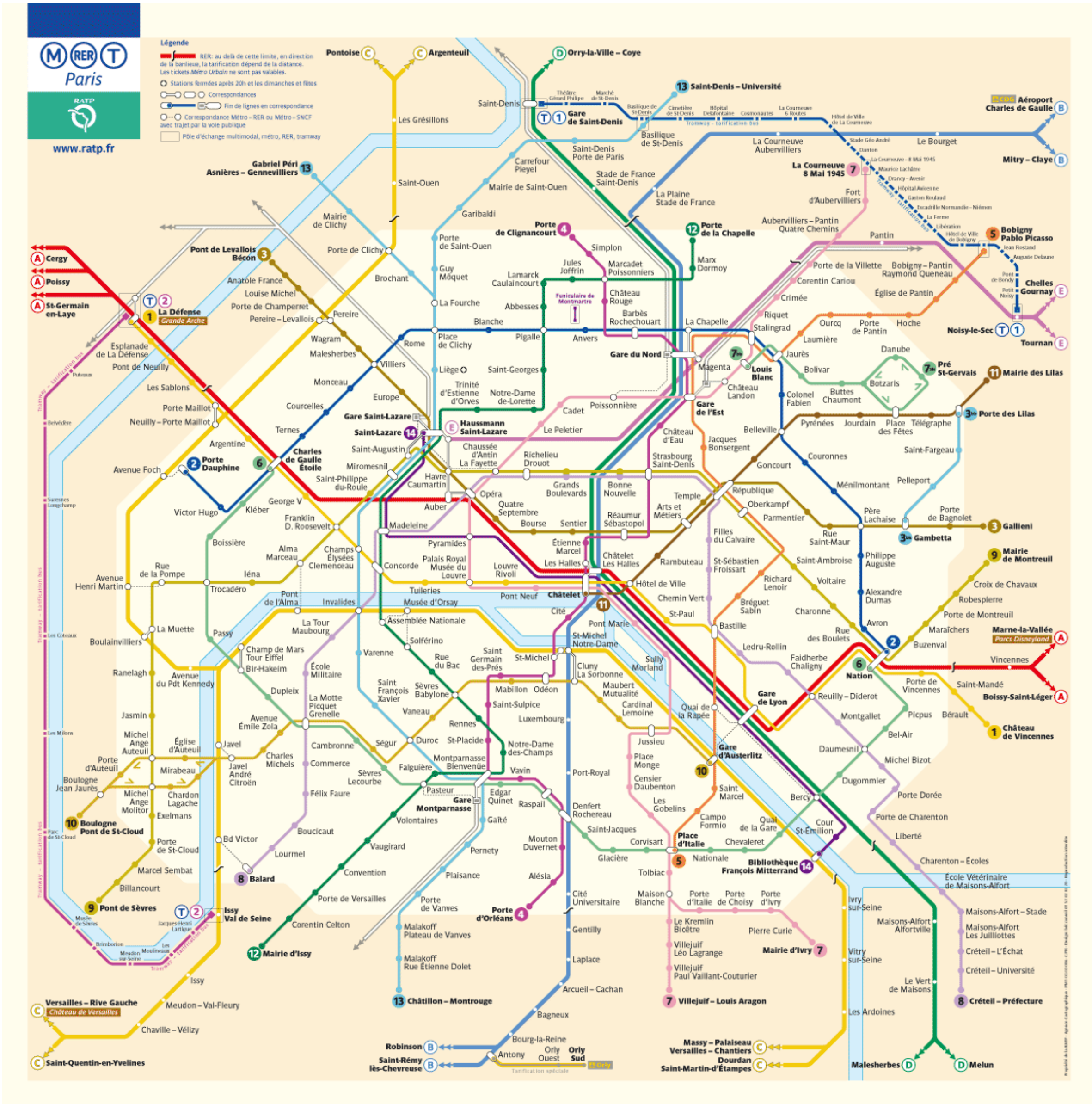


Connectedness




Remote
Site A
VPN
Security
Device
Avaya
IP Office
Site to Site
VPN tunnels
ISP router
Internet
Remote
Site B
VPN
Security
Device
Corporate
VPN Security
Device
Avaya
IP Office
VPN
Security
Device
Remote
Site C
Avaya
IP Office
Server
Voicemail
Server
Avaya
IP Office

#neo4j



Paris
www.ratp.fr
Légende
Pontoise
Argenteuil
Orry-la-Ville – Coye
Saint-Denis – Université
Aéroport Charles de Gaulle
Mitry – Claye
Gare de Saint-Denis
Gabriel Péri Asnières – Gennevilliers
Cergy
Poissy
St-Germain en-Laye
La Défense Grande Arche
Pont de Levallois Bécon
Porte de Clignancourt
Porte de la Chapelle
La Courneuve 8 Mai 1945
Bobigny Pablo Picasso
Chelles Gournay
Noisy-le-Sec
Tournan
Mairie des Lilas
Porte des Lilas
Gare du Nord
Gare de l'Est
Gare Saint-Lazare
Haussmann Saint-Lazare
Charles de Gaulle Étoile
Porte Dauphine
Gallieni
Mairie de Montreuil
Marne-la-Vallée Parcs Disneyland
Boissy-Saint-Léger
Château de Vincennes
Nation
Gare de Lyon
Gare d'Austerlitz
Châtelet
Châtelet Les Halles
Gare Montparnasse
Boulogne Pont de St-Cloud
Pont de Sèvres
Balard
Mairie d'Issy
Issy Val de Seine
Versailles – Rive Gauche Château de Versailles
Saint-Quentin-en-Yvelines
Châtillon – Montrouge
Porte d'Orléans
Robinson
Saint-Rémy lès-Chevreuse
Orly Ouest
Orly Sud
Massy – Palaiseau
Versailles – Chantiers
Dourdan
Saint-Martin-d'Étampes
Villejuif – Louis Aragon
Mairie d'Ivry
Bibliothèque François Mitterrand
Place d'Italie
Créteil – Préfecture
Malesherbes
Melun

#neo4j

Frequently Bought Together


+

+


Price For All Three: £69.50

Add all three to Basket

Show availability and delivery details

- ☑ **This item:** Patterns of Enterprise Application Architecture (The Addison-Wesley Signature Series) by Martin Fowler Hardcover £24.00
- ☑ Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions (Addison-Wesley Signature Series (Fowler) Addison-Wesley Sign) by Gregor Hohpe Hardcover £22.00
- ☑ Domain-driven Design: Tackling Complexity in the Heart of Software by Eric Evans Hardcover £23.50

Customers Who Bought This Item Also Bought

Enterprise Integration Patterns: Designing, …

Gregor Hohpe

(16)

Hardcover

£22.00

Service Design Patterns: Fundamental Design …

Robert Daigneau

(2)

Hardcover

£19.97

Design patterns : elements of reusable …

Erich Gamma

(51)

Hardcover

£21.00

Domain-driven Design: Tackling Complexity in …

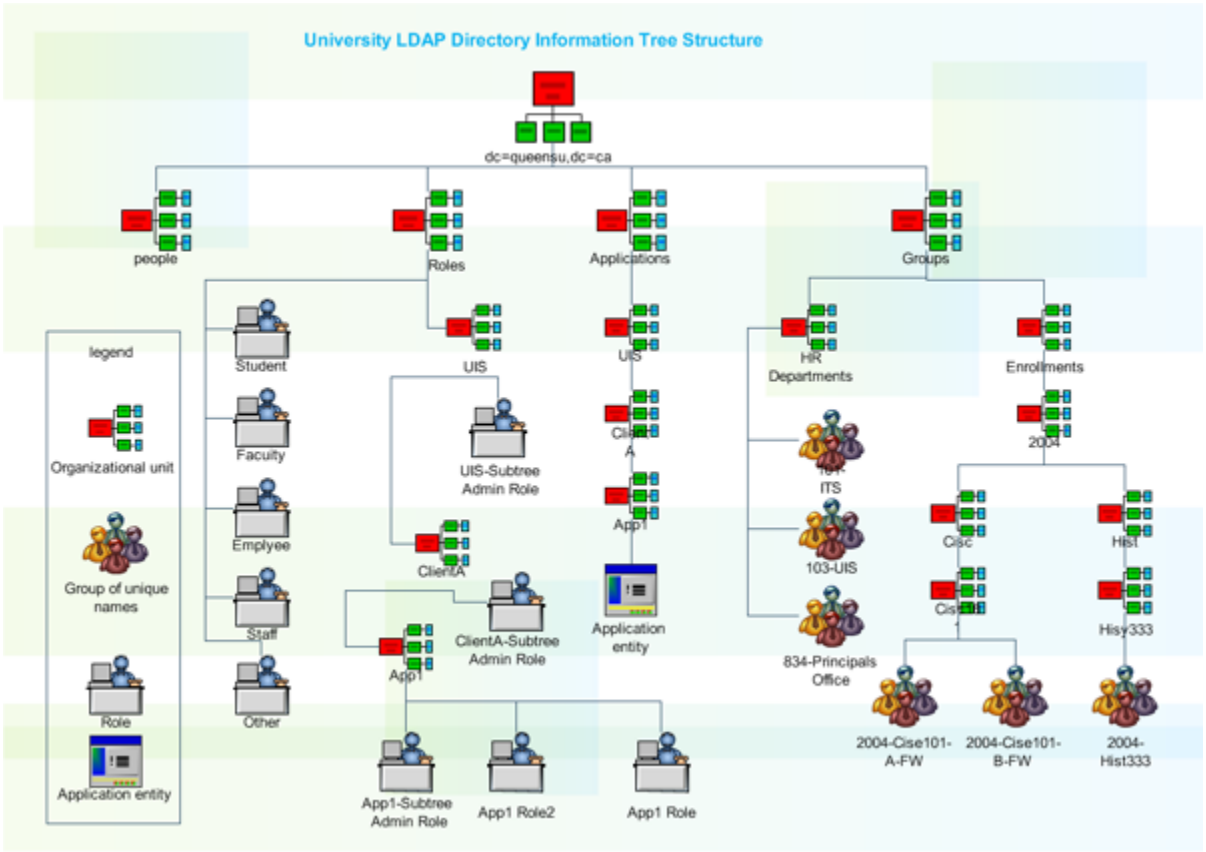
Eric Evans

(13)

Hardcover

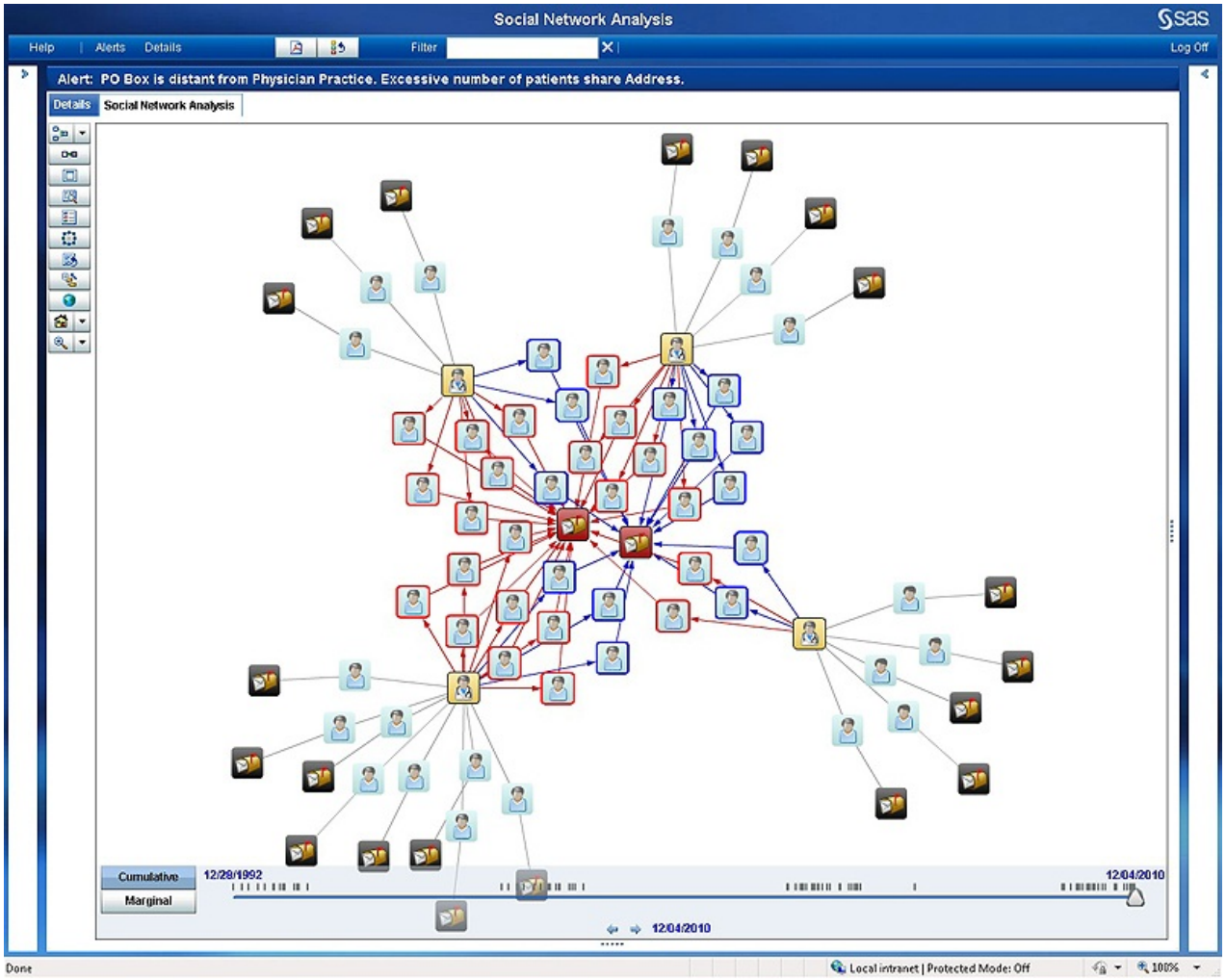
£23.50

#neo4j

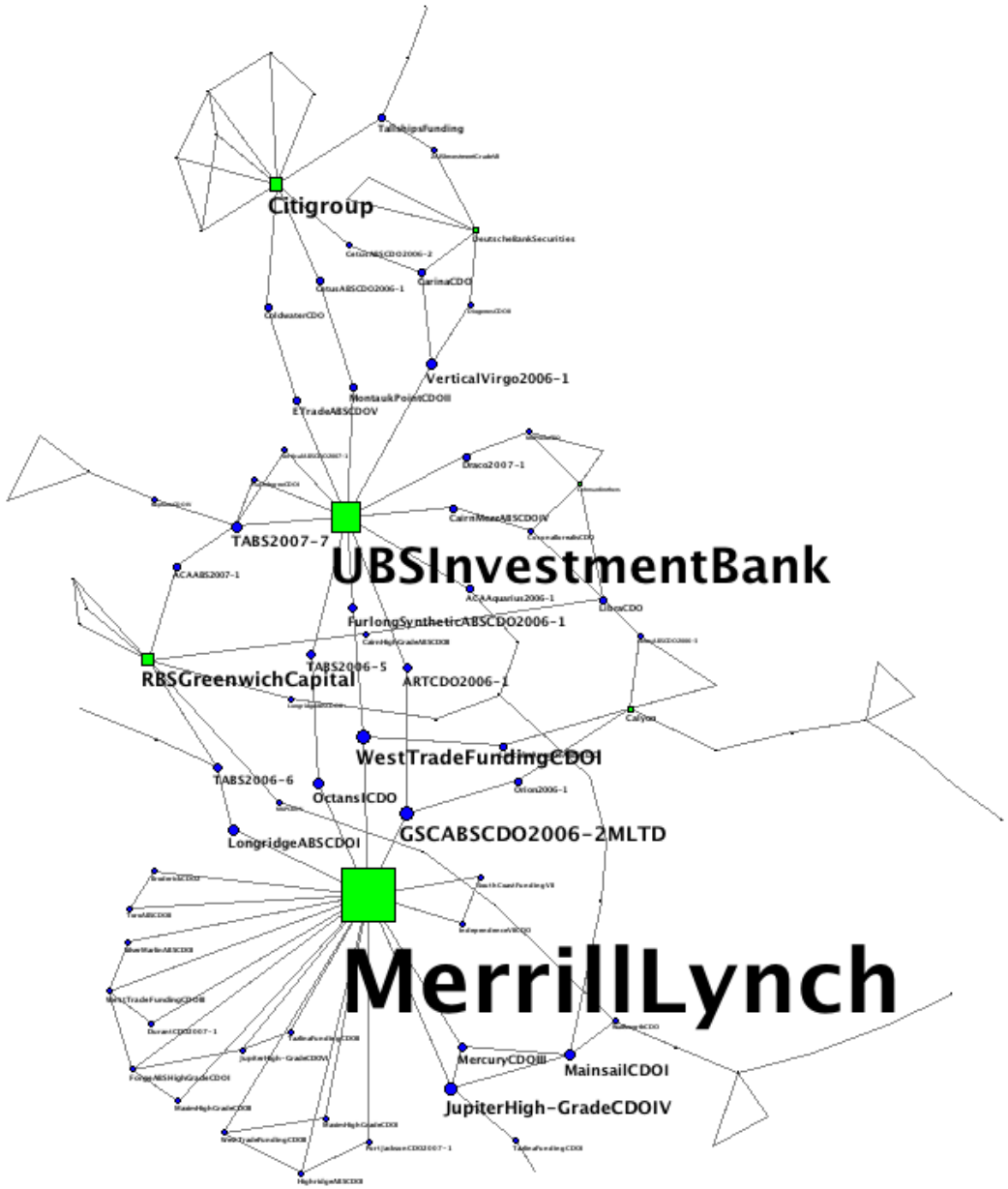

University LDAP Directory Information Tree Structure
dc=queensu,dc=ca
people
Roles
Applications
Groups
legend
Organizational unit
Group of unique names
Role
Application entity
Student
Faculty
Emplyee
Staff
Other
UIS
UIS-Subtree Admin Role
ClientA
ClientA-Subtree Admin Role
App1
App1-Subtree Admin Role
App1 Role2
App1 Role
UIS
App1
Application entity
HR Departments
ITS
103-UIS
834-Principals Office
Enrollments
2004
Cisc
Hist
Hisy333
2004-Cise101-A-FW
2004-Cise101-B-FW
2004-Hist333

#neo4j

#neo4j

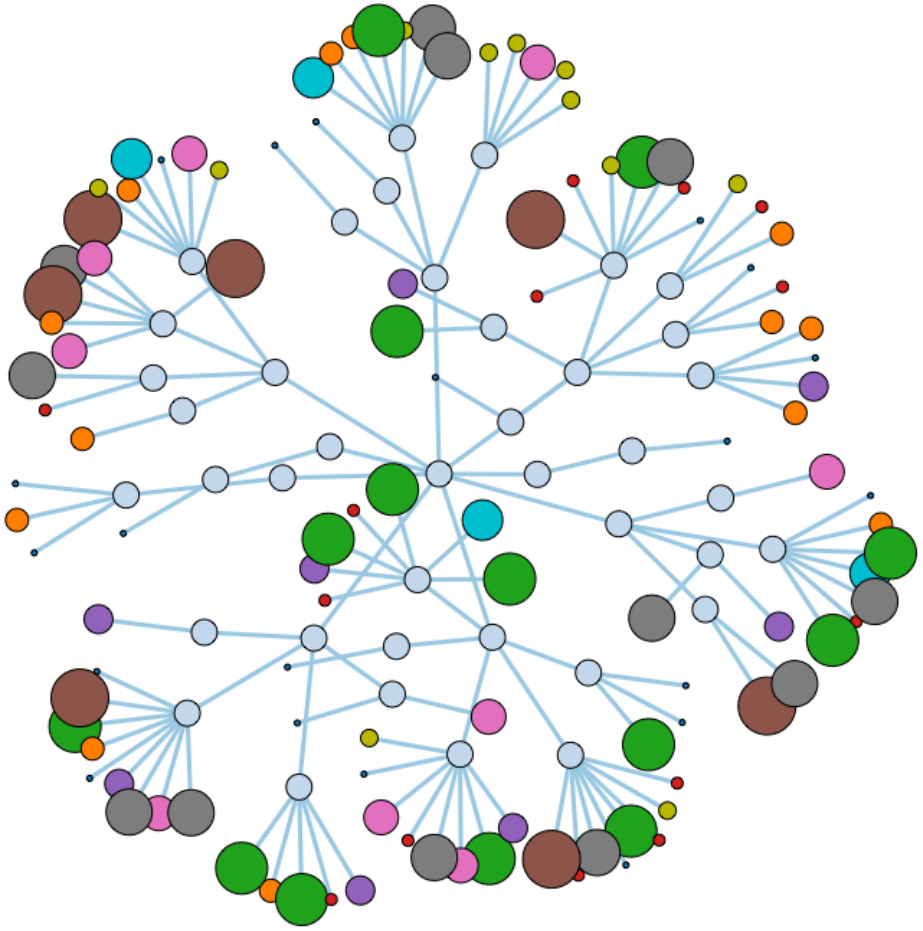

Social Network Analysis
sas
Help
Alerts
Details
Filter
Log Off
Alert: PO Box is distant from Physician Practice. Excessive number of patients share Address.
Details
Social Network Analysis
Cumulative
Marginal
12/28/1992
12/04/2010
12/04/2010
Done
Local intranet | Protected Mode: Off
100%

#neo4j

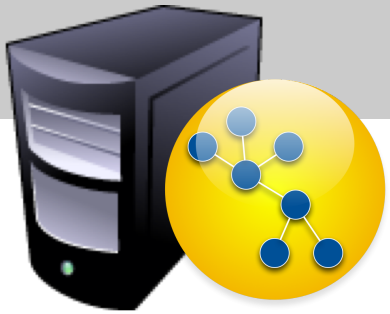

Citigroup
VerticalVirgo2006-1
TABS2007-7
UBSInvestmentBank
FurlongSyntheticABSCDO2006-1
RBSGreenwichCapital
TABS2006-5
ARTCDO2006-1
WestTradeFundingCDOI
TABS2006-6
OctansICDO
GSCABSCDO2006-2MLTD
LongridgeABSCDOI
MerrillLynch
MercuryCDOIII
MainsailCDOI
JupiterHigh-GradeCDOIV

#neo4j

Graphs Are Everywhere



Neo4j is a Graph Database

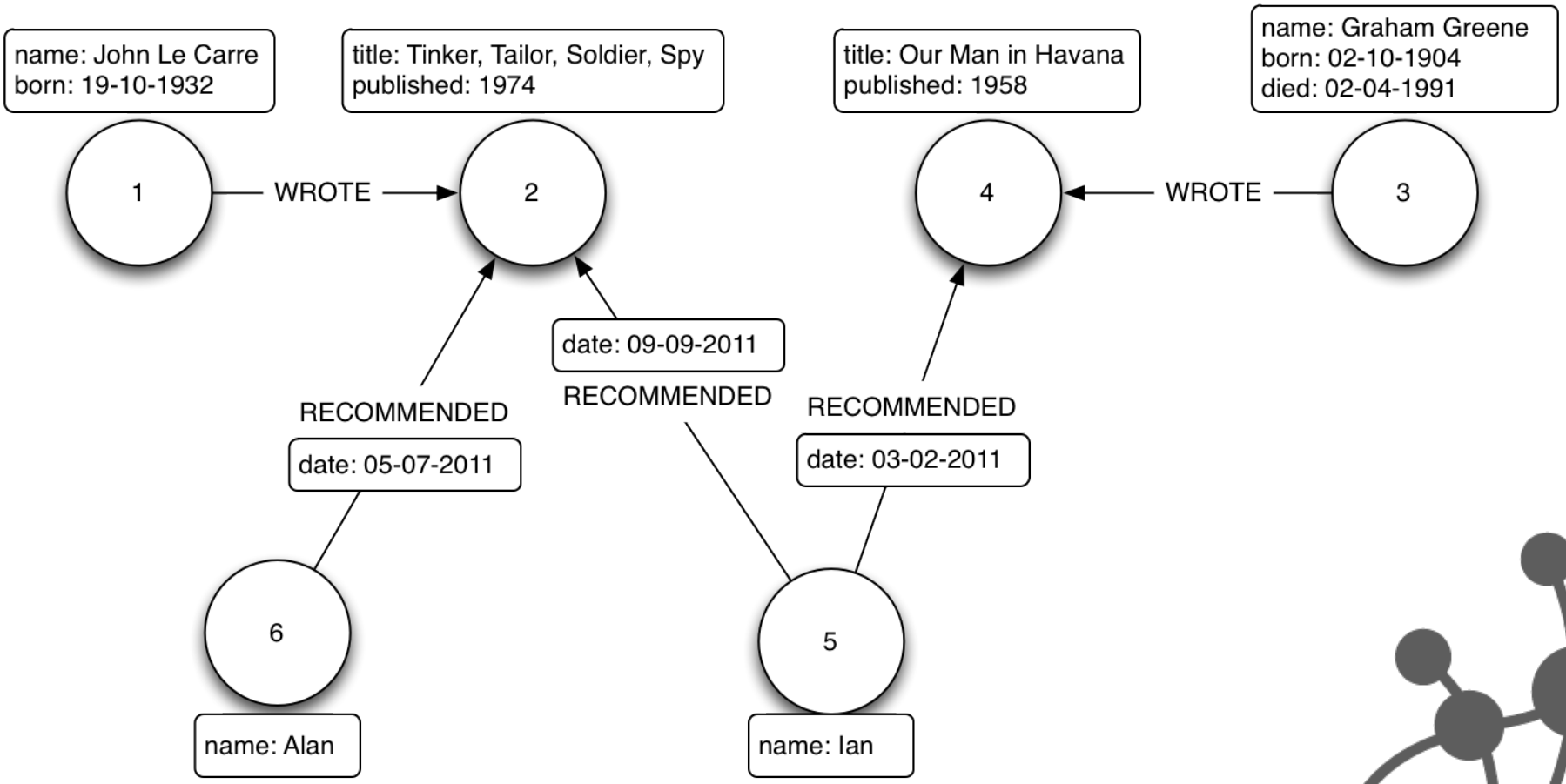


- Store
- Manage
- Query

data

as a graph

Property Graph Data Model



Graph Database Benefits

Densely-connected, semi-structured domains

- Lots of join tables? Connectedness
- Lots of sparse tables? Semi-structure

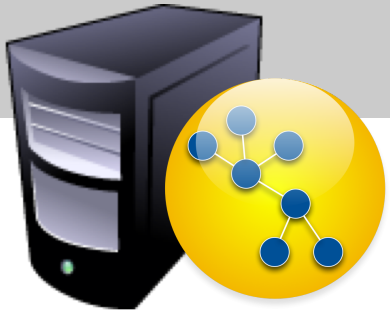
Data Model Volatility

- Easy to evolve

Join Complexity and Performance

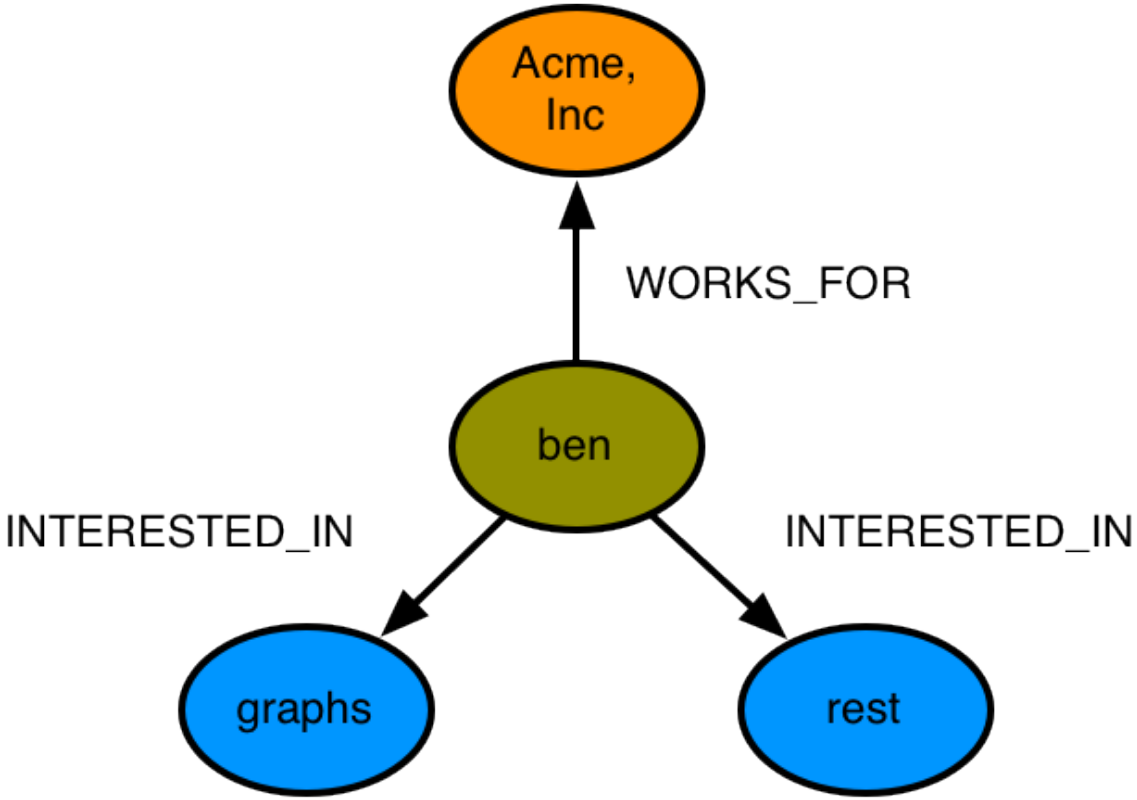
- Millions of 'joins' per second
- Consistent query times as dataset grows

Querying Graph Data

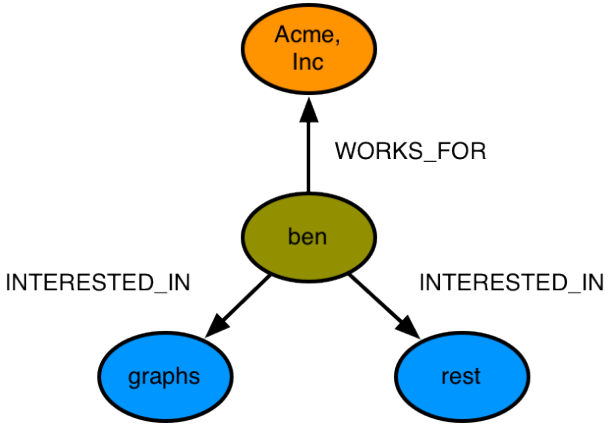


- A language for describing graphs
- Creating nodes, relationships and properties
- Querying data

Describing Graphs

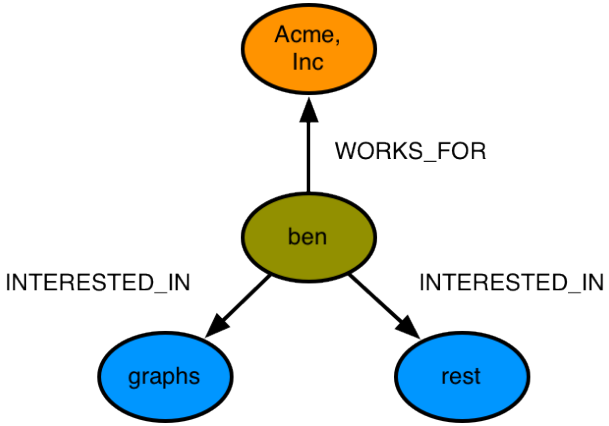


Cypher




```
(graphs)<-[:INTERESTED_IN]-(ben)-[:INTERESTED_IN]->(rest),
(ben)-[:WORKS_FOR]->(acme)
```

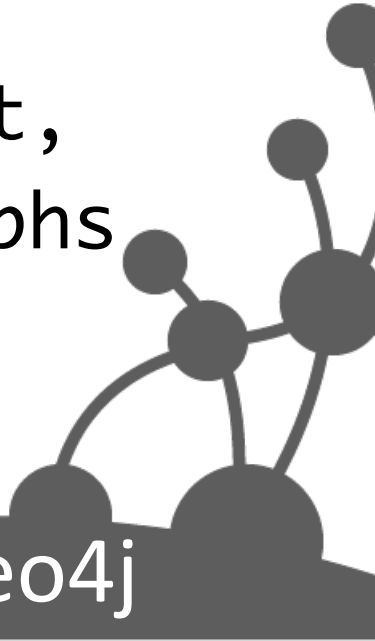
Cypher




```
(ben)-[:INTERESTED_IN]->(graphs),
(ben)-[:INTERESTED_IN]->(rest),
(ben)-[:WORKS_FOR]->(acme)
```

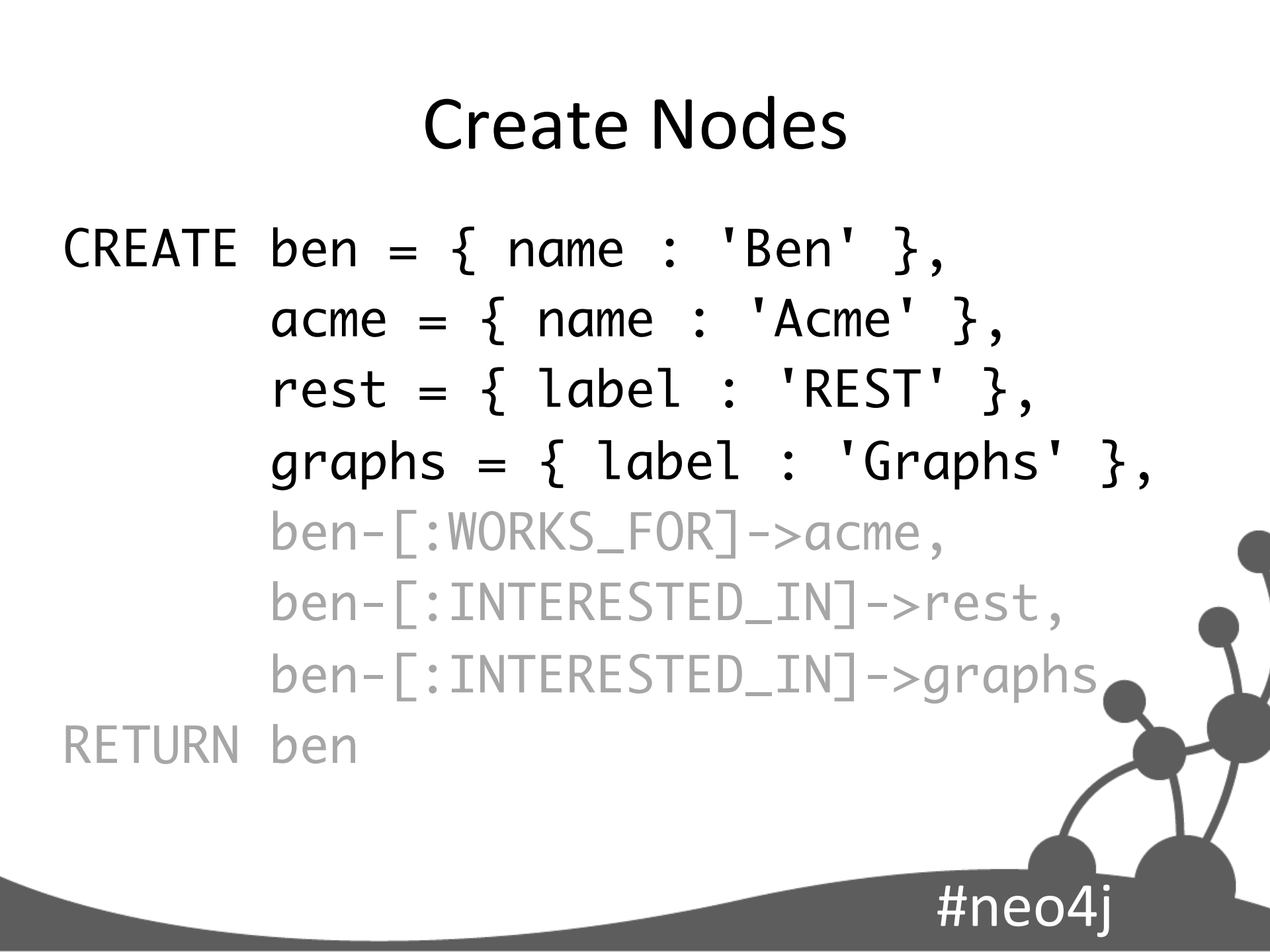
Create Some Data

```
CREATE ben = { name : 'Ben' },
       acme = { name : 'Acme' },
       rest = { label : 'REST' },
       graphs = { label : 'Graphs' },
       ben-[:WORKS_FOR]->acme,
       ben-[:INTERESTED_IN]->rest,
       ben-[:INTERESTED_IN]->graphs
RETURN ben
```



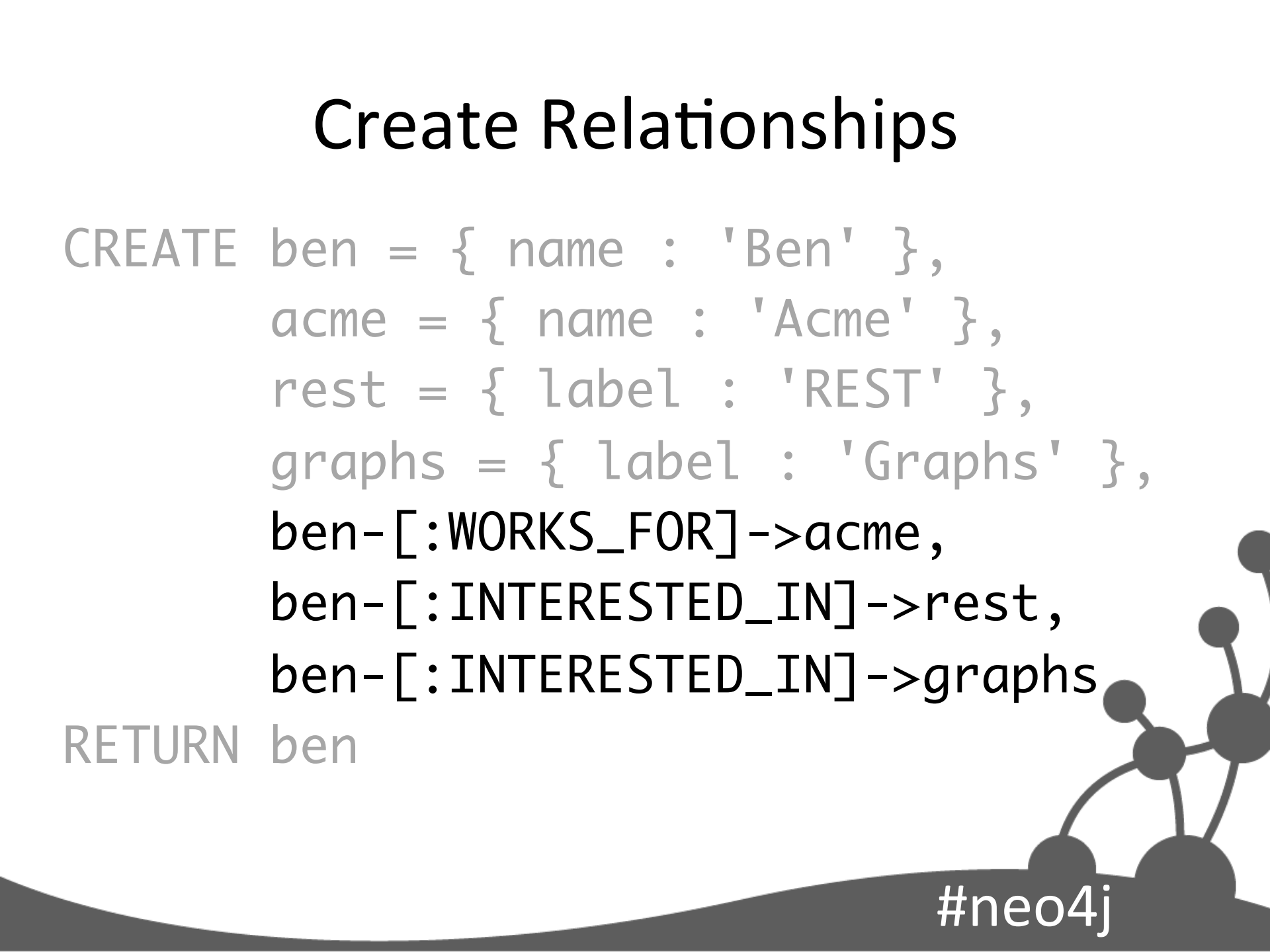
Create Nodes

```
CREATE ben = { name : 'Ben' },
       acme = { name : 'Acme' },
       rest = { label : 'REST' },
       graphs = { label : 'Graphs' },
       ben-[:WORKS_FOR]->acme,
       ben-[:INTERESTED_IN]->rest,
       ben-[:INTERESTED_IN]->graphs
RETURN ben
```



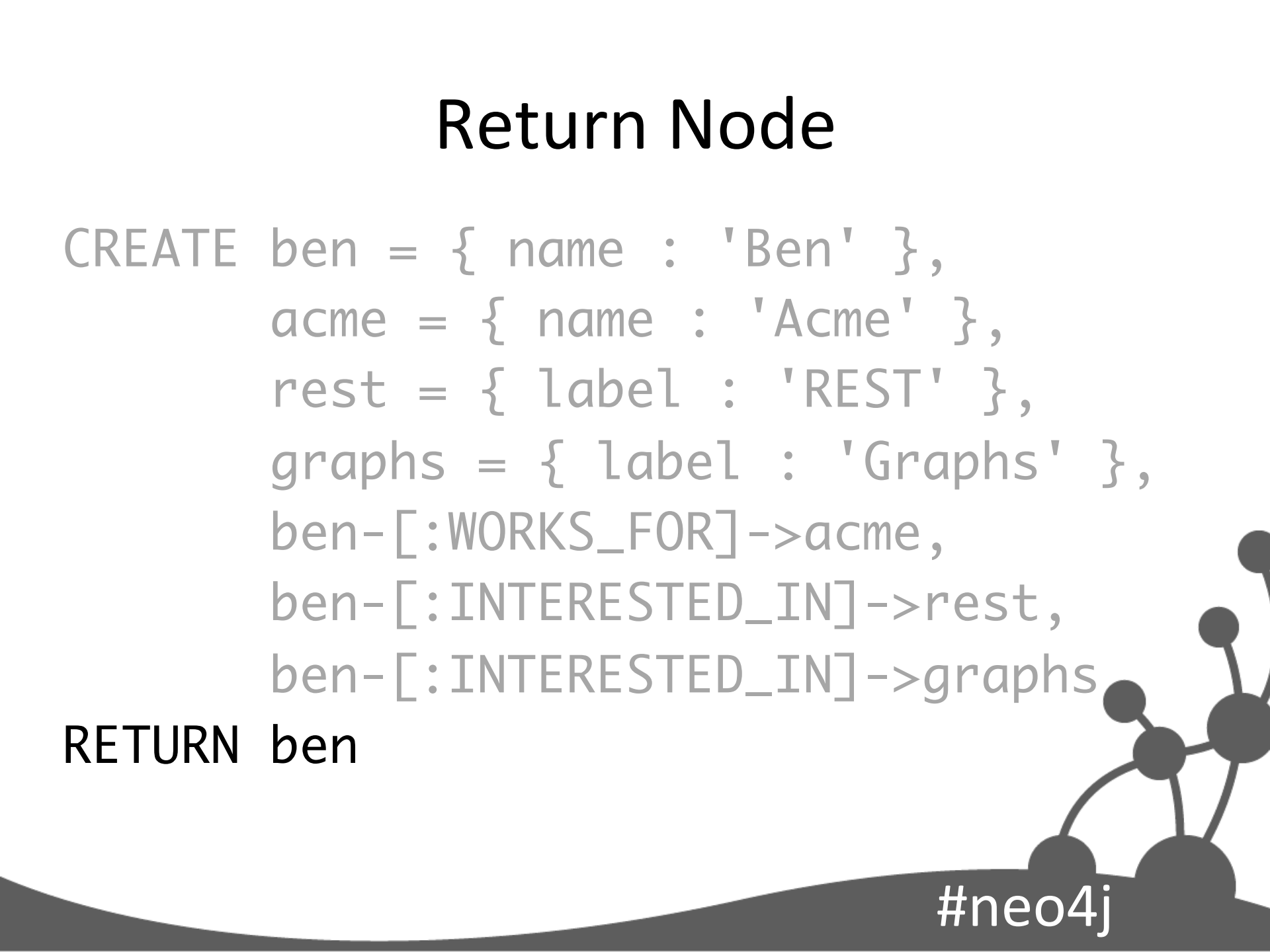
Create Relationships

```
CREATE ben = { name : 'Ben' },
       acme = { name : 'Acme' },
       rest = { label : 'REST' },
       graphs = { label : 'Graphs' },
       ben-[:WORKS_FOR]->acme,
       ben-[:INTERESTED_IN]->rest,
       ben-[:INTERESTED_IN]->graphs
RETURN ben
```

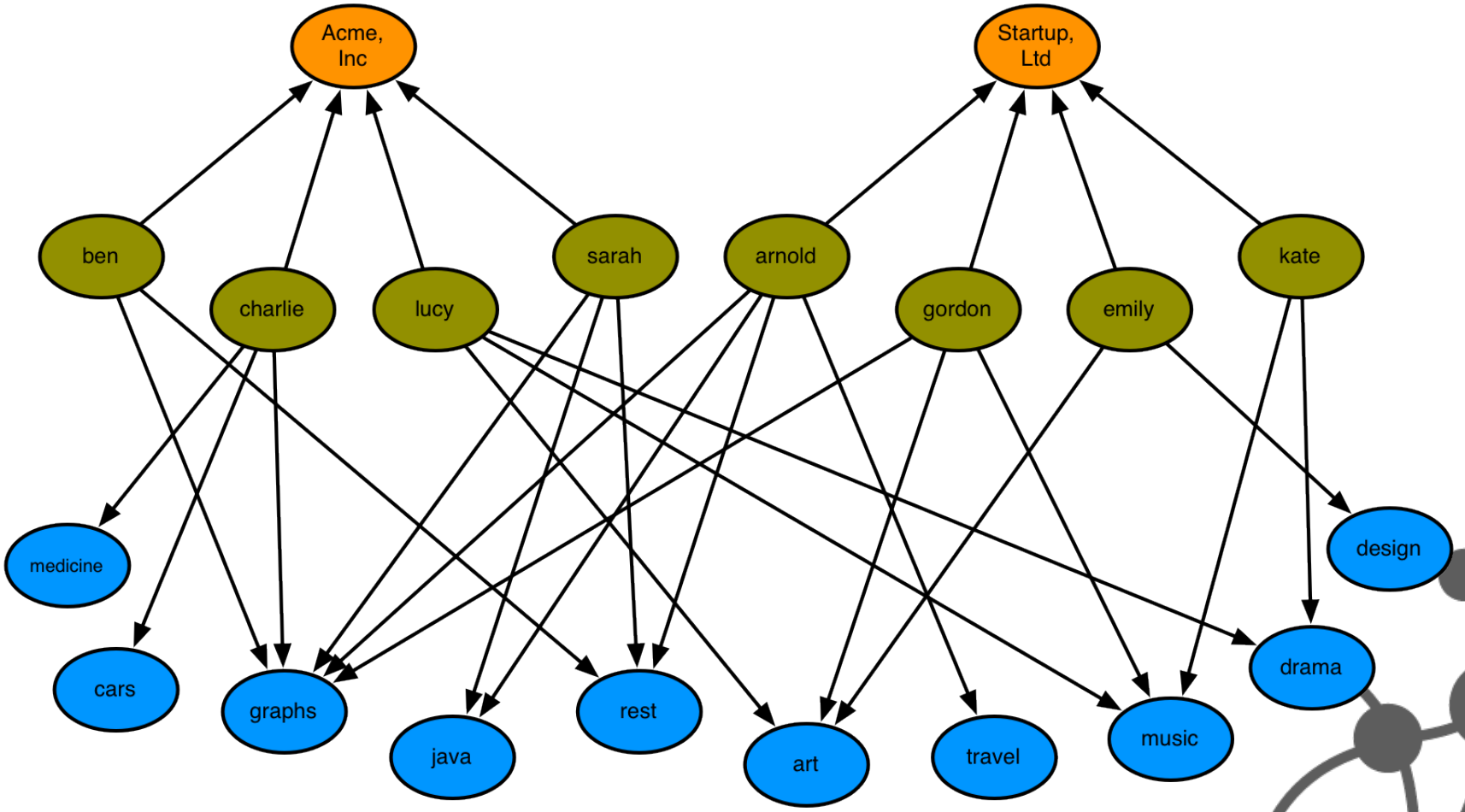


Return Node

```
CREATE ben = { name : 'Ben' },
       acme = { name : 'Acme' },
       rest = { label : 'REST' },
       graphs = { label : 'Graphs' },
       ben-[:WORKS_FOR]->acme,
       ben-[:INTERESTED_IN]->rest,
       ben-[:INTERESTED_IN]->graphs
RETURN ben
```



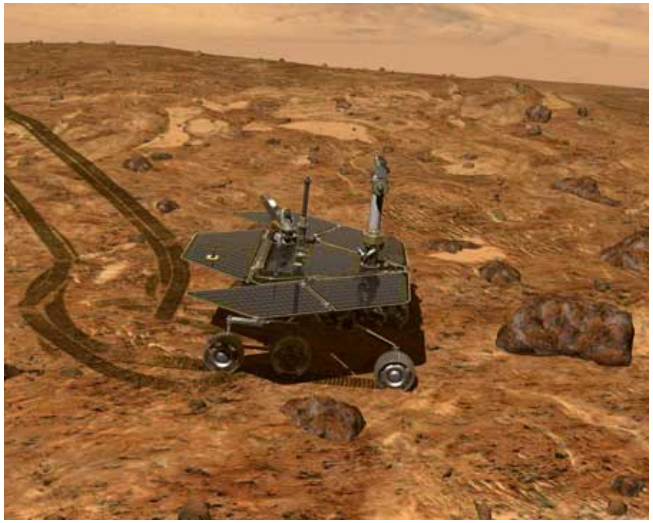
Eventually…

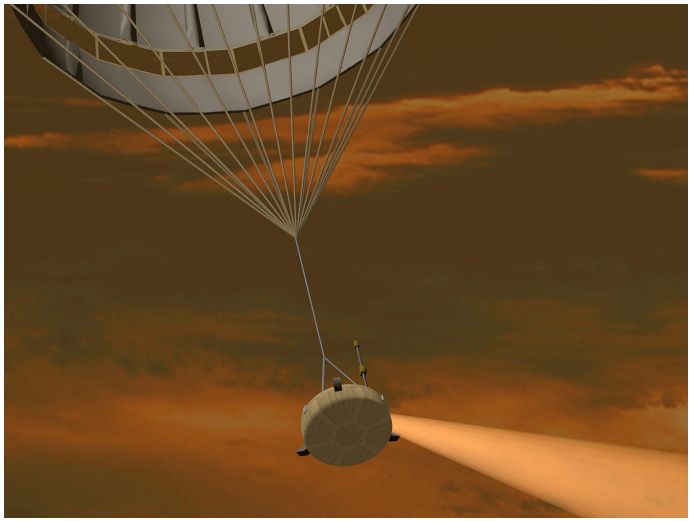

Acme, Inc
Startup, Ltd
ben
charlie
lucy
sarah
arnold
gordon
emily
kate
medicine
cars
graphs
java
rest
art
travel
music
drama
design

Querying a Graph

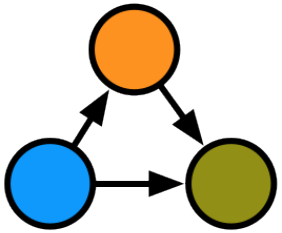
Graph local:

- One or more start nodes
- Explore surrounding graph
- Millions of joins per second

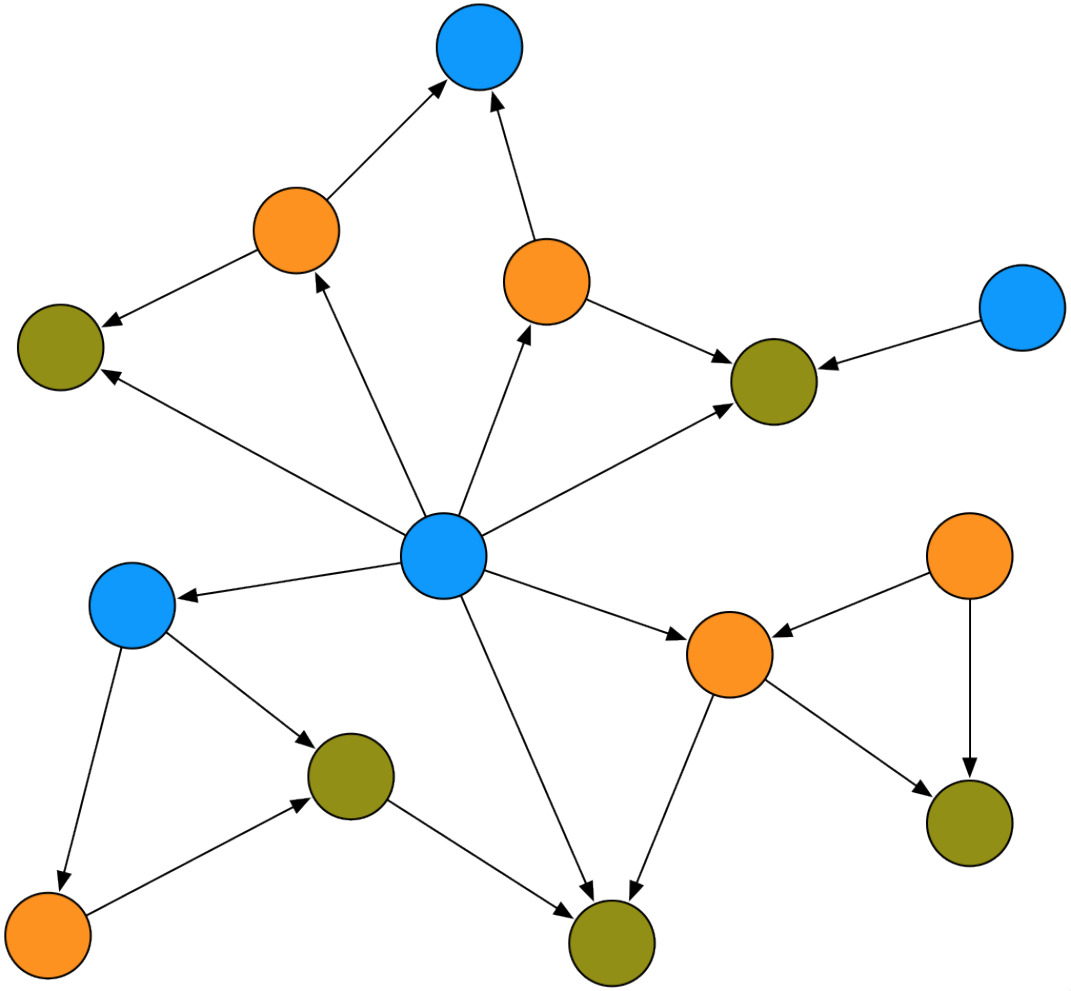




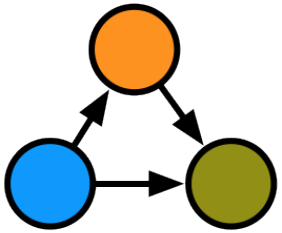
Queries: Pattern Matching



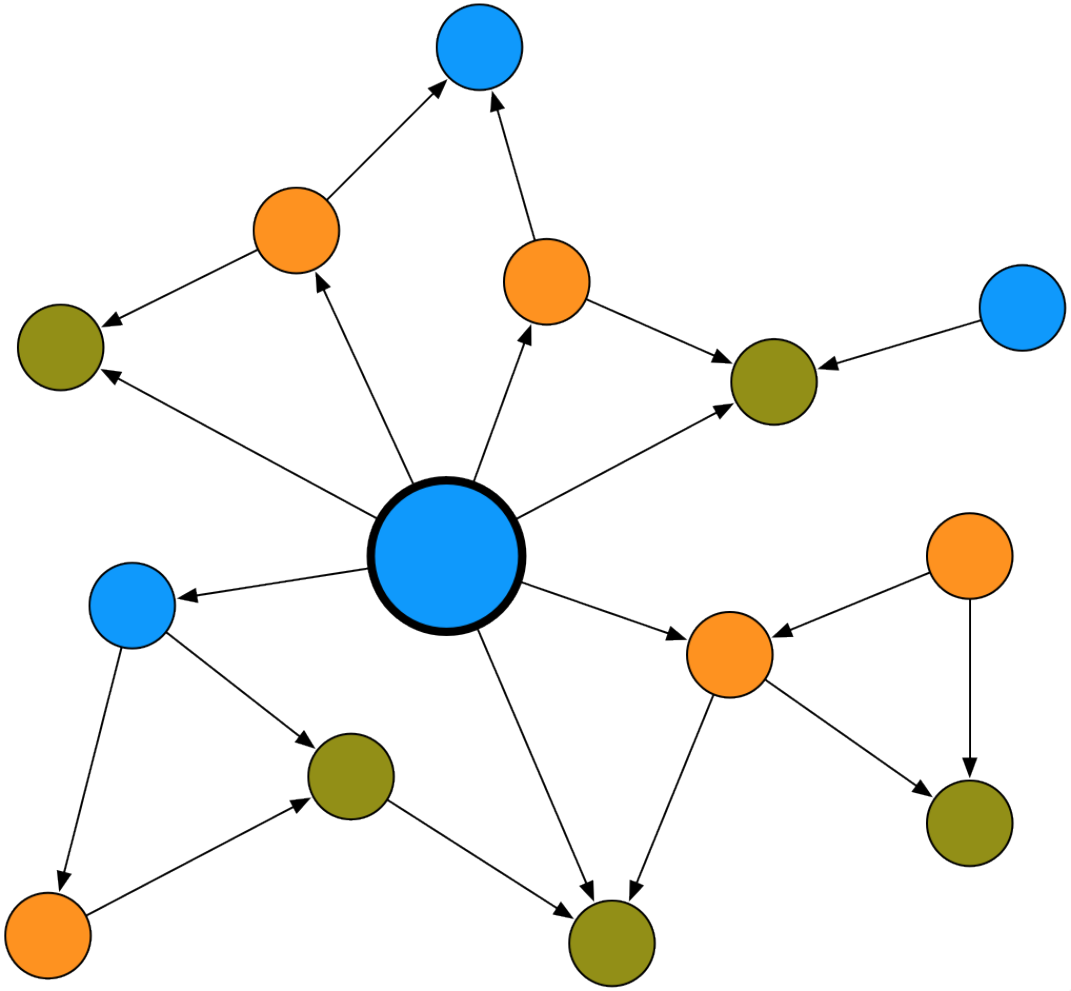
Pattern



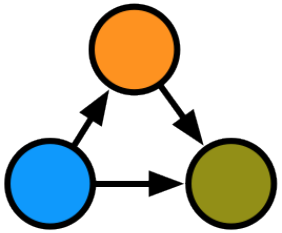
Start Node



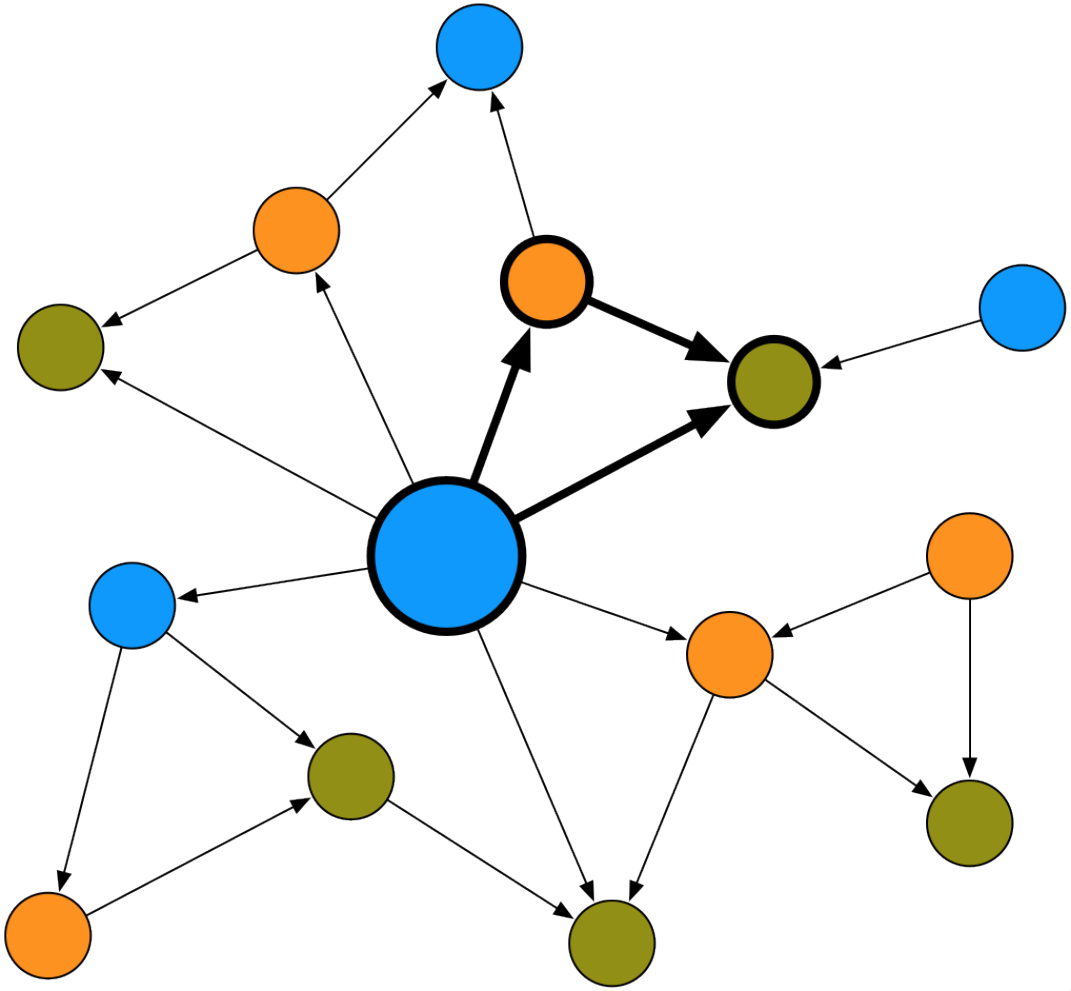
Pattern



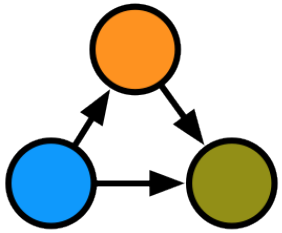
Match



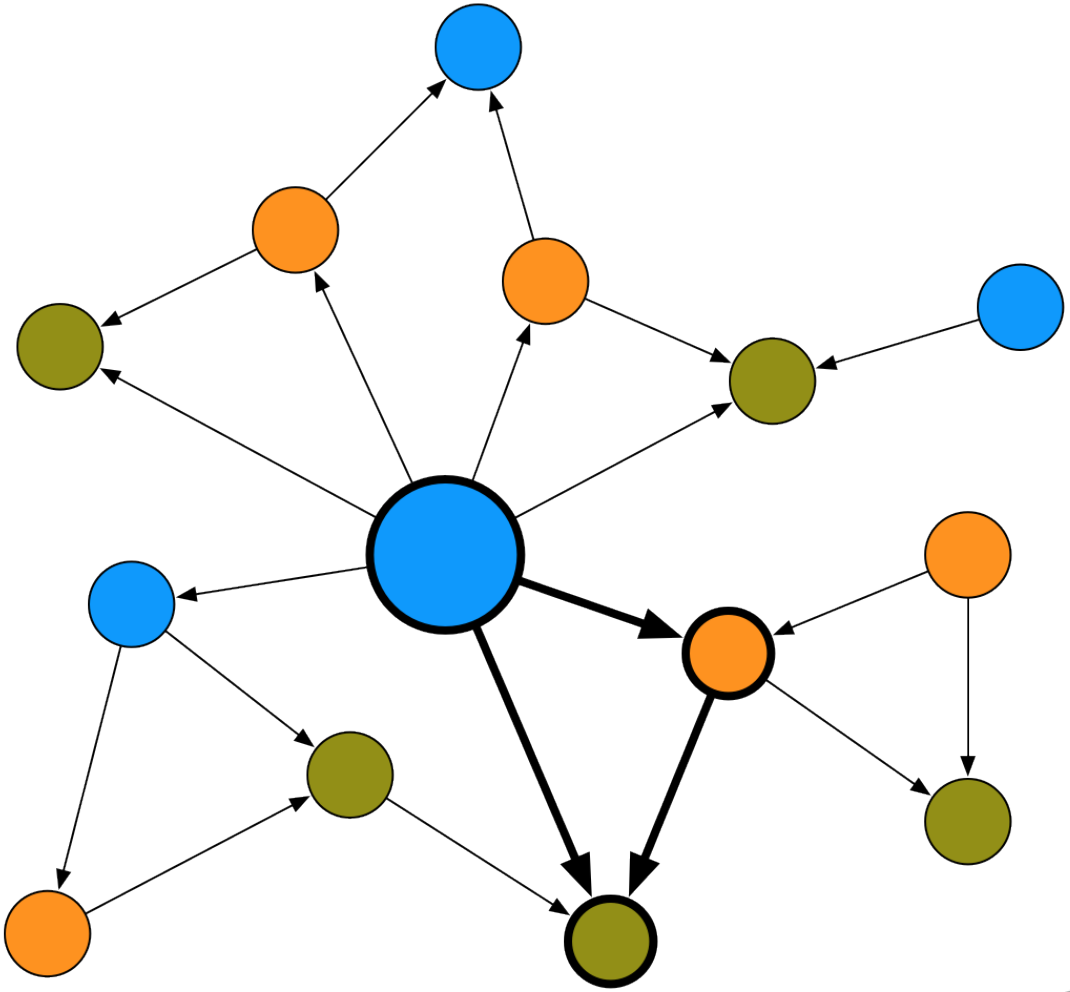
Pattern



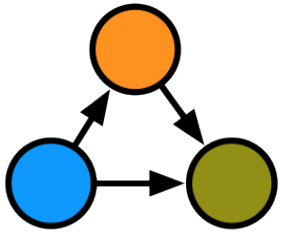
Match



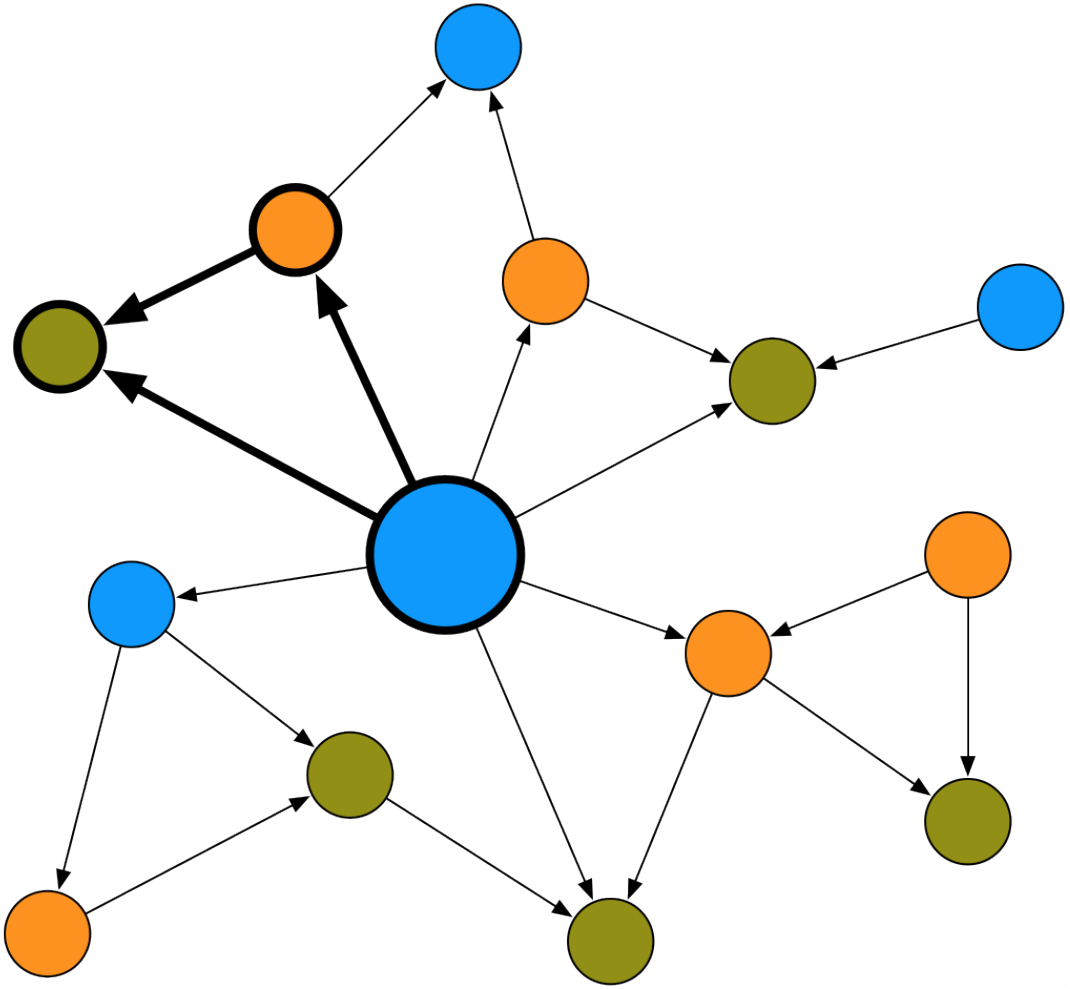
Pattern



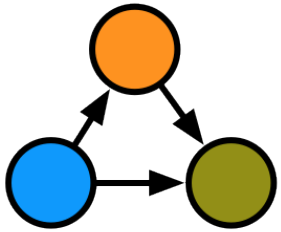
Match



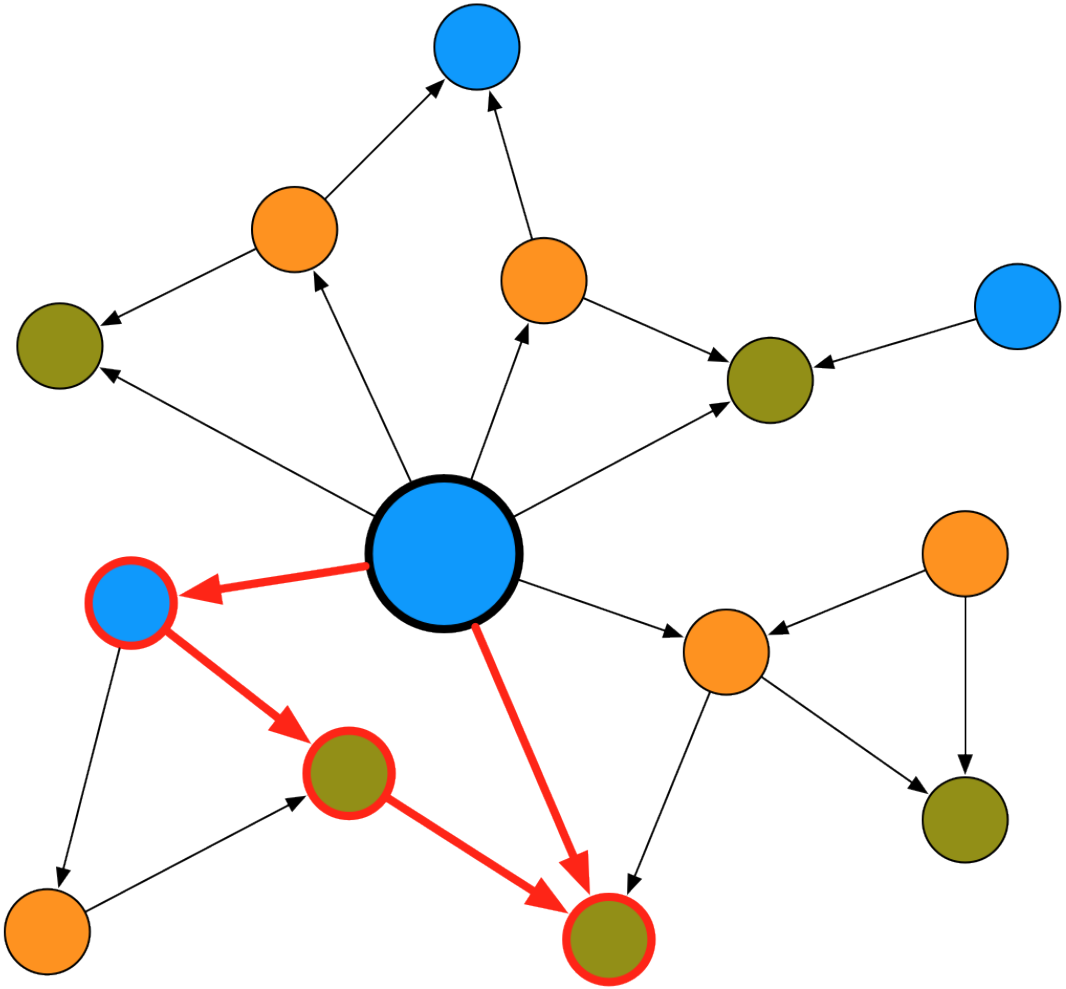
Pattern



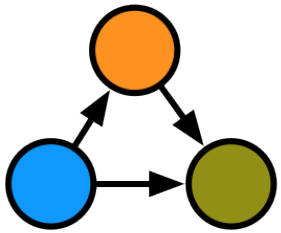
Non-Match



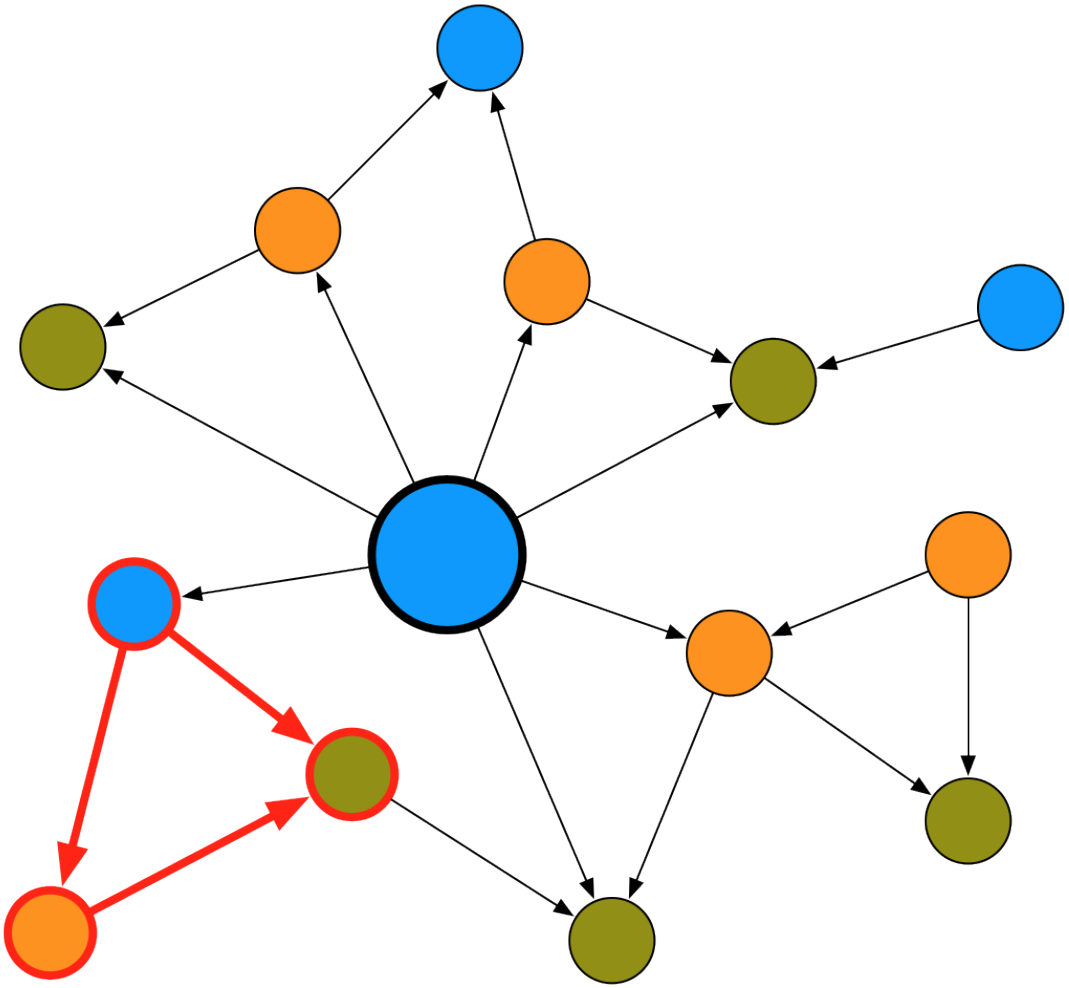
Pattern



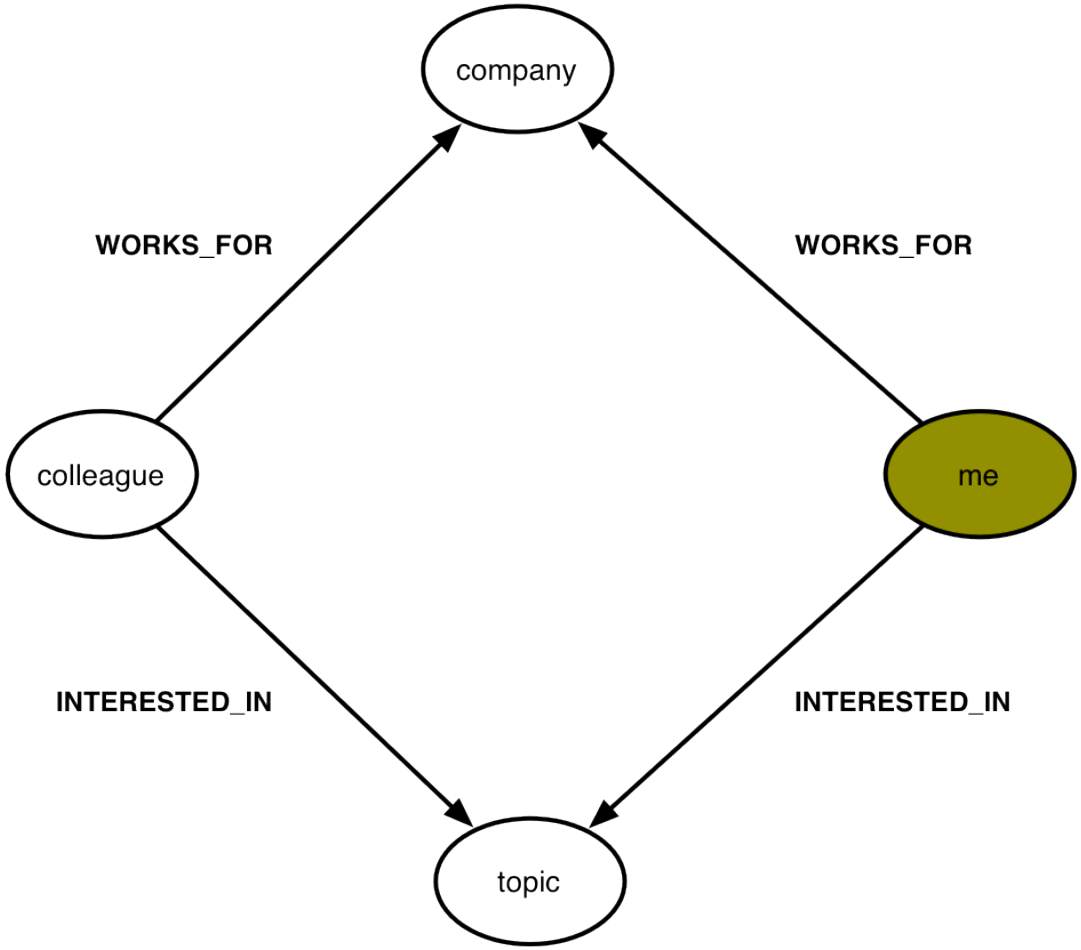
Non-Match



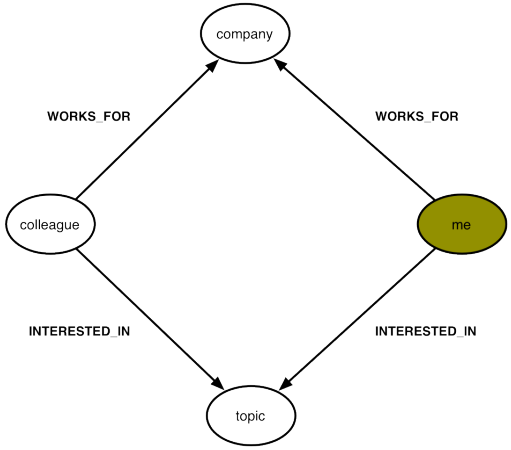
Pattern



Colleagues Who Share My Interests



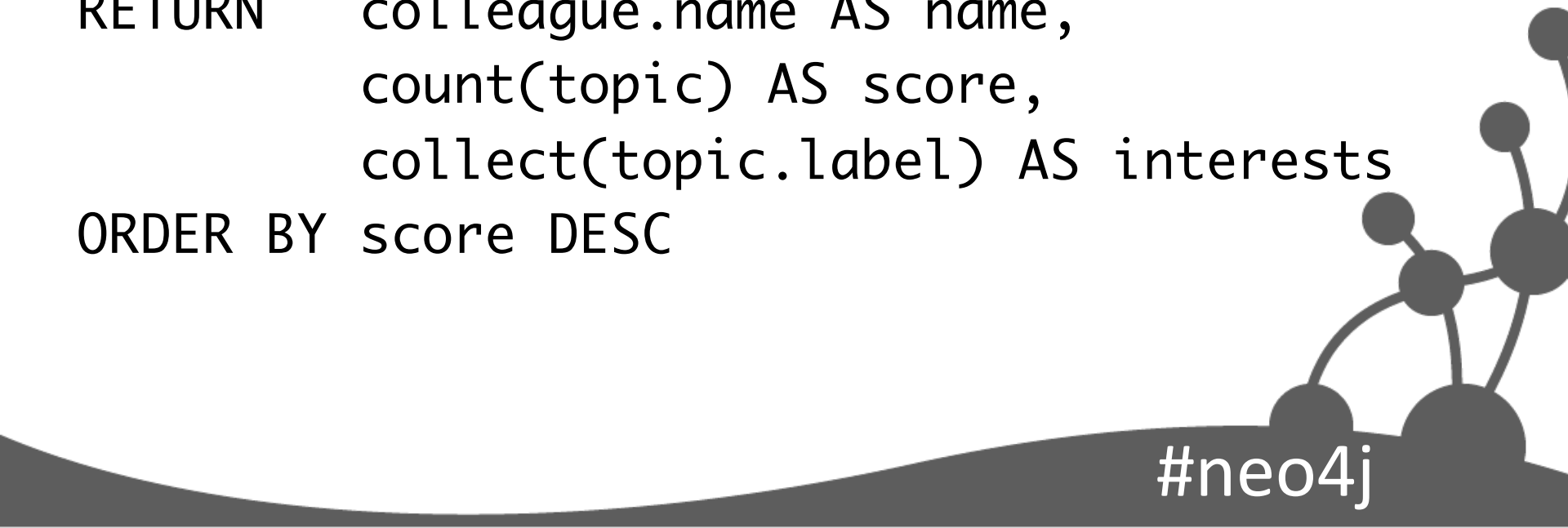
Cypher Pattern




```
me-[:INTERESTED_IN]->topic,
me-[:WORKS_FOR]->company,
colleague-[:INTERESTED_IN]->topic,
colleague-[:WORKS_FOR]->company
```

Find Colleagues

```
START    me=node:users(name = "sarah")
MATCH    me-[:INTERESTED_IN]->topic,
         me-[:WORKS_FOR]->company,
         colleague-[:INTERESTED_IN]->topic,
         colleague-[:WORKS_FOR]->company
RETURN   colleague.name AS name,
         count(topic) AS score,
         collect(topic.label) AS interests
ORDER BY score DESC
```



Find Start Node

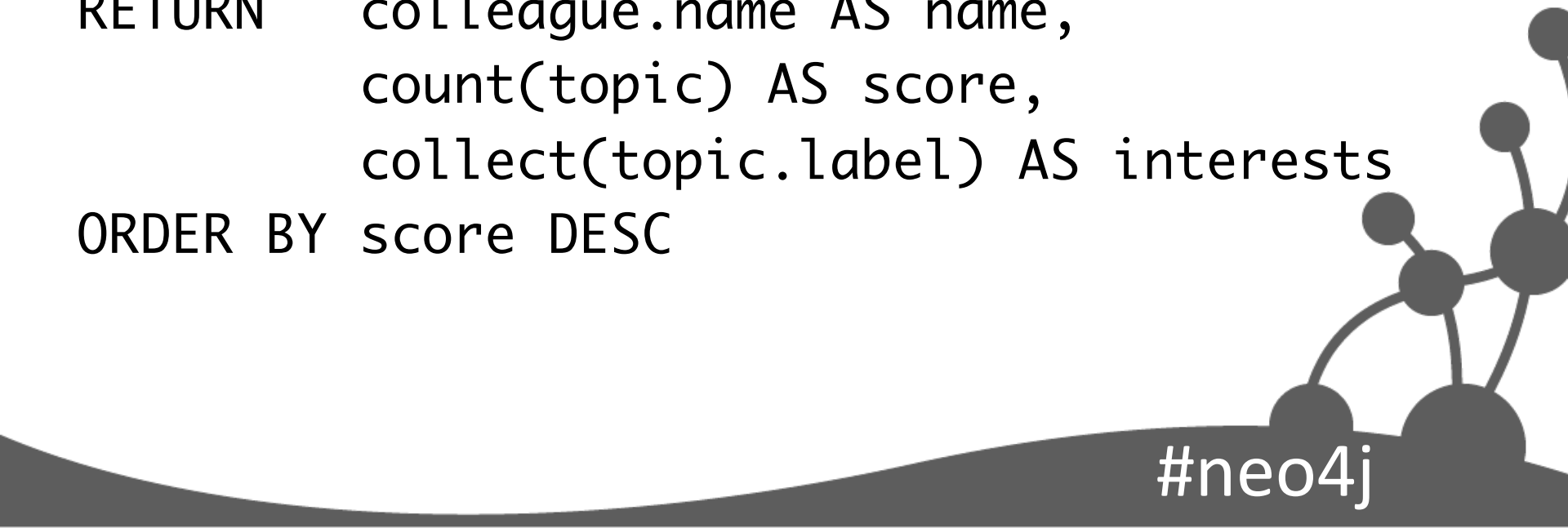
```
START    me=node:users(name = "sarah")
MATCH    me-[:INTERESTED_IN]->topic,
         me-[:WORKS_FOR]->company,
         colleague-[:INTERESTED_IN]->topic,
         colleague-[:WORKS_FOR]->company
RETURN   colleague.name AS name,
         count(topic) AS score,
         collect(topic.label) AS interests
ORDER BY score DESC
```

Describe Pattern

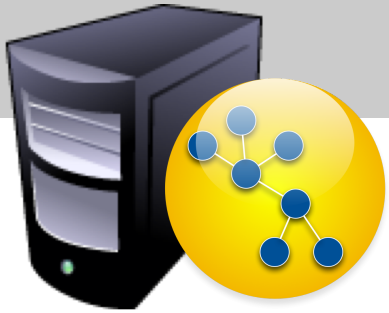
```
START    me=node:users(name = "sarah")
MATCH    me-[:INTERESTED_IN]->topic,
         me-[:WORKS_FOR]->company,
         colleague-[:INTERESTED_IN]->topic,
         colleague-[:WORKS_FOR]->company
RETURN   colleague.name AS name,
         count(topic) AS score,
         collect(topic.label) AS interests
ORDER BY score DESC
```

Construct Results

```
START    me=node:users(name = "sarah")
MATCH    me-[:INTERESTED_IN]->topic,
         me-[:WORKS_FOR]->company,
         colleague-[:INTERESTED_IN]->topic,
         colleague-[:WORKS_FOR]->company
RETURN   colleague.name AS name,
         count(topic) AS score,
         collect(topic.label) AS interests
ORDER BY score DESC
```

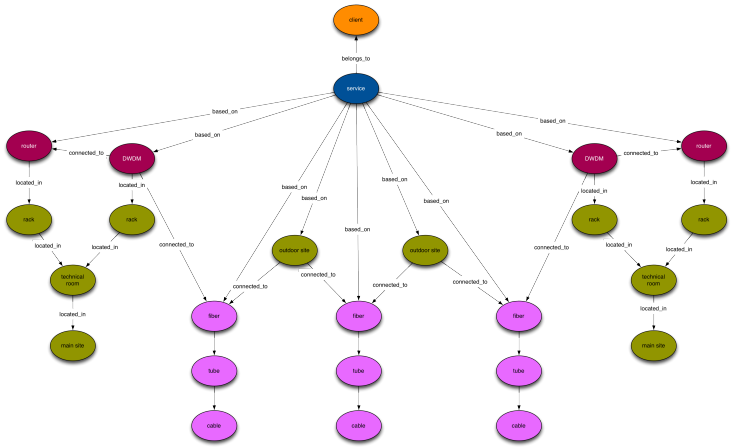


Case Studies



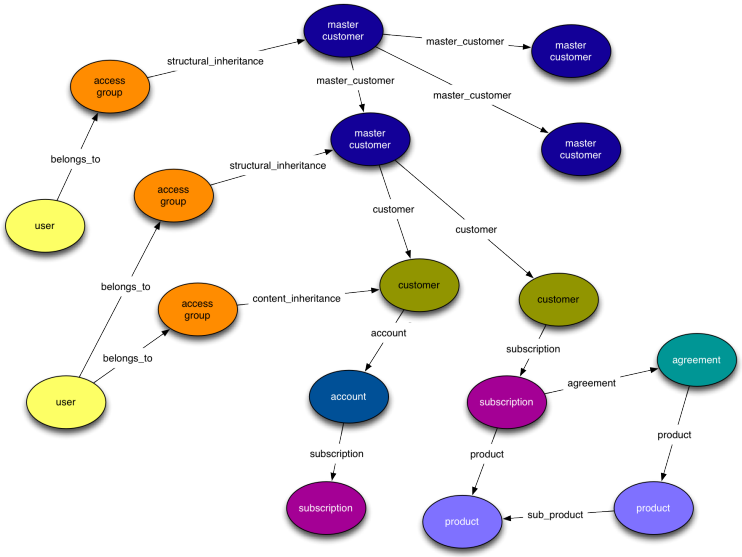
Network Impact Analysis

- Which parts of network does a customer depend on?
- Who will be affected if we replace a network element?



Asset Management & Access Control

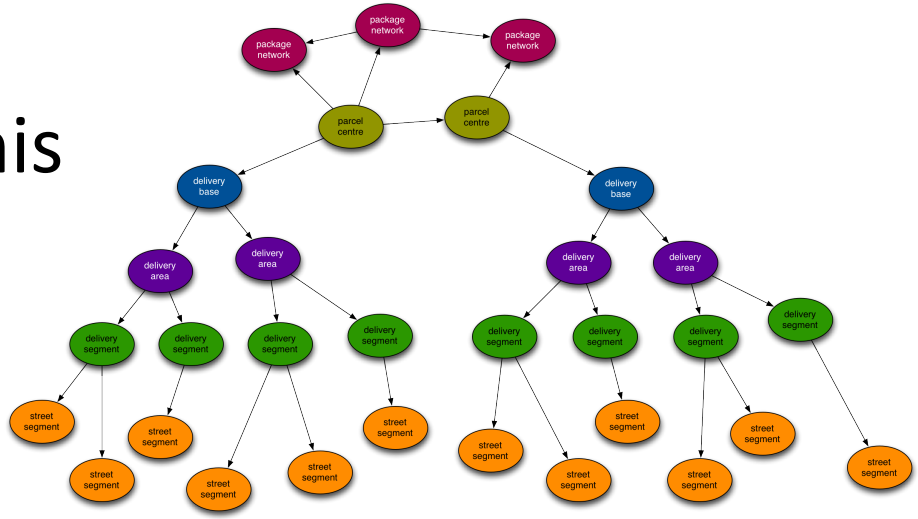
- Which assets can an admin control?
- Who can change my subscription?



#neo4j

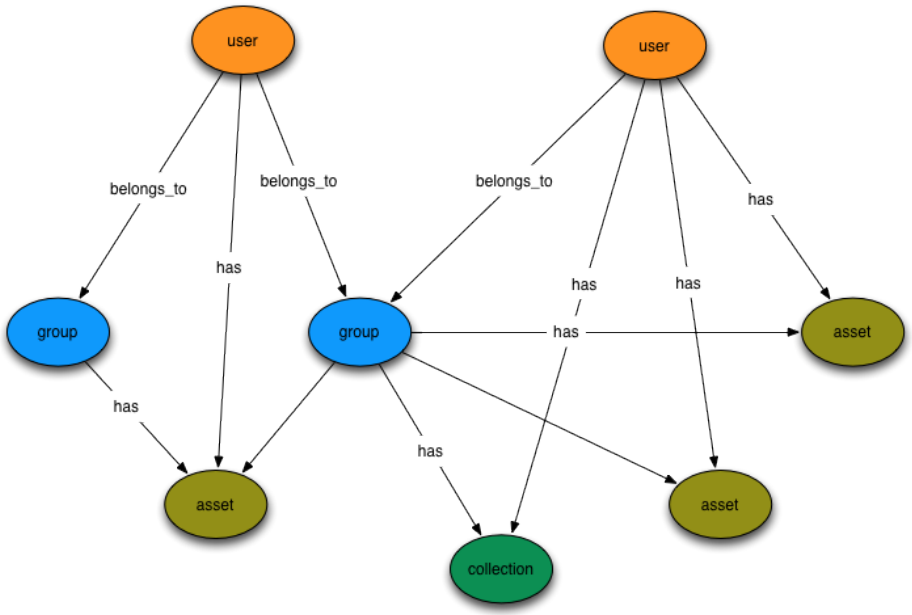
Logistics

- What's the quickest delivery route for this parcel?



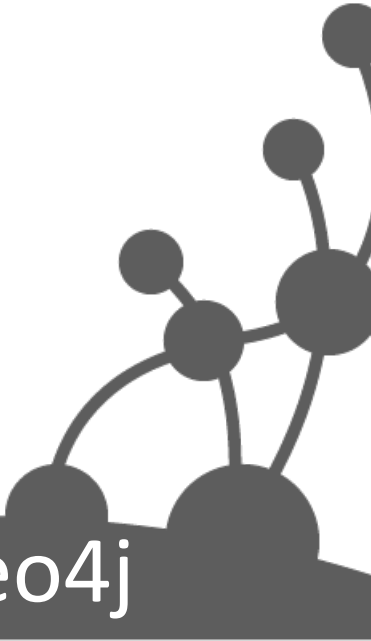
Social Network & Recommendations

- Which assets can I access?
- Who shares my interests?



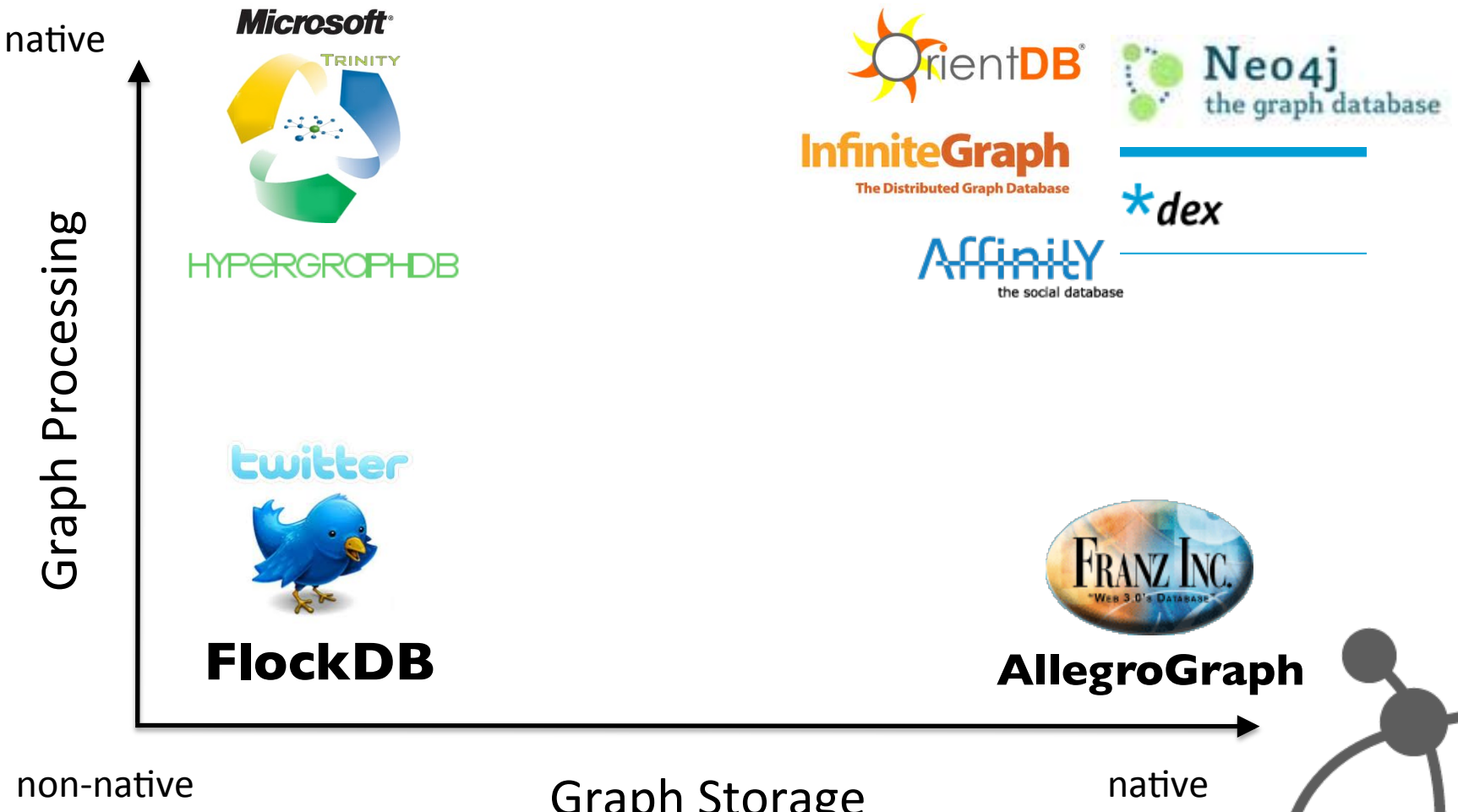
Thank You

@iansrobinson
ian.robinson@neotechnology.com



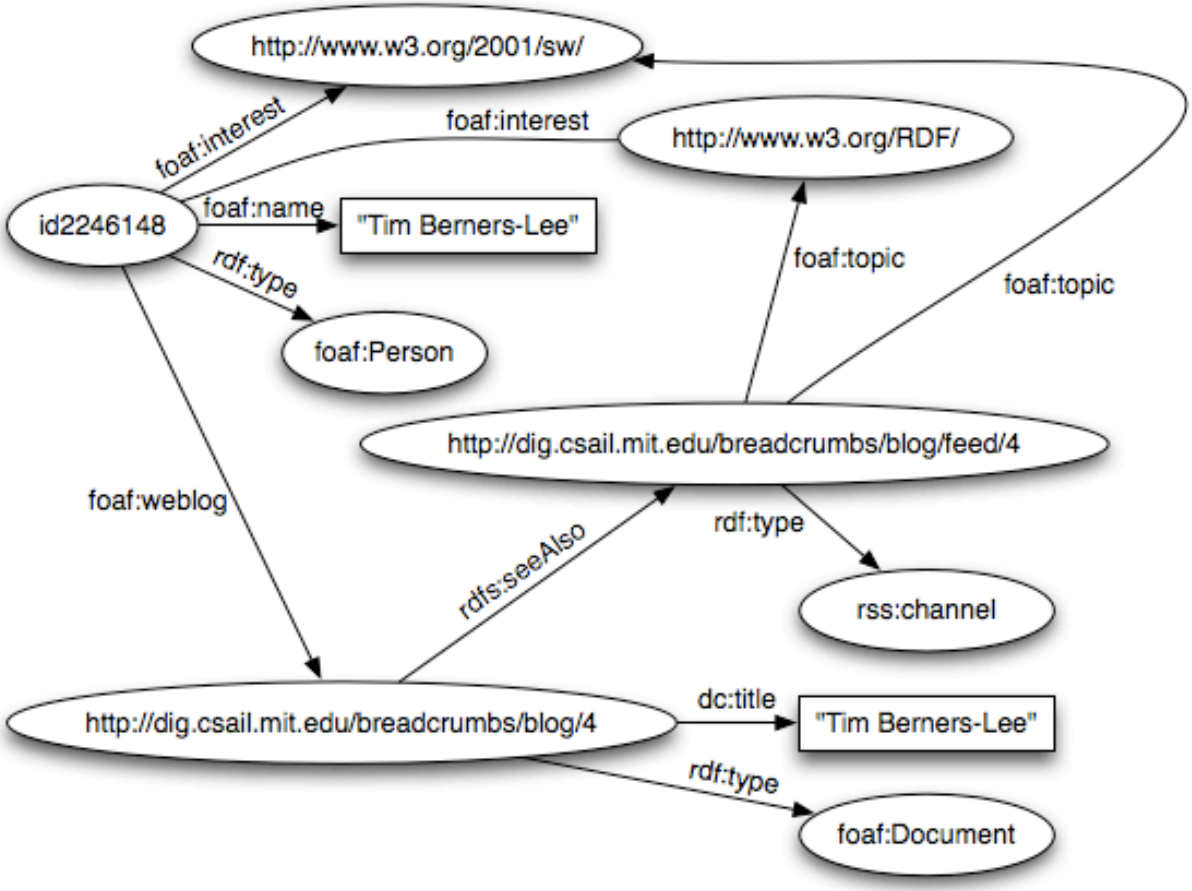
#neo4j

Graph Databases



#neo4j

RDF Triples


http://www.w3.org/2001/sw/
foaf:interest
foaf:interest
http://www.w3.org/RDF/
id2246148
foaf:name
"Tim Berners-Lee"
rdf:type
foaf:Person
foaf:topic
foaf:topic
http://dig.csail.mit.edu/breadcrumbs/blog/feed/4
foaf:weblog
rdf:type
rdfs:seeAlso
rss:channel
http://dig.csail.mit.edu/breadcrumbs/blog/4
dc:title
"Tim Berners-Lee"
rdf:type
foaf:Document

Hypergraph