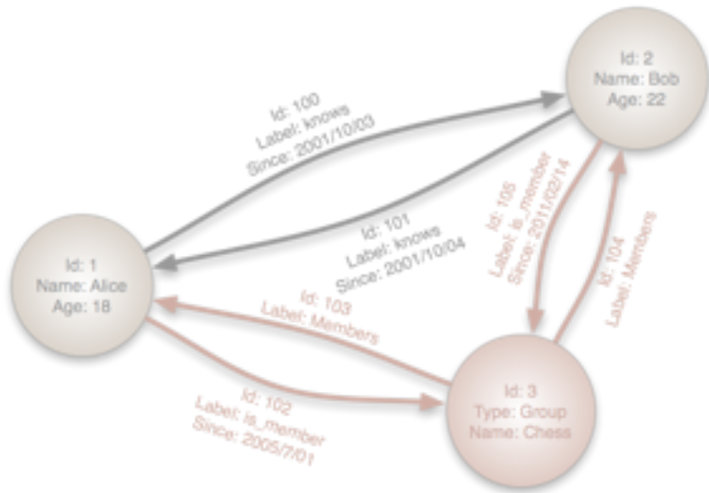
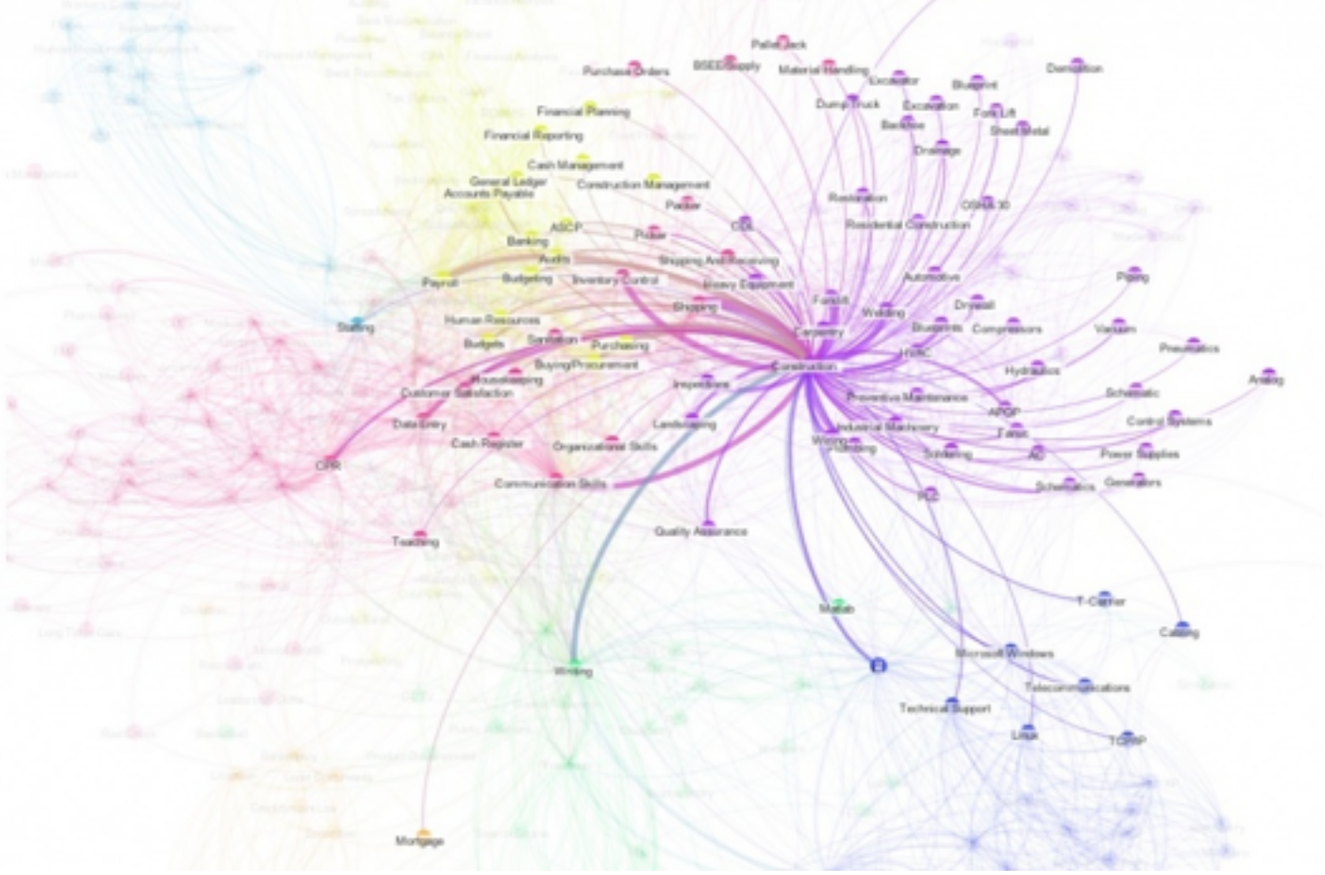
neotechnology
graphs are everywhere

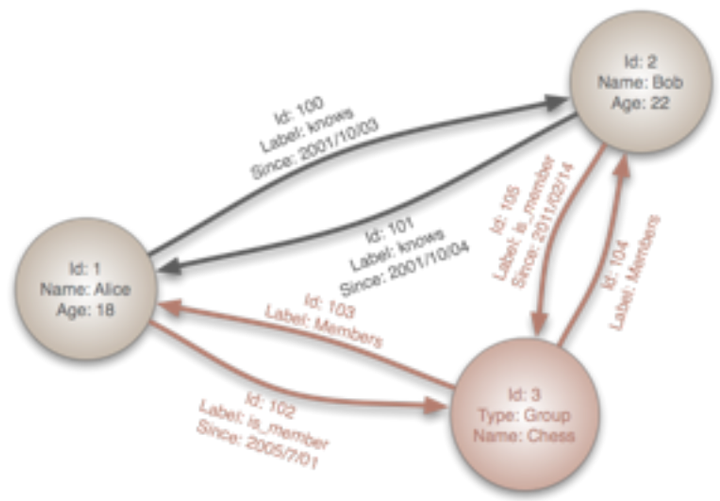
Graph Databases Use Cases



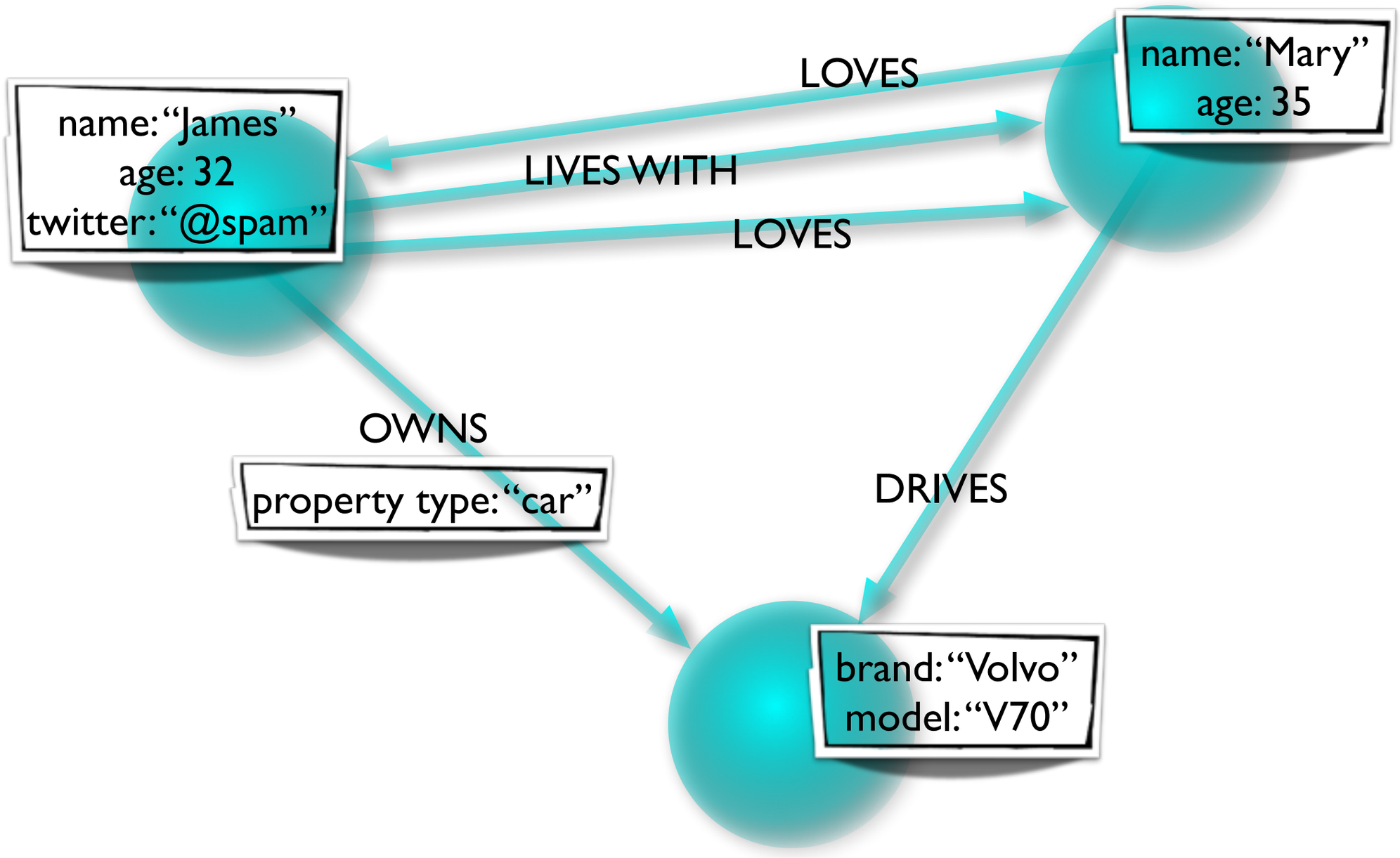
neotechnology
graphs are everywhere

What's a Graph?





Graph data model



Relational Tables

SQL Join Hell (1)

Customer		
Id	Name	Address
1	Robert	3
2	Lars	7
3	Michael	23

Address	
Id	Location
3	Berlin
4	Munich
7	Dresden
23	Leipzig

1:1 Relationship

Customer	
Id	Name
1	Robert
2	Lars
3	Michael

Address		
Id	Customer	Location
3	1	Berlin
7	2	Dresden
8	2	New York
23	3	Leipzig

1:n Relationship

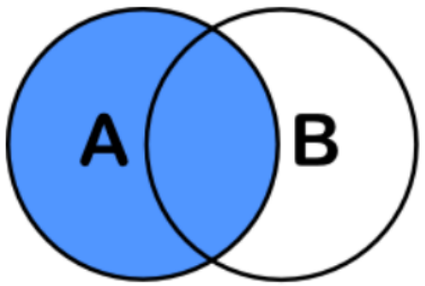
Customer	
Id	Name
1	Robert
2	Lars
3	Michael

CustomerAddress	
CId	AId
1	3
2	7
2	8
3	23

Address	
Id	Location
3	Berlin
7	Dresden
8	New York
23	Leipzig

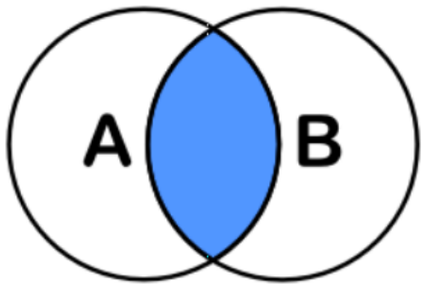
m:n Relationship

Join this way…

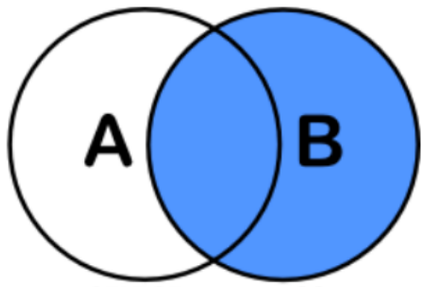


SELECT <auswahl>
FROM tabelleA A
LEFT JOIN tabelleB B
ON A.key = B.key

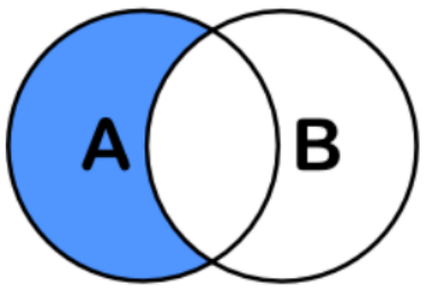
CHEATSHEET
SQL
JOINS



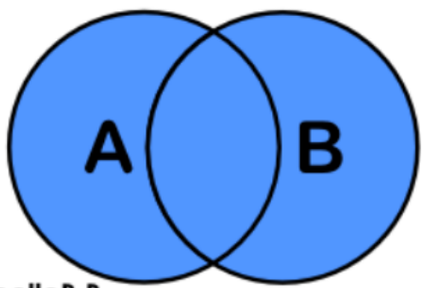
SELECT <auswahl>
FROM tabelleA A
INNER JOIN tabelleB B
ON A.key = B.key



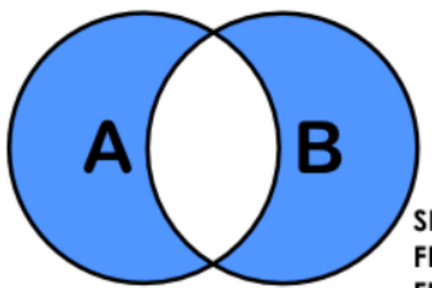
SELECT <auswahl>
FROM tabelleA A
RIGHT JOIN tabelleB B
ON A.key = B.key



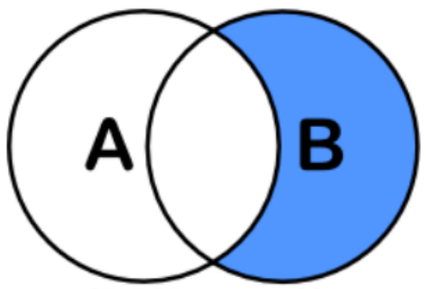
SELECT <auswahl>
FROM tabelleA A
LEFT JOIN tabelleB B
ON A.key = B.key
WHERE B.key IS NULL



SELECT <auswahl>
FROM tabelleA A
FULL OUTER JOIN tabelleB B
ON A.key = B.key



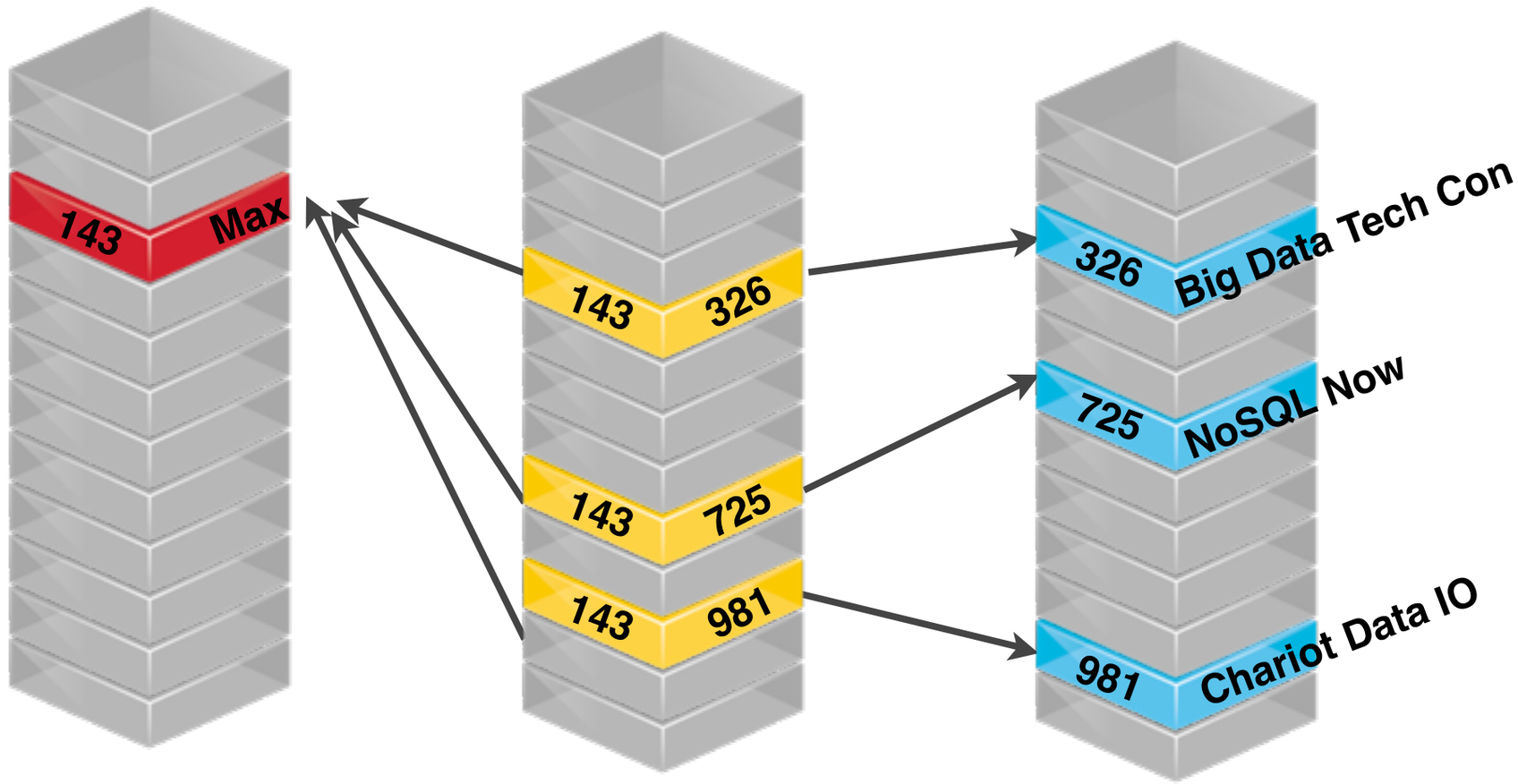
SELECT <auswahl>
FROM tabelleA A
FULL OUTER JOIN tabelleB B
ON A.key = B.key
WHERE A.key IS NULL
OR B.key IS NULL



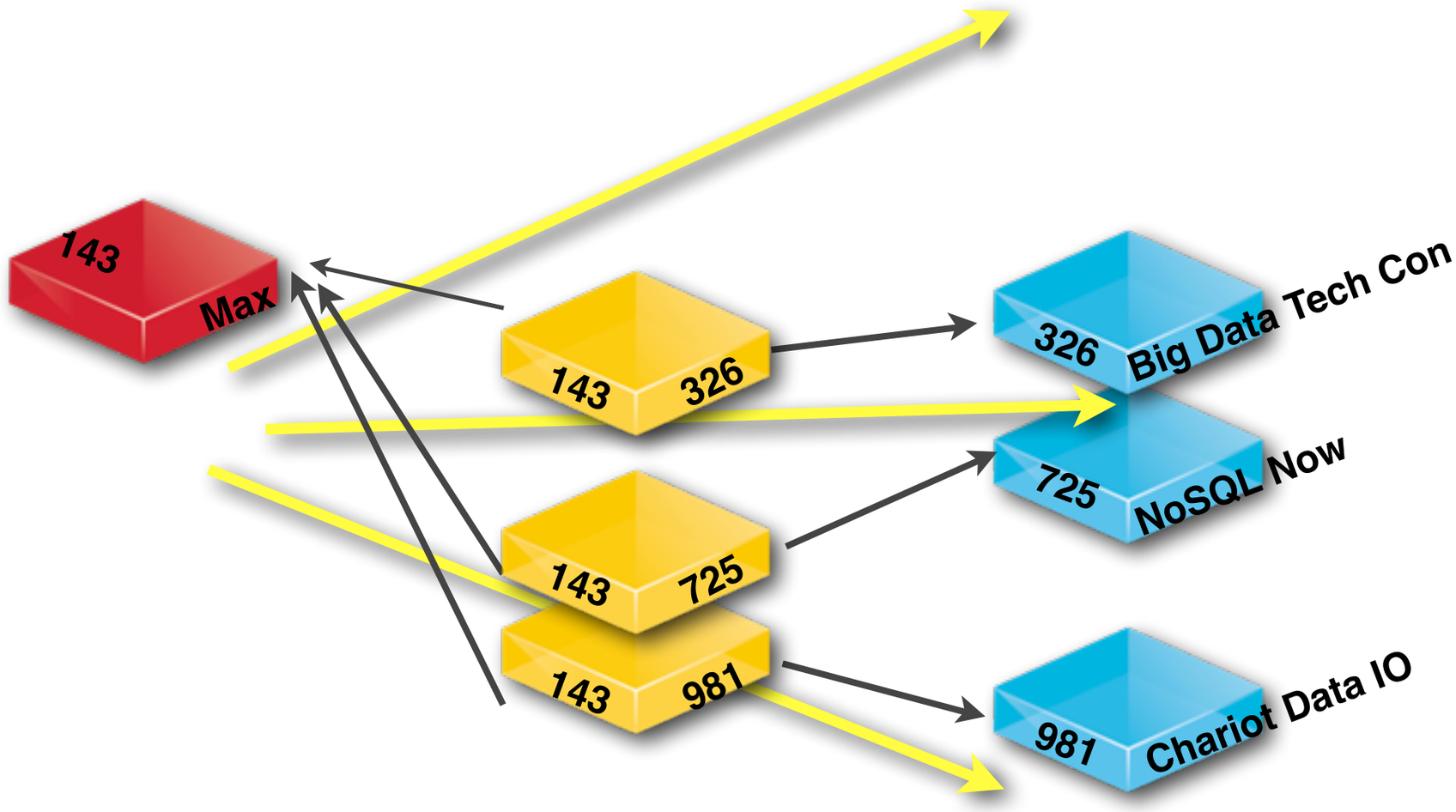
SELECT <auswahl>
FROM tabelleA A
RIGHT JOIN tabelleB B
ON A.key = B.key
WHERE A.key IS NULL

The Problem

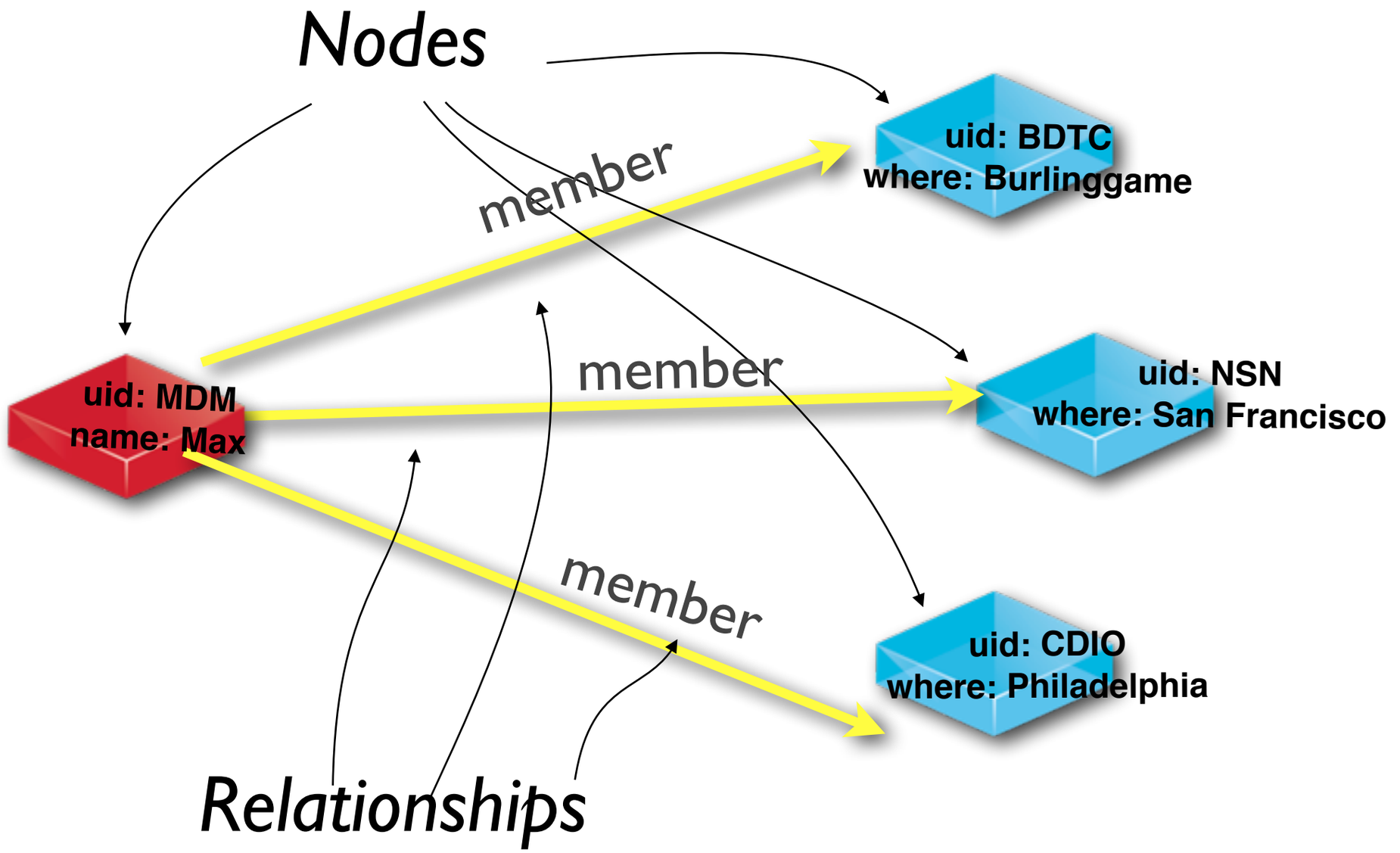
- all JOINs are executed **every time** you query (traverse) the relationship
- executing a JOIN means to **search** for a key in another table
- with Indices executing a JOIN means to lookup a key
- B-Tree Index: O(log(n))
- more entries => more lookups => slower JOINs



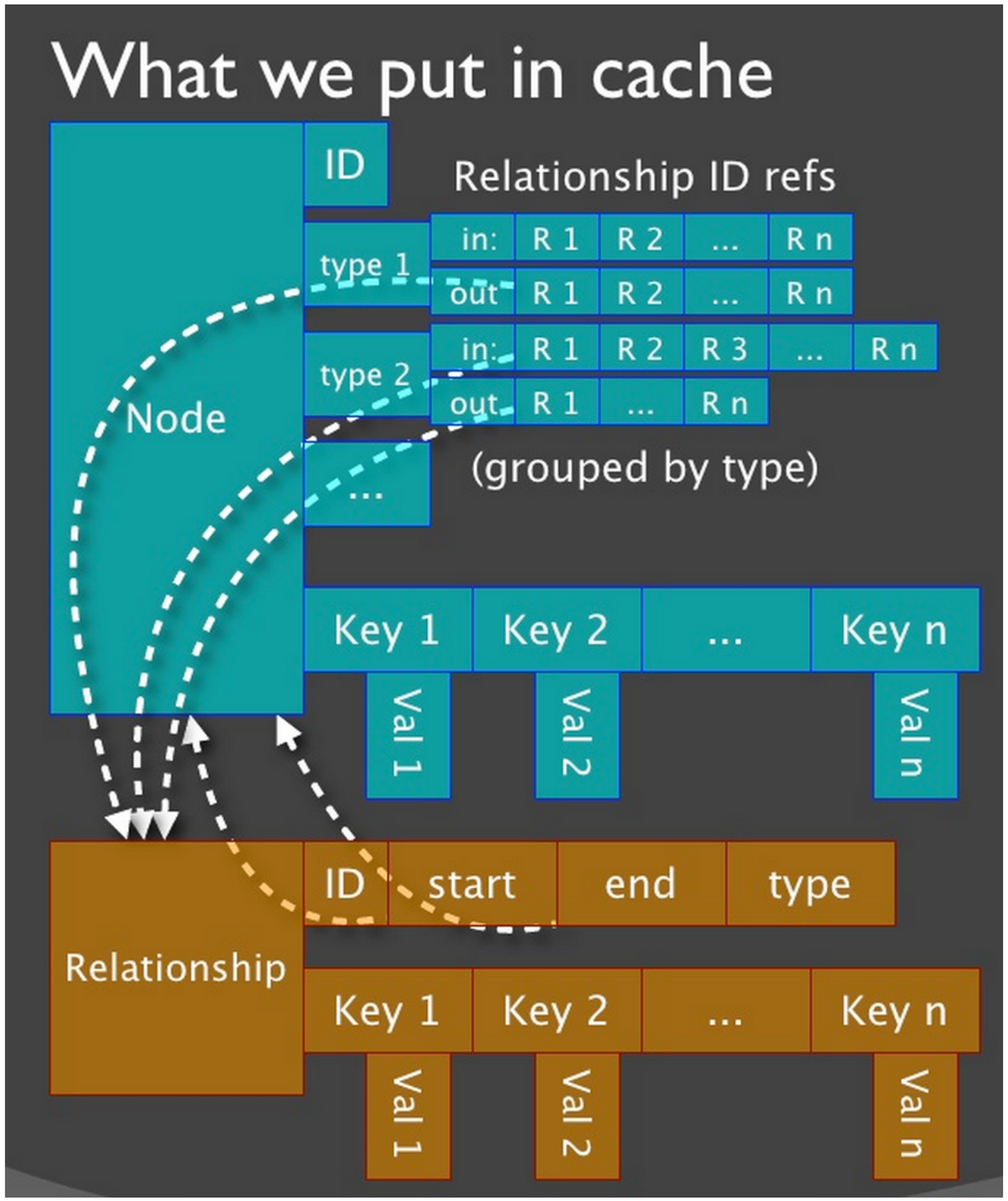
People Attend Conferences


143
Max
143
326
326
Big Data Tech Con
725
NoSQL Now
143
725
143
981
981
Chariot Data IO

A Property Graph



The Neo4j Secret Sauce



- Pointers instead of look-ups
- Do all your "Joining" on creation
- Spin spin spin through this data structure

Graph Buzz!

neo technology
graphs are everywhere

MacArthur 'Genius Grant' Winner Maria Chudnovsky on Graph Theory

Wednesday, October 03, 2012

Bright Launches Bright Packed With Jobs Data Seeking Tips

InfoWorld Home / InfoWorld Tech Watch / Buzz grows around graph databases

The First Word on Tech
INFOWORLD TECH WATCH

AUGUST 29, 2012

Buzz grows around graph databases

Interest in graph databases will continue to grow, given its ability to analyze data delivered in a non-relational format, such as social networking data

By Paul Krill | InfoWorld

Follow @pjkrill

Graph Databases: The New Way to Access Super Fast Social Data

September 26, 2012 by Emil Eifrem

1

259 Like

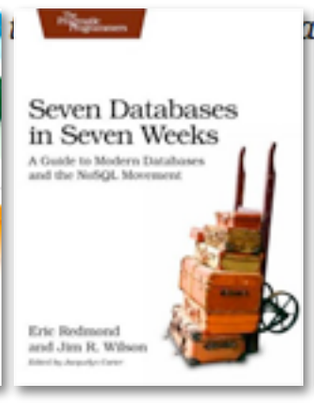
969 Tweet

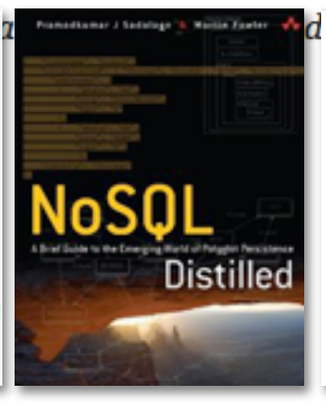
48 +1

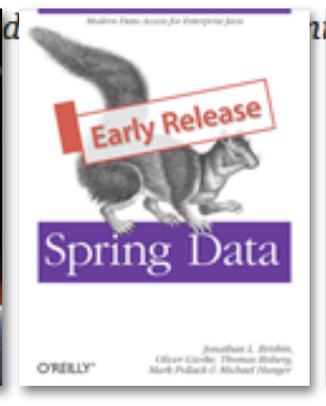
173 Share

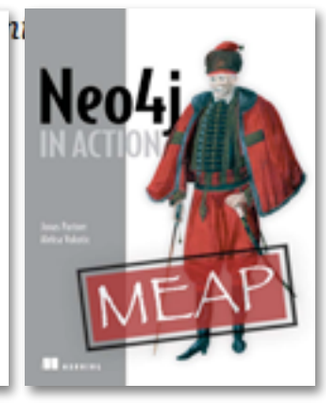
Facebook's Social Graph, Neo4j show rising use of graph databases

Summary: *Facebook's Social Graph -- the database underlying its Graph Search engine unveiled yesterday-- is just one of many graph databases being employed for complex, connected data. Neo4j*









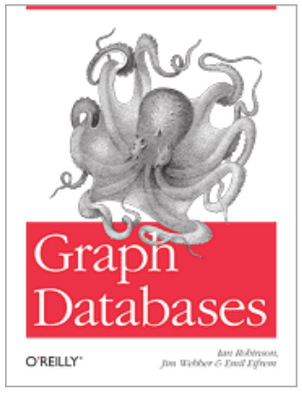
I saw my own Interest Graph and it's scary-accurate. We'd certainly pay for the ability to use the Gravity personalization technology I saw today at TechCrunch to help target content to users.

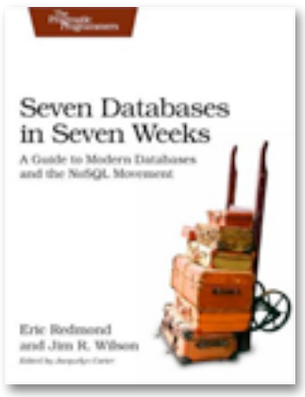
TC **Michael Arrington, TechCrunch**

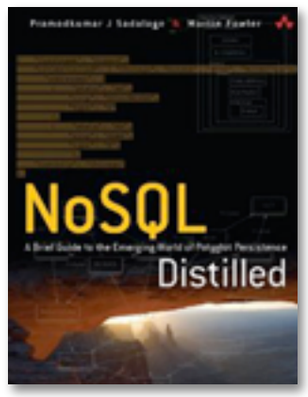
The Neo4j Graph Database

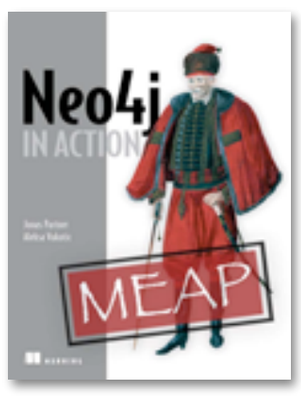
neotechnology
graphs are everywhere

- Neo4j is the **leading graph database** in the world today
  - Most widely deployed: **500,000+ downloads**
  - **Largest ecosystem**: active forums, code contributions, etc
  - Most mature product: in development since 2000, **in 24/7 production since 2003**









Early Adopters of Graph Tech

neotechnology
graphs are everywhere

facebook

Google™

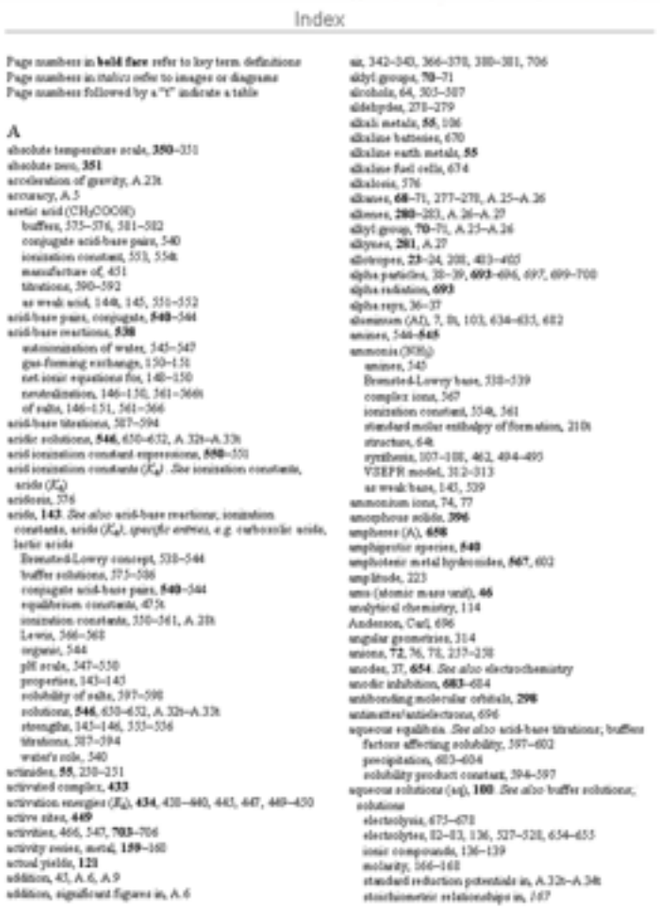
Survival of the Fittest

Evolution of Web Search

neotechnology
graphs are everywhere

Pre-1999

WWW Indexing

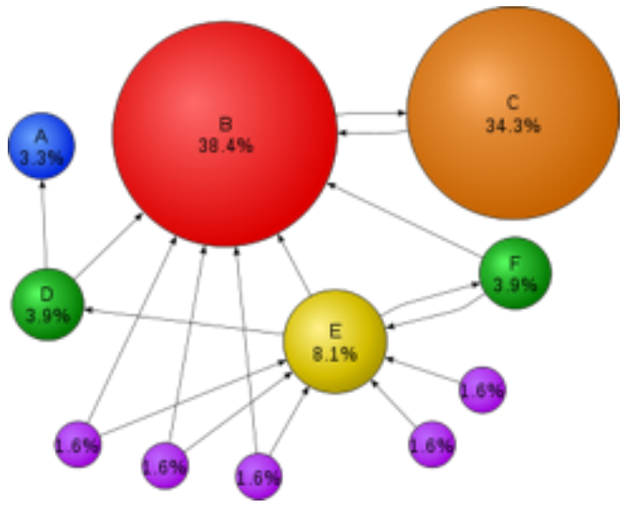

Index

Page numbers in **bold face** refer to key term definitions
Page numbers in *italics* refer to images or diagrams
Page numbers followed by a "t" indicate a table

A
absolute temperature scale, **350**–351
absolute zero, **351**
acceleration of gravity, A-23t
accuracy, A-5
acetic acid (CH_3COOH)
  buffers, 575–576, 581–582
  conjugate acid-base pairs, 540
  ionization constant, 553, 554t
  manufacture of, 451
  titrations, 590–592
  as weak acid, 144t, 145, 551–552
acid-base pairs, conjugate, **540**–544
acid-base reactions, **538**
  autoionization of water, 545–547
  gas-forming exchange, 150–151
  net ionic equations for, 148–150
  neutralization, 146–150, 561–566t
  of salts, 146–151, 561–566
acid-base titrations, 587–594
acidic solutions, **546**, 630–632, A-32t–A-33t
acid ionization constant expressions, **650**–651
acid ionization constants (K_a). *See* ionization constants, acids (K_a)
acidosis, 576
acids, **143**. *See also* acid-base reactions; ionization constants, acids (K_a); *specific entries, e.g.* carboxylic acids, lactic acids
  Brønsted-Lowry concept, 538–544
  buffer solutions, 575–586
  conjugate acid-base pairs, **540**–544
  equilibrium constants, 675t
  ionization constants, 550–561, A-21t
  Lewis, 566–568
  organic, 544
  pH scale, 547–550
  properties, 143–145
  solubility of salts, 597–598
  solutions, **546**, 630–632, A-32t–A-33t
  strengths, 143–146, 553–556
  titrations, 587–594
  water's role, 540
actinides, **55**, 250–251
activated complex, **433**
activation energies (E_a), **434**, 438–440, 443, 447, 449–450
active sites, **669**
activities, 666, 547, **703**–706
activity series, metal, **159**–160
actual yields, **121**
addition, 43, A-6, A-9
addition, significant figures in, A-6
air, 342–343, 366–370, 380–381, 706
alkyl groups, **70**–71
alcohols, 64, 303–307
aldehydes, 278–279
alkali metals, **55**, 106
alkaline batteries, 670
alkaline earth metals, **55**
alkaline fuel cells, 674
alkalosis, 576
alkanes, **68**–71, 277–278, A-25–A-26
alkenes, **280**–283, A-26–A-27
alkyl group, **70**–71, A-23–A-26
alkynes, **281**, A-27
allotropes, 23–24, 208, 403–405
alpha particles, 38–39, **693**–696, 697, 699–700
alpha radiation, **693**
alpha rays, 36–37
aluminum (Al), 7, 10, 103, 634–635, 682
amines, 544–**545**
ammonia (NH_3)
  amines, 545
  Brønsted-Lowry base, 538–539
  complex ions, 567
  ionization constant, 554t, 561
  standard molar enthalpy of formation, 210t
  structure, 64t
  synthesis, 107–108, 462, 494–495
  VSEPR model, 312–313
  as weak base, 143, 539
ammonium ions, 74, 77
amorphous solids, **396**
amphoteric species, **540**
amphoteric metal hydroxides, **667**, 602
amplitude, 223
amu (atomic mass unit), **46**
analytical chemistry, 114
Anderson, Carl, 696
angular geometries, 314
anions, **72**, 76, 78, 237–238
anodes, 37, **654**. *See also* electrochemistry
anodic inhibitors, **683**–684
antibonding molecular orbitals, **298**
antimatter/antielectrons, 696
aqueous equilibria. *See also* acid-base titrations; buffers
  factors affecting solubility, 597–602
  precipitation, 603–604
  solubility product constant, 594–597
aqueous solutions (aq), **100**. *See also* buffer solutions; solutions
  electrolysis, 675–678
  electrolytes, 82–83, 136, 527–528, 654–655
  ionic compounds, 136–139
  molarity, 166–168
  standard reduction potentials in, A-32t–A-34t
  stoichiometric relationships in, *167*

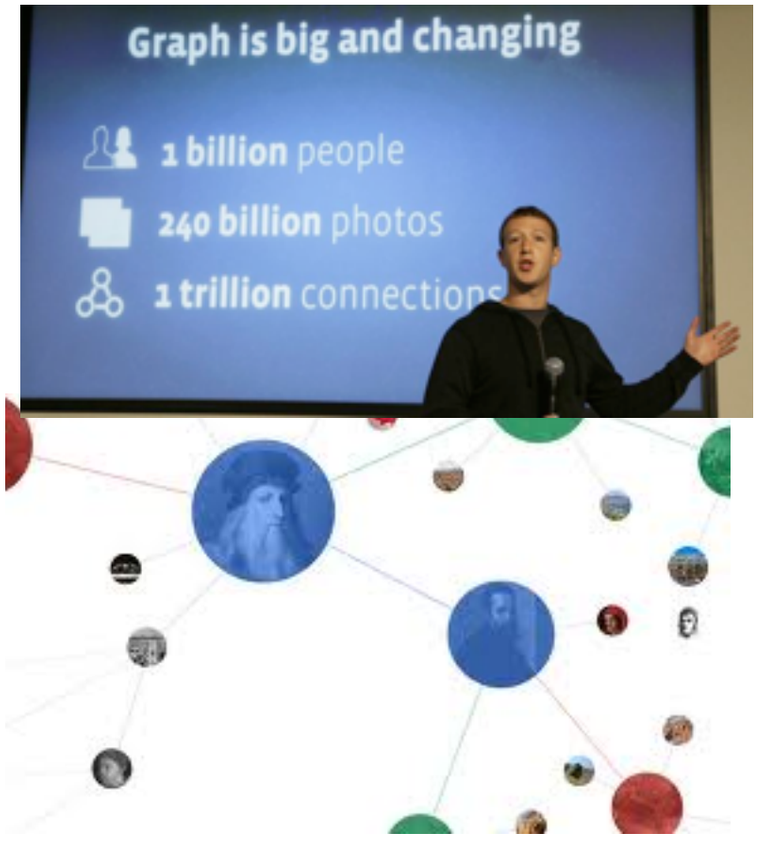
1999 - 2012

Google Invents PageRank



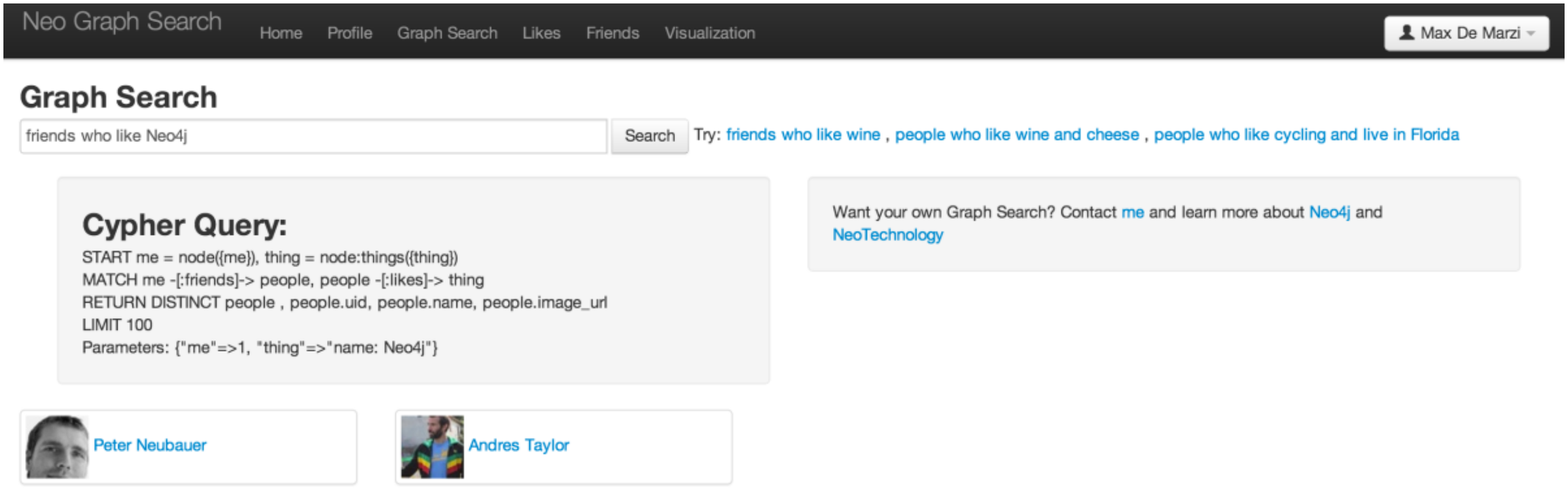
2012-?

Google Knowledge Graph, Facebook Graph Search



Discrete Data + Connected Data (Simple) + Connected Data (Rich)

Open Source Example



http://maxdemarzi.com/2013/01/28/facebook-graph-search-with-cypher-and-neo4j/

Survival of the Fittest

Evolution of Online Recruiting

neotechnology
graphs are everywhere

1999

Keyword Search

Home > **Job Search**

Search over **150,000** U.S. jobs.
Perform your search below or get tips on searching.

Location Search:

-------- Select a location --------
Alaska-Anchorage
Alaska-Fairbanks
Alaska-Juneau
Alaska-Valdez
Alabama-Anniston

Category Search:

-------- Select a category --------
Accounting/Finance/Banking
Administrative/Clerical
Creative Arts/Media
Education/Training
Engineering/Architecture/Design

Keyword Search:

Search Jobs | Clear

2011-12

Social Discovery

glassdoor™ careerbuilder®
viadeo CareerArcGroup

Most jobs are found through an inside connection
Each friend that joins Glassdoor allows you to see more connections at more companies

9 friends on glassdoor	3,905 inside connections	3,882 companies

Invite more friends — ask them to share their connections

Jobs with Connections

Sr. Statistical Analyst, Product Innovation
Netflix – Los Gatos, CA
From: Job.com – 1 days ago

Creative Director
frog design – San Francisco, CA
From: Experteer – 8 days ago

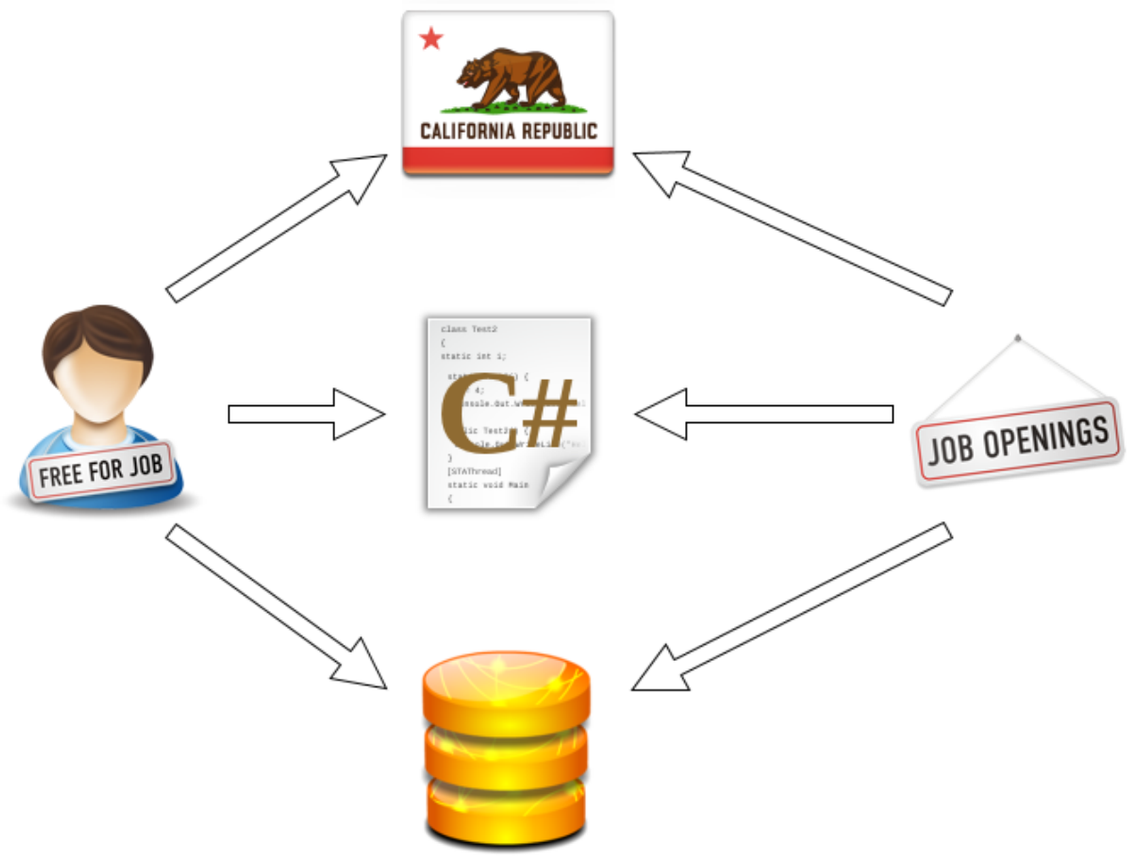
Java Server Software Engineer
Electronic Arts – Redwood City, CA
From: Experteer – 3 days ago

EMERGENCY MEDICAL TECH
U. S. NAVY – Fremont, CA
From: Monster – 14 days ago

Want better jobs? Tell us your current job title

Discrete Data + Connected Data

Open Source Example



http://maxdemarzi.com/2012/10/18/matches-are-the-new-hotness/

Open Source Example

```
START me=node:users_index(name={user})
MATCH skills<-[:has]-me-[:lives_in]->city<-[:in_location]-job-[:requires]->requirements
WHERE me-[:has]->()<-[:requires]-job
WITH DISTINCT city.name AS city_name,
     job.name AS job_name,
     LENGTH(me-[:has]->()<-[:requires]-job) AS matching_skills,
     LENGTH(job-[:requires]->()) AS job_requires,
     COLLECT(DISTINCT requirements.name) AS req_names,
     COLLECT(DISTINCT skills.name) AS skill_names
RETURN city_name, job_name,
FILTER(name IN req_names WHERE NOT name IN skill_names) AS missing
ORDER BY matching_skills / job_requires DESC, job_requires
LIMIT 10
```

http://maxdemarzi.com/2012/10/18/matches-are-the-new-hotness/

Open Source Example

Matching Jobs for Daniel

Reload the page or click Match Another User

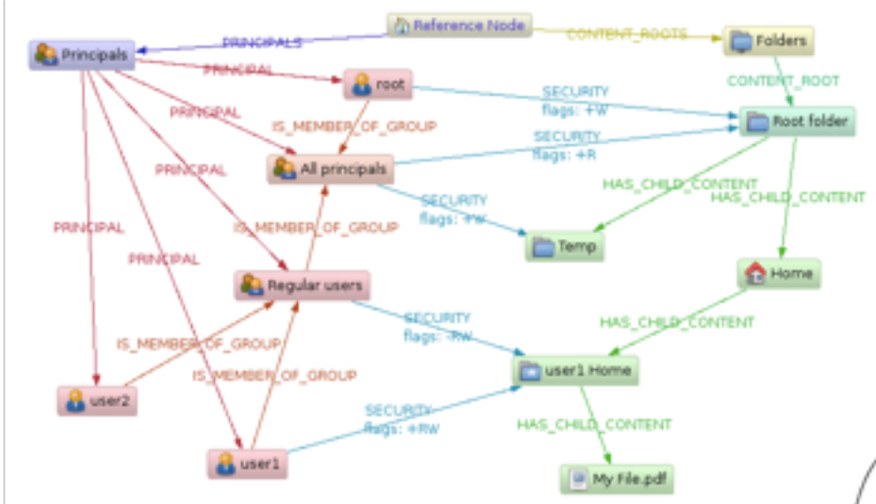
Job	Match	Missing
Spring-C-Django-SQL in *Dallas*	100% Match	
Neo4j-C-CSS in *Dallas*	Partial Match	+ CSS
C-Spring-Java in *Dallas*	Partial Match	+ Java
Java-Redis-C-Neo4j in *Dallas*	Partial Match	+ Java + Redis

http://maxdemarzi.com/2012/10/18/matches-are-the-new-hotness/

Emergent Graph in Other Industries

(Actual Neo4j Graphs)

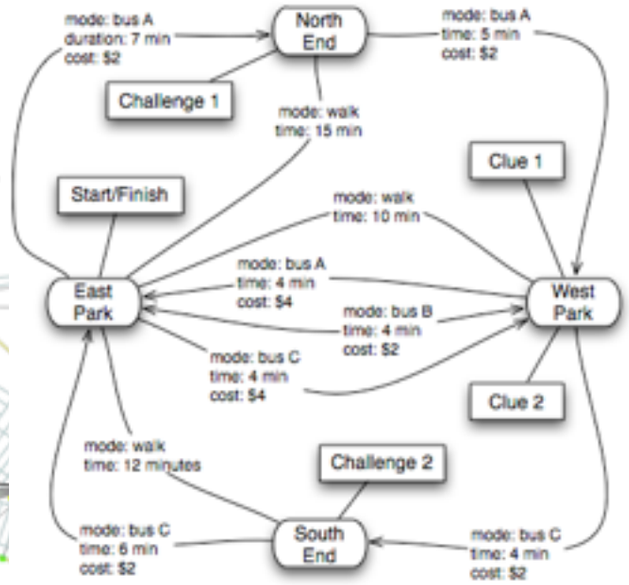
Content Management & Access Control



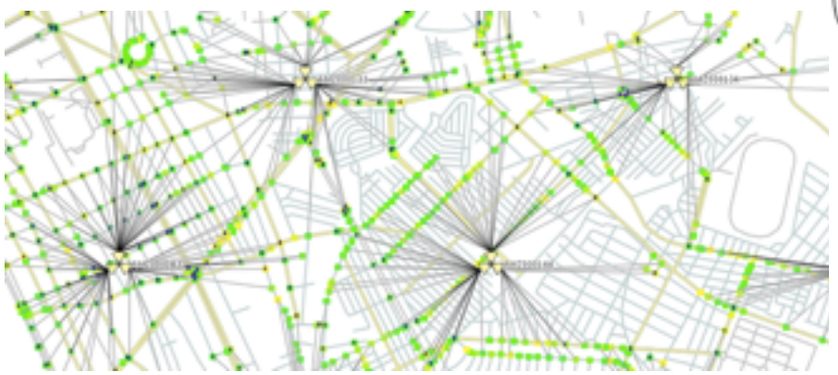
Insurance Risk Analysis

Geo Routing

(Public Transport)



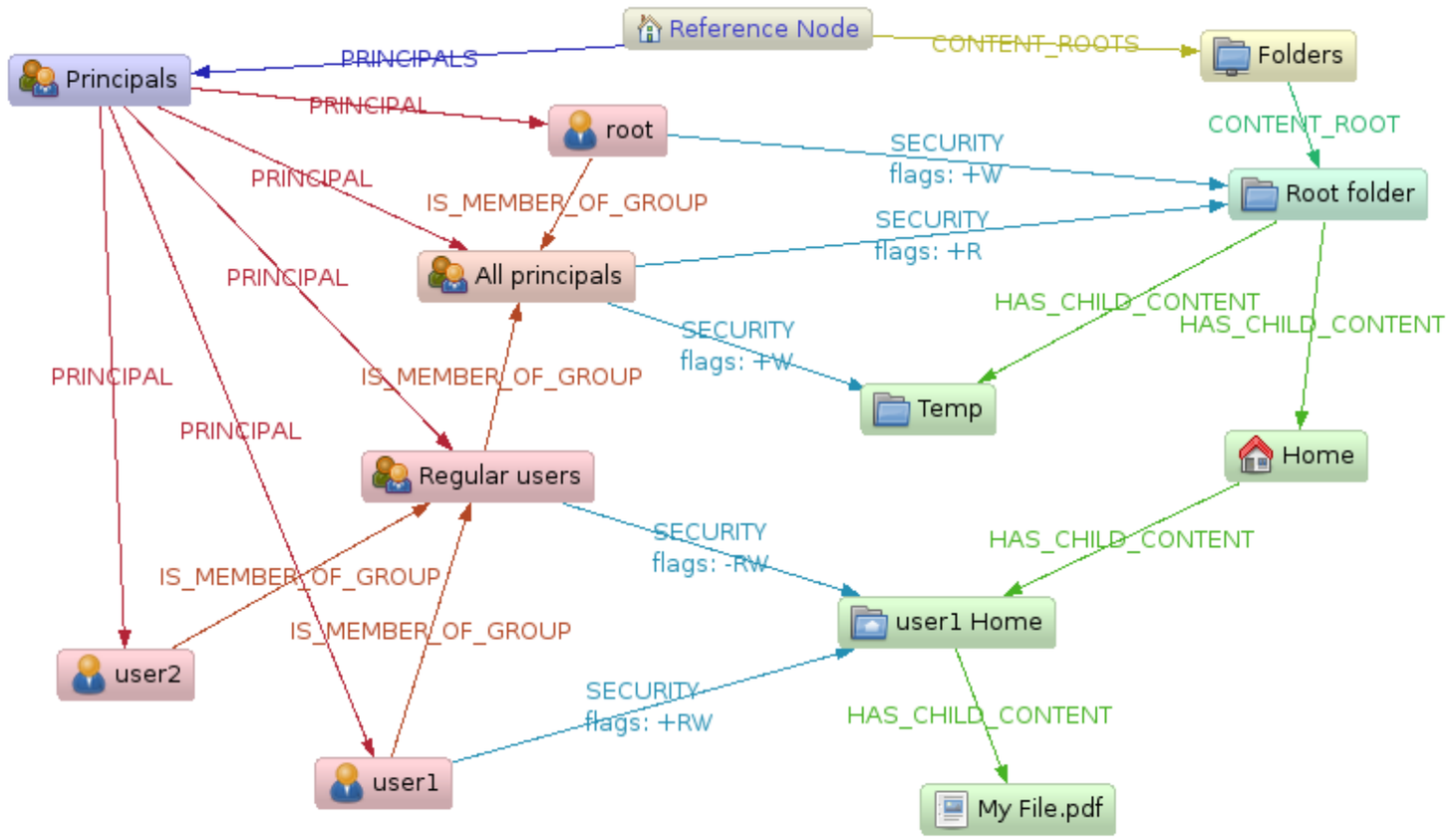
Network Cell Analysis



Network Asset Management

BioInformatics

Open Source Example



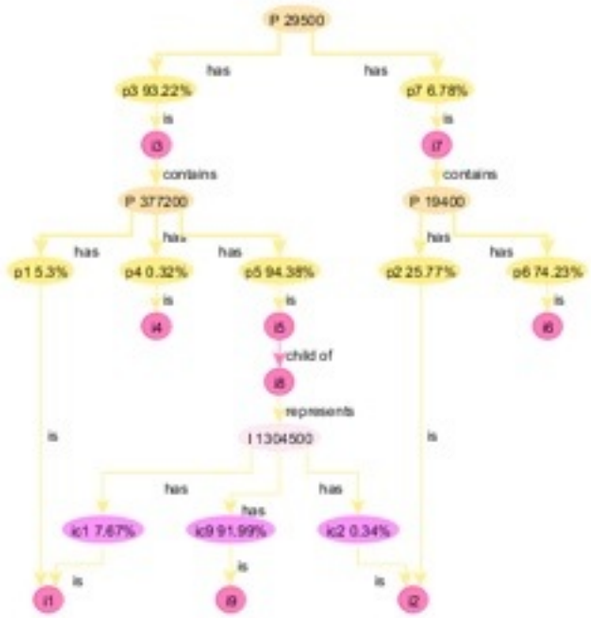
http://maxdemarzi.com/2013/03/18/permission-resolution-with-neo4j-part-1/

Emergent Graph in Other Industries

(Actual Neo4j Graphs)

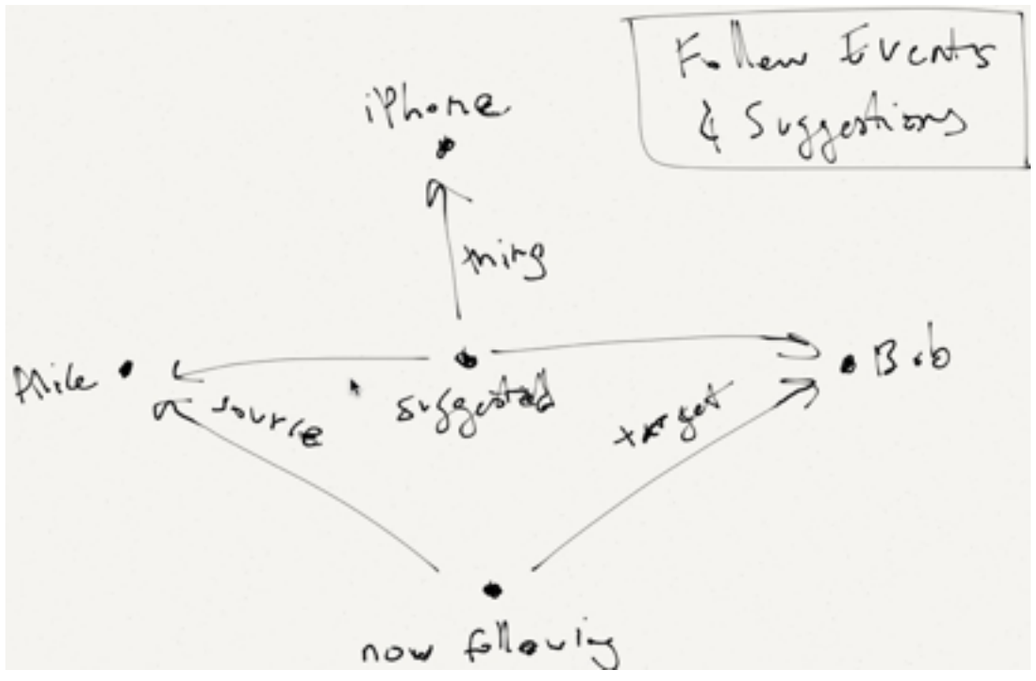
Web Browsing

Portfolio Analytics



Gene Sequencing

Mobile Social Application



Open Source Example

Neo Love Matching Match Another User

Matching Loves for Luvenia

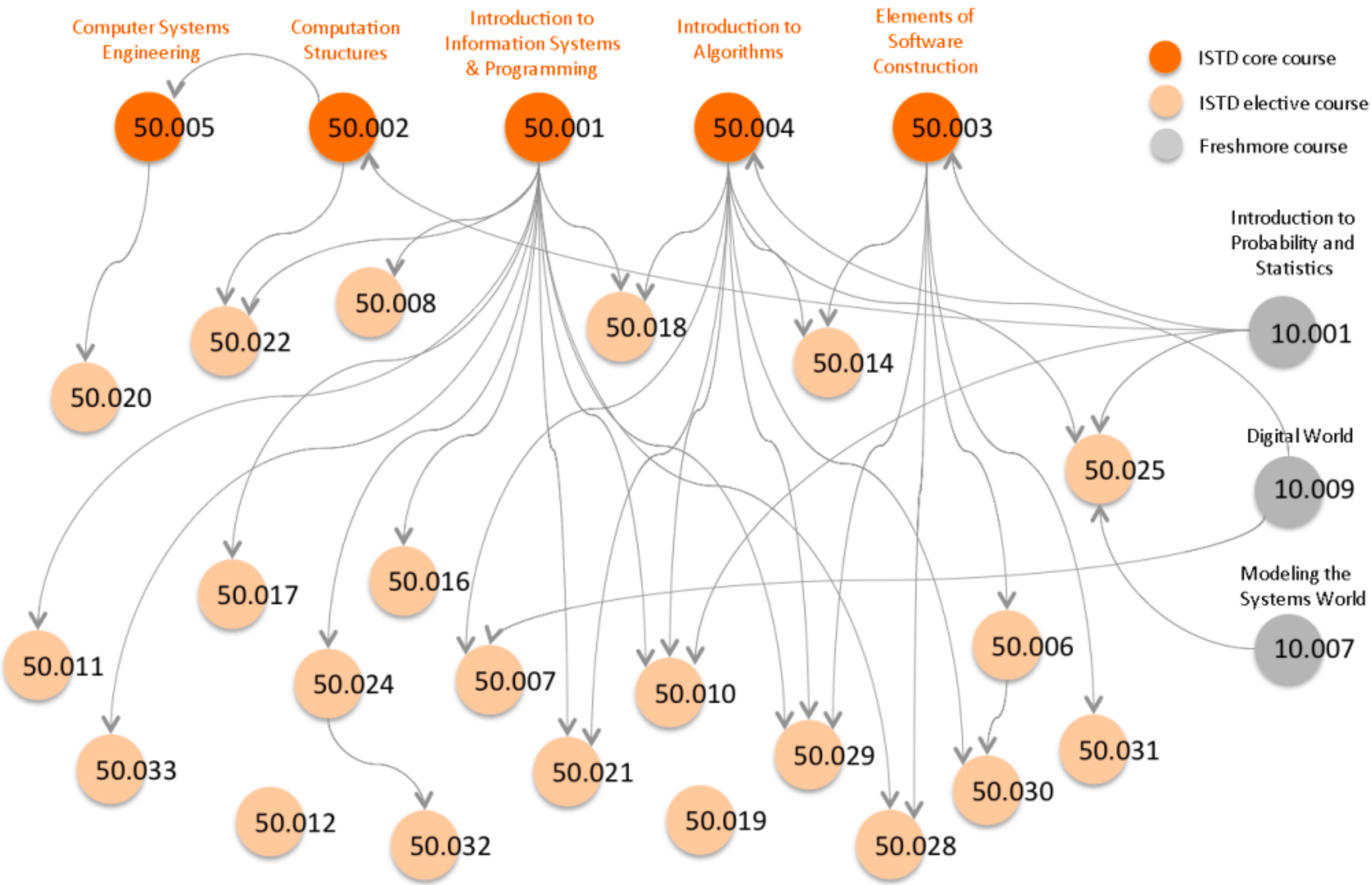
Reload the page or click Match Another User

Love	Compatibility	Your Matches	Their Matches
Chet in *Philadelphia*	2.0	✔ Humorous ✔ Thoughtful	✔ Likeable
Mckinley in *Philadelphia*	1.0	✔ Serious	✔ Creative
Gene in *Philadelphia*	1.0	✔ Serious	✔ Likeable
Keith in *Philadelphia*	1.0	✔ Quiet	✔ Inventive

http://maxdemarzi.com/2013/04/19/match-making-with-neo4j/

Curriculum Graph

SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN
Established in collaboration with MIT



Early Adopter Segments

(What we expected to happen - view from several years ago)

Core Industries & Use Cases:	Web / ISV	Finance & Insurance	Datacom / Telecom
Network & Data Center Management			
MDM			
Social			
Geo			

Neo4j Adoption Snapshot

Select Commercial Customers (Community Users Not Included)

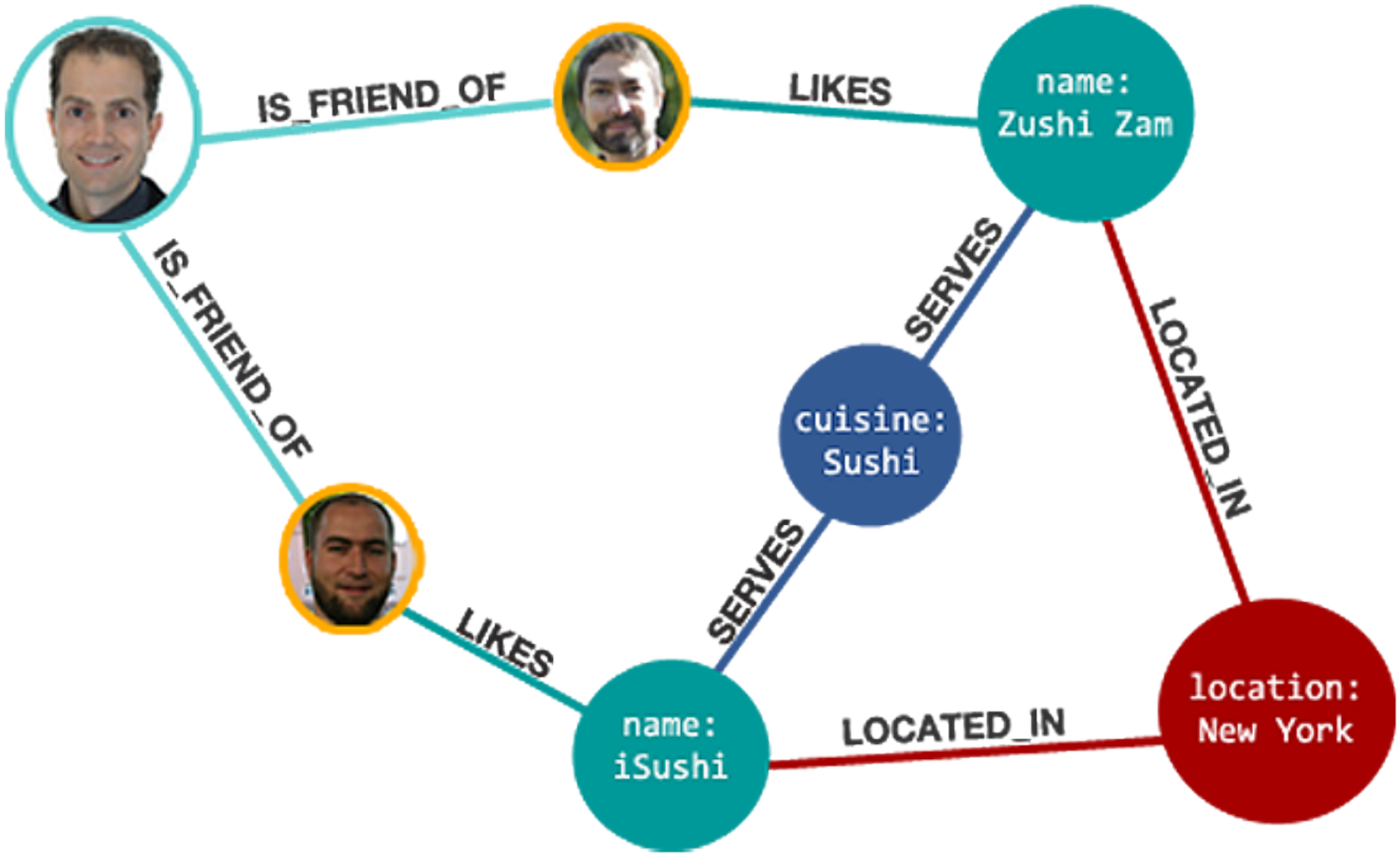
neotechnology
graphs are everywhere

Core Industries & Use Cases:	Software	Financial Services	Telecommunications	Web Social, HR & Recruiting	Health Care & Life Sciences	Media & Publishing	Energy, Services, Automotive, Gov't, Logistics, Education, Gaming, Other
Network & Data Center Management	Zenoss, Juniper, NetApp, Serena, gen, Virtual Instruments		HP, SFR				
MDM / System of Record			Cisco, Deutsche Telekom, maaii	Wooz, Equilar, viadeo, glassdoor, CareerArcGroup, classmates.com	Zephyr Health, HealthUnlocked, SharePractice		Juice Plus+, onefinestay, teachscape
Social	Glowbl	ICE				LifeChurch.tv, Squidoo, indiatimes	
Geo	DingLicom		Justdial				gamesys, Accenture, Global 500 Logistics, shuti
Identity & Access Mgmt	aikux.com, identropy	Global 500 Finance	telenor				
Content Management	springcm, Adobe, Liquid Common			Dshini, SHRM, SevenBridges, hinge, careerbuilder, InfoJobs, moviepilot	Curaspan Healthgroup, Janssen	zeebox, LifeWay, decibel, <fuseworks/>, Perigee, CHIP	DOSB New Media GmbH, research now, compete
Recommend-ations	Axon Active, kitedesk						DRAKER, Impact Technologies, Lockheed Martin
BI, CRM, Impact Analysis, Fraud Detection, Resource Optimization, etc.	SODIFRANCE, idmission, Humanvest.co, DRW Trading Group, NexLP		Global 500 Telecommunication				Global 500 Energy, Global 500 Aerospace

neo technology
graphs are everywhere

What Can You Do With Graphs?


Sushi restaurants in **New York, New York** that my friends like
Sushi restaurants in **New York, New York** that my friends like

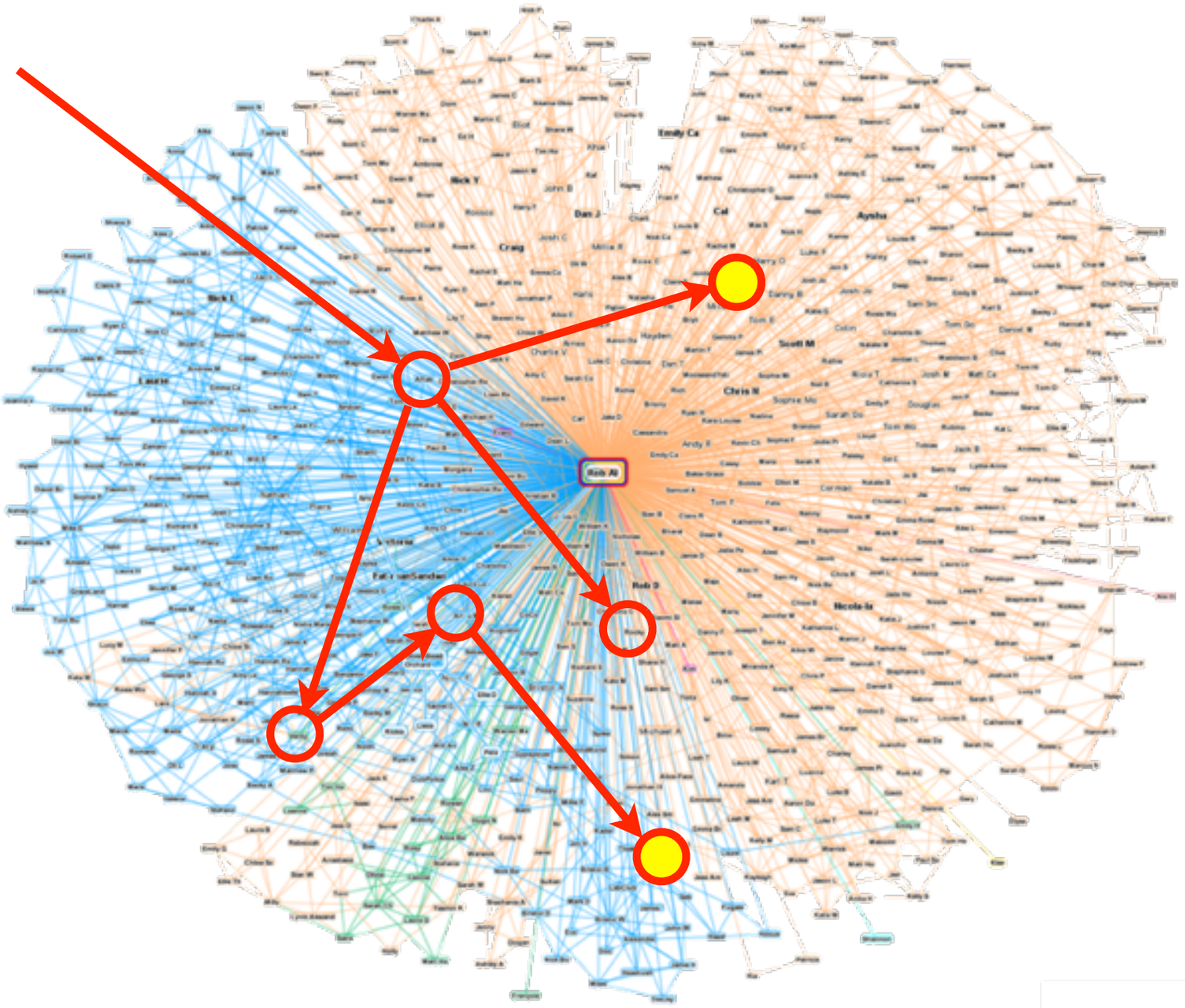

IS_FRIEND_OF
LIKES
name:
Zushi Zam
IS_FRIEND_OF
SERVES
LOCATED_IN
cuisine:
Sushi
LIKES
SERVES
name:
iSushi
LOCATED_IN
location:
New York


```
MATCH (me:Person)-[:IS_FRIEND_OF]->(friend),
    (friend)-[:LIKES]->(restaurant),
       (restaurant)-[:LOCATED_IN]->(city:Location),
       (restaurant)-[:SERVES]->(cuisine:Cuisine)

WHERE me.name = 'Philip' AND city.location='New York' AND
      cuisine.cuisine='Sushi'

RETURN restaurant.name
```

http://maxdemarzi.com/?s=facebook

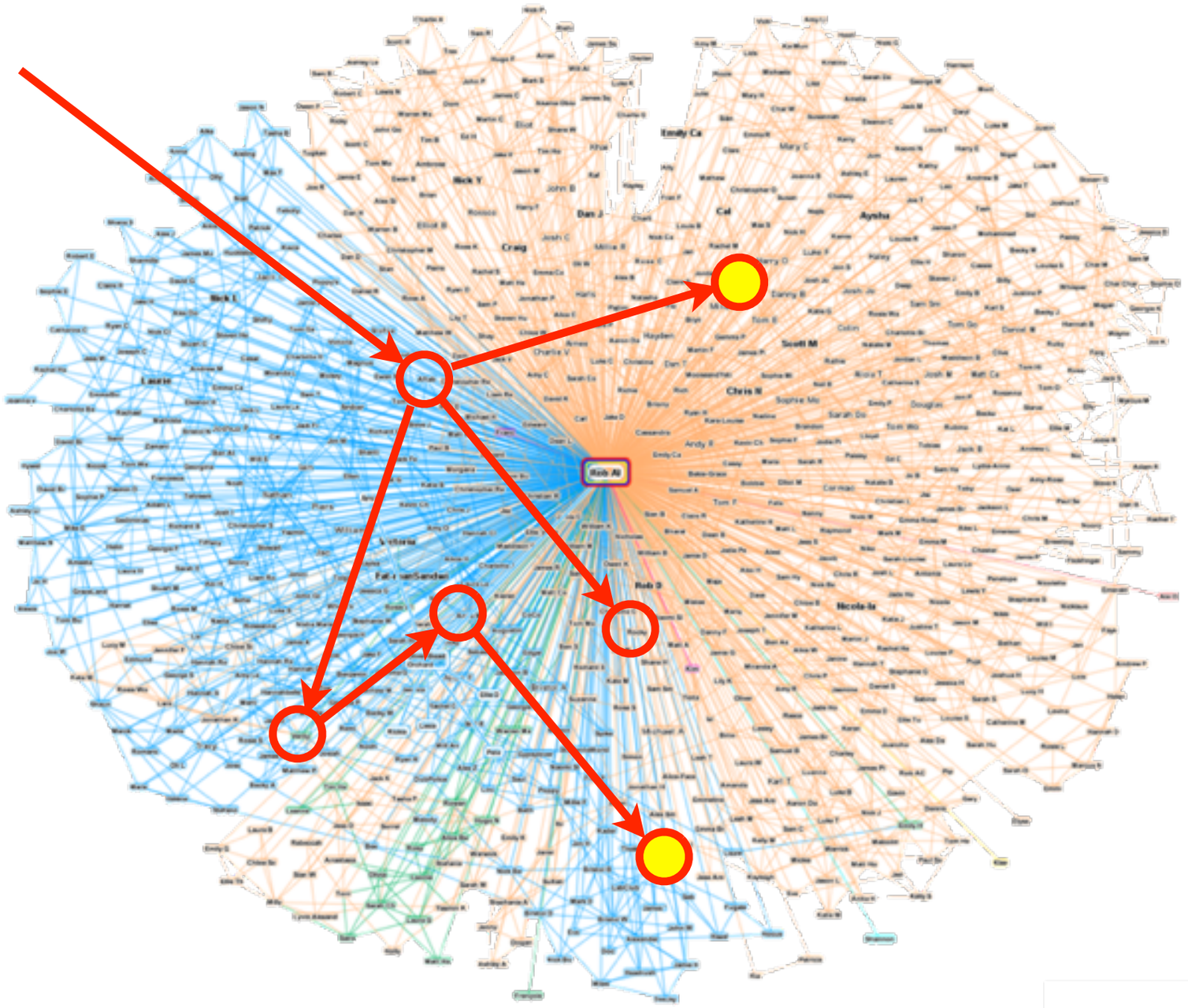
** Cypher query language example*


Sushi restaurants in **New York, New York** that my friends like
Sushi restaurants in **New York, New York** that my friends like



Of course.. a graph is a graph is a graph

What drugs will bind to protein X and not interact with drug Y?



neo technology
graphs are everywhere

Connected Query Performance

Connected Query Performance

Query Response Time =*
f(graph density, graph size, query degree)

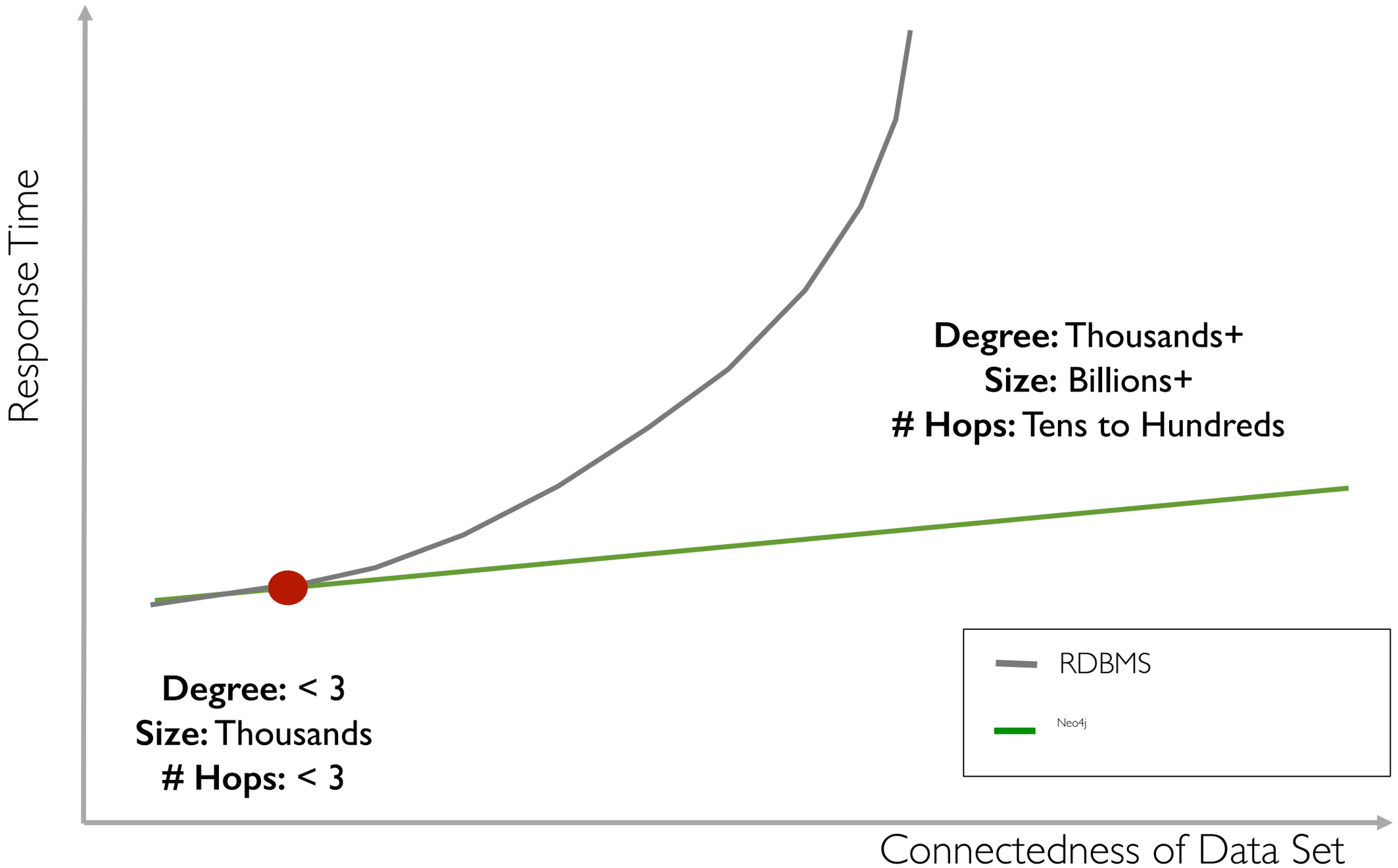
- **Graph density** (avg # rel's / node)
- **Graph size** (total # of nodes in the graph)
- **Query degree** (# of hops in one's query)

RDBMS:
>> exponential slowdown as each factor increases

Neo4j:
>> Performance remains constant as graph size increases
>> Performance slowdown is linear or better as density & degree increase

Connected Query Performance

RDBMS vs. Native Graph Database



Graph db performance

- a sample social graph
  - with ~1,000 persons
- average 50 friends per person
- pathExists(a,b) limited to depth 4
- caches warmed up to eliminate disk I/O

Database	# persons	query time
MySQL		
Neo4j		
Neo4j		

Graph db performance

- a sample social graph
  - with ~1,000 persons
- average 50 friends per person
- pathExists(a,b) limited to depth 4
- caches warmed up to eliminate disk I/O

Database	# persons	query time
MySQL	1,000	2,000 ms
Neo4j		
Neo4j		

Graph db performance

- a sample social graph
  - with ~1,000 persons
- average 50 friends per person
- pathExists(a,b) limited to depth 4
- caches warmed up to eliminate disk I/O

Database	# persons	query time
MySQL	1,000	2,000 ms
Neo4j	1,000	2 ms
Neo4j		

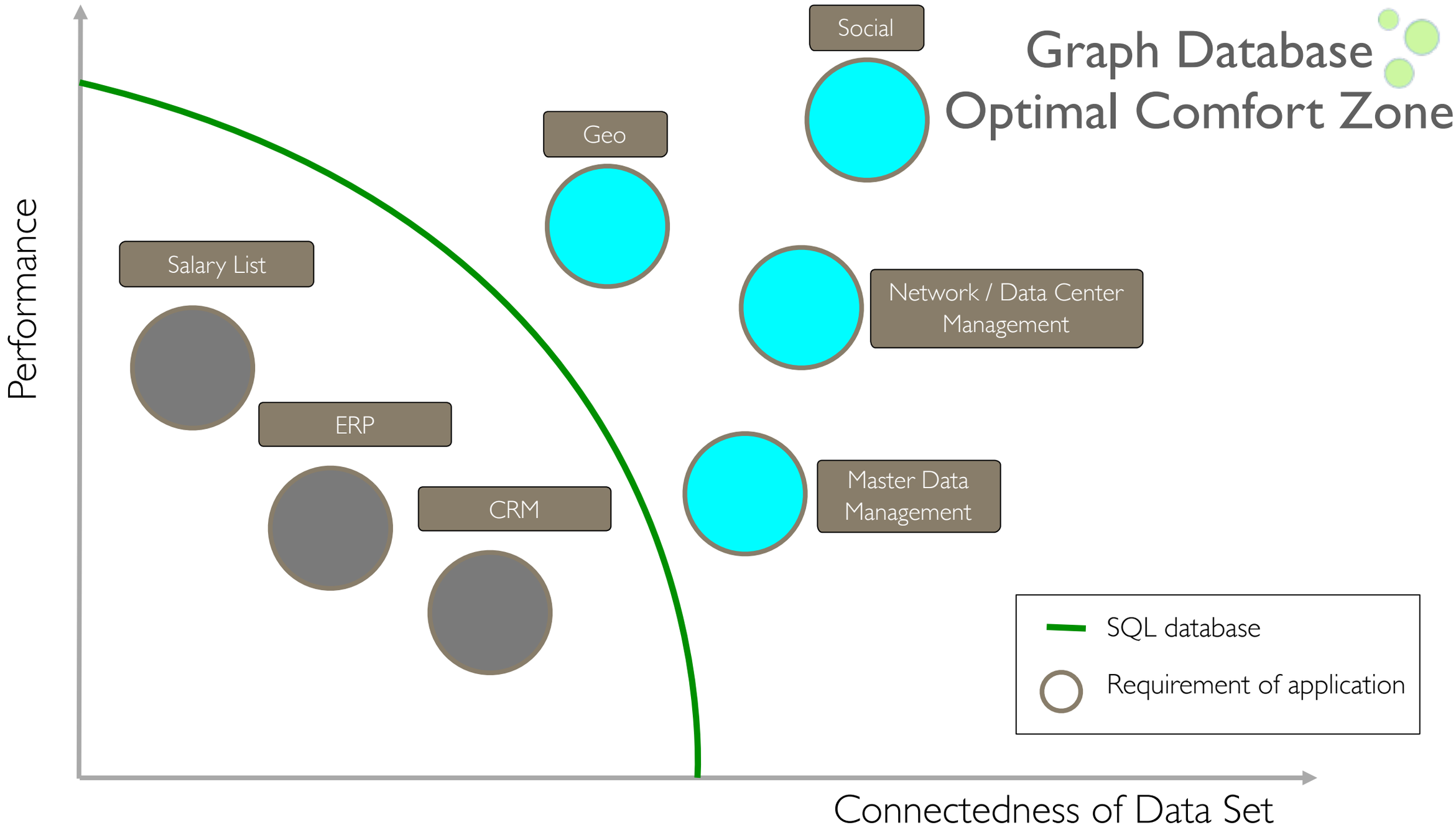
Graph db performance

- a sample social graph
  - with ~1,000 persons
- average 50 friends per person
- pathExists(a,b) limited to depth 4
- caches warmed up to eliminate disk I/O

Database	# persons	query time
MySQL	1,000	2,000 ms
Neo4j	1,000	2 ms
Neo4j	**1,000,000**	2 ms

*Additional Third Party Benchmark Available in *Neo4j in Action*: http://www.manning.com/partner/

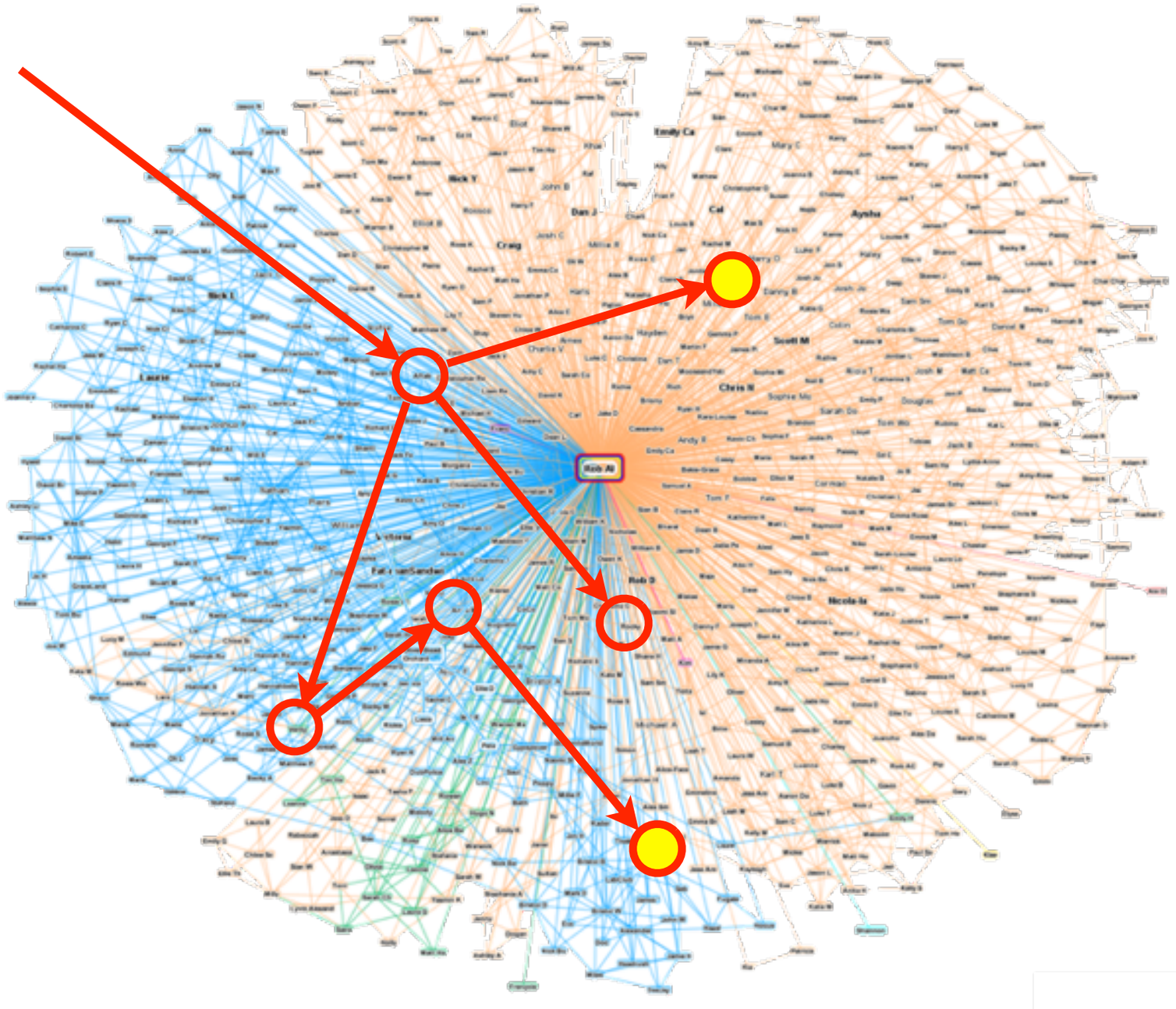
The Zone of SQL Adequacy



neo technology
graphs are everywhere

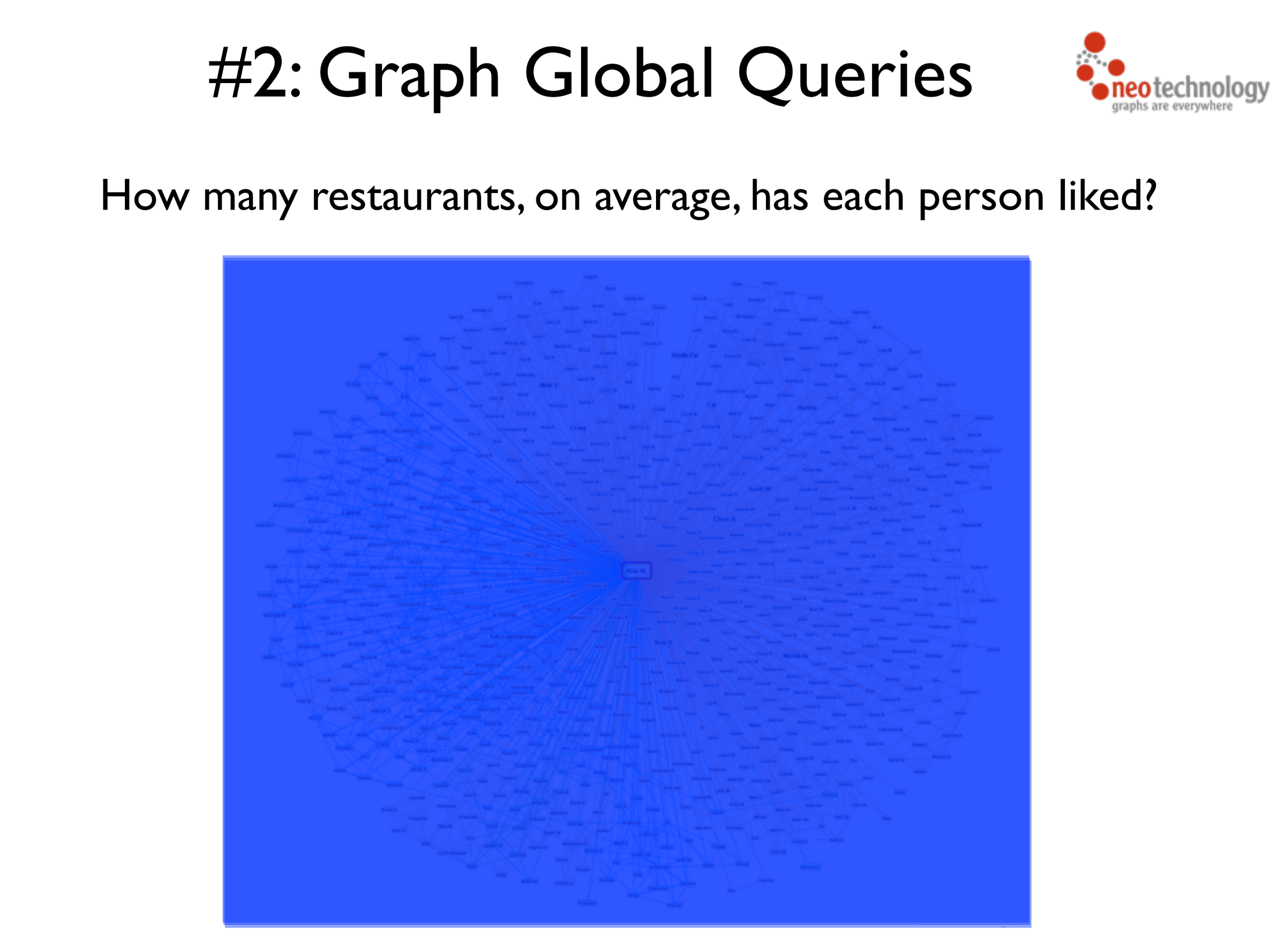
Graph Technology Ecosystem

#1: Graph Local Queries



e.g. Recommendations, Friend-of-Friend, Shortest Path

#2: Graph Global Queries



How many restaurants, on average, has each person liked?

What is a Graph Database

"A **graph database**... is an online database management system with CRUD methods that expose a graph data model"[1]

- Two important properties:
  - **Native graph storage** engine: *written from the ground up to manage graph data*
  - **Native graph processing**, including *index-free adjacency* to facilitate traversals

1] Robinson, Webber, Eifrem. *Graph Databases*. O'Reilly, 2013. p. 5. ISBN-10: 1449356265

Graph Databases are Designed to:

1. Store inter-connected data
2. Make it easy to make sense of that data
3. Enable extreme-performance operations for:
    - Discovery of connected data patterns
    - Relatedness queries > depth 1
    - Relatedness queries of arbitrary length
4. Make it easy to evolve the database

Top Reasons People Use Graph Databases

1. **Problems with Join performance.**
2. **Continuously evolving data set** (often involves wide and sparse tables)
3. The **Shape of the Domain** is naturally a graph
4. **Open-ended business requirements** necessitating fast, iterative development.

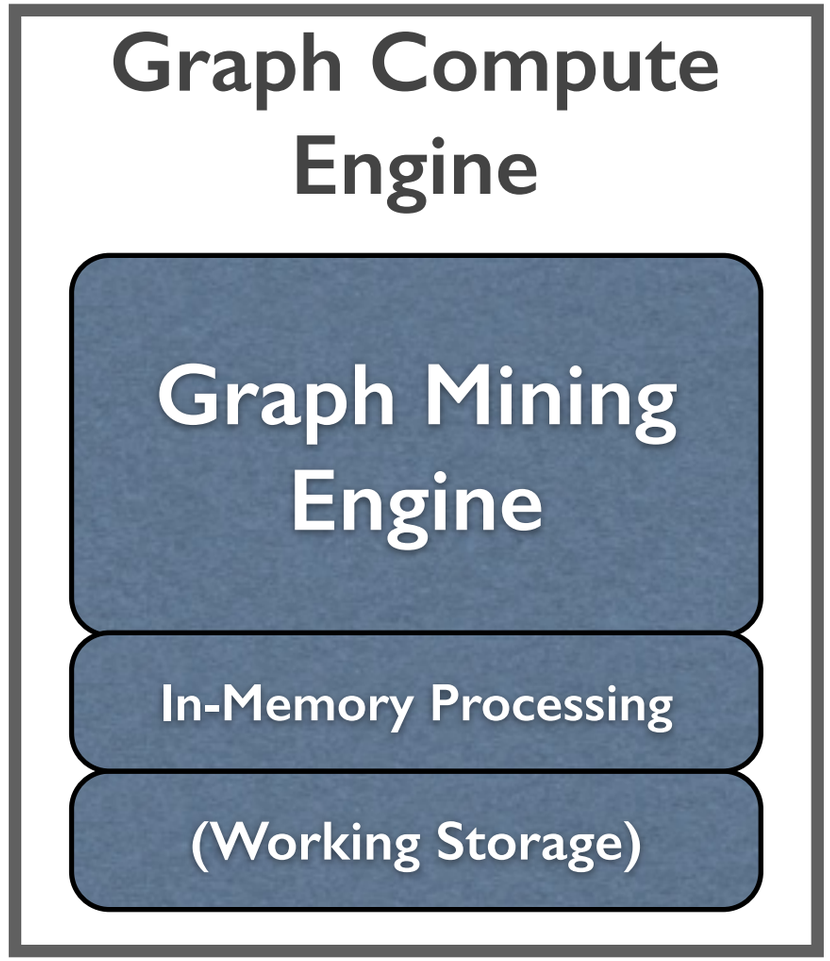
Graph Compute Engine

neotechnology
graphs are everywhere

Processing engine that enables graph global computational algorithms to be run against large data sets

System(s) of Record

Data extraction, transformation, and load

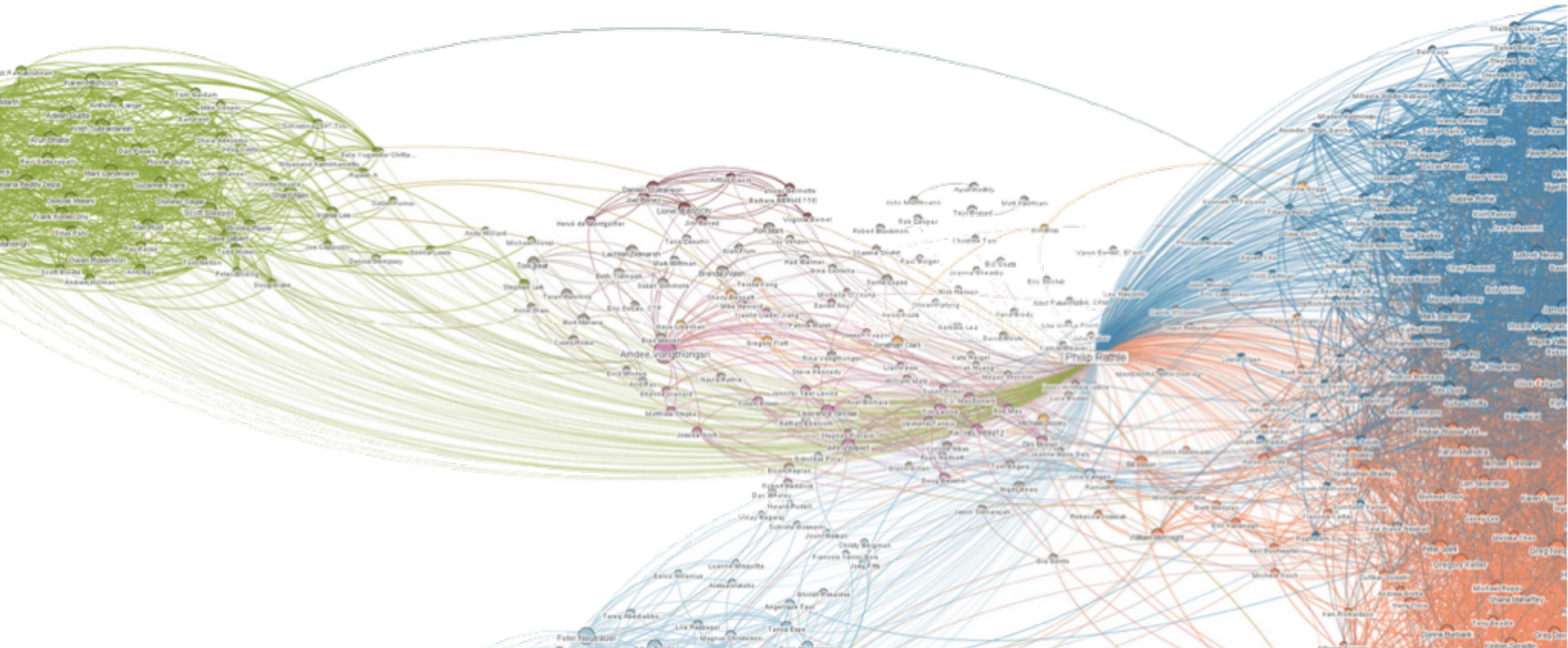


Connected Data

neotechnology
graphs are everywhere

Real-Time/ OLTP

Offline/ Batch



ABOUT THE VIADEO GROUP

The Viadeo Group forms the world's second largest professional social network with over 50 million members.

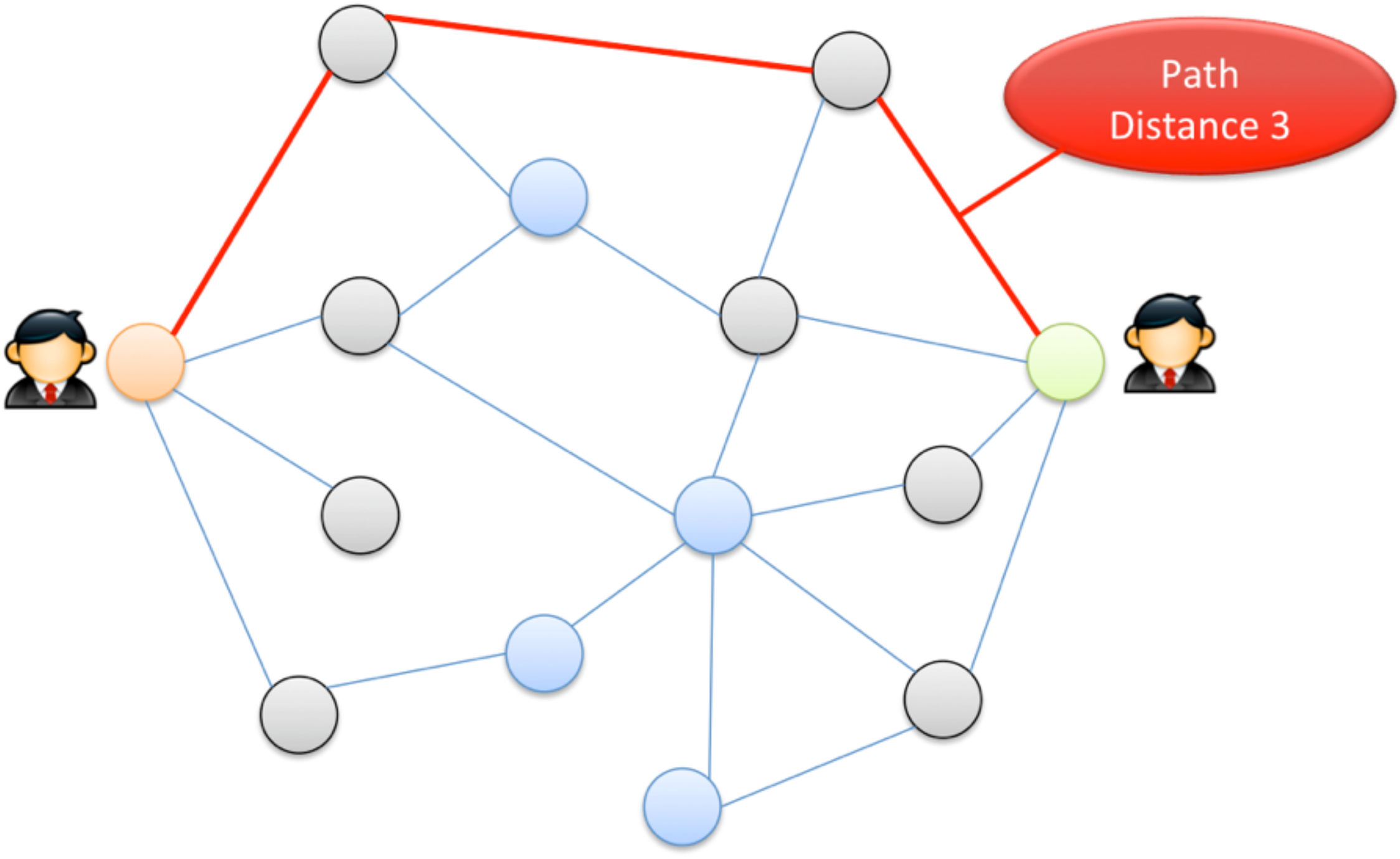
- N°1 in France with over 7 million members,
- N°1 in China with over 14 millions members.
- Strong presence in Europe and in emerging markets, strong growth in South America
- Presence in Russia since December 2011 through a joint venture with leading media group : Sanoma Independent Media
- April 2012 : record funding of 24 million euros (FSI, historical shareholders and new entrants)
- Headquarters in Paris and local offices on 5 continents
- 400 employees

- The Viadeo Group consists of:

viadeo ApnaCircle.com 天際網 www.tianji.com

viadeo

neotechnology
graphs are everywhere



viadeo

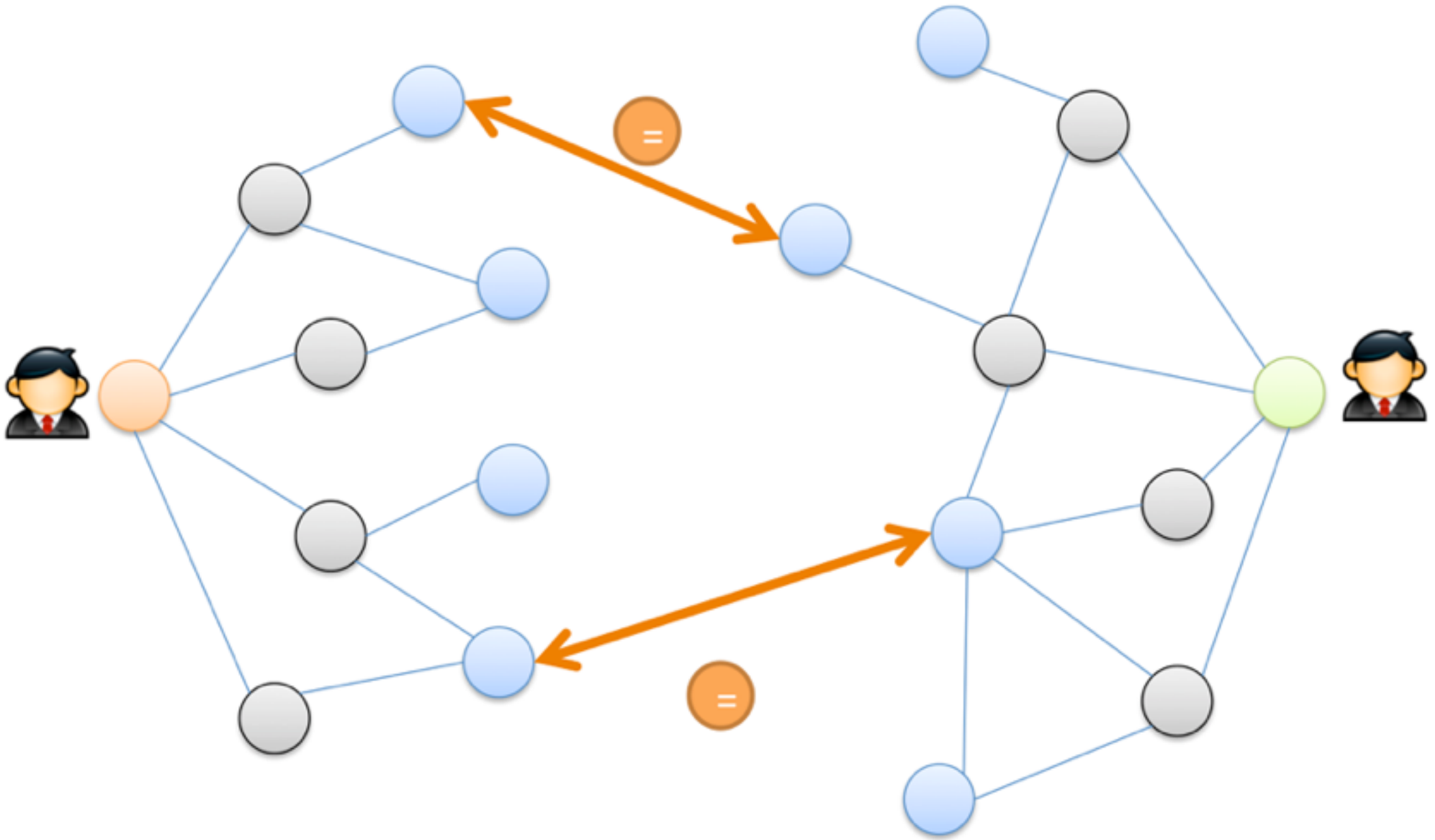
PREHISTORY 2006-2011

- In-house algorithm
- Network storage in **MySQL Database**

```
CREATE TABLE `Network` (
  `memberId` int(11) NOT NULL DEFAULT '0',
  `L1` mediumblob NOT NULL,
  `L2` mediumblob NOT NULL,
  PRIMARY KEY (`memberId`)
) ENGINE=InnoDB;
```

viadeo

PREHISTORY 2006-2011



viadeo

PREHISTORY 2006-2011

LIMITATIONS

1) Important latency for complete update

2) Massive bandwidth impact for internal network

3) 48 hours to restart from scratch

viadeo

PREHISTORY 2006-2011

Update the network (old-fashioned style)

Member A and Member B are now in contacts

- Update of A.L1 + B.L1 and A.L2 + B.L2
- Retrieving A.L1 + B.L1 and update *.L2

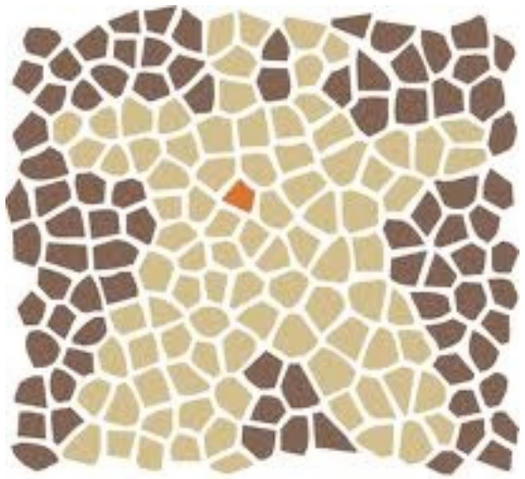
Example:

- A has 500 contacts
- B has 150 contacts

➔ 500 + 150 + 2 = **652 updates!**

GRAPH DATABASE

neotechnology
graphs are everywhere



APACHE
GIRAPH

Wait what?

New Users?
Real Time Updates?

viadeo

WHY Neo4J

BENEFITS

Instantaneous
graph updates

Very easy to integrate
(less than 2 months)

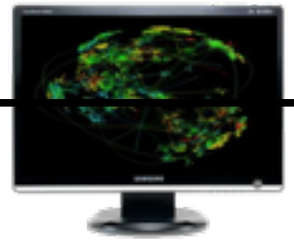
Backup /
Restore

High Availability

Graph Database Deployment

neotechnology
graphs are everywhere

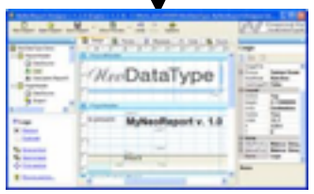
End User



Ad-hoc visual navigation & discovery

Graph Visualization

Graph-Dashboards & Ad-hoc Analysis



Reporting

Application

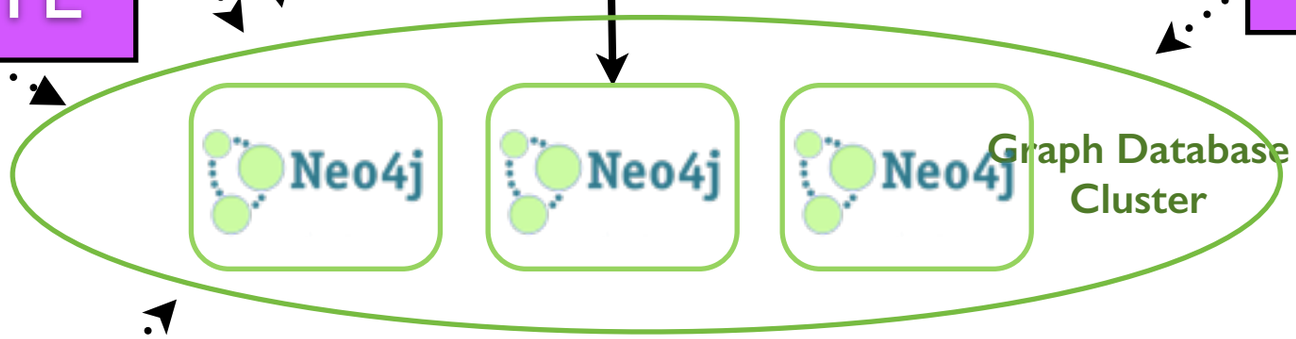
Other Databases

Bulk Analytic Infrastructure

(e.g. Graph Compute Engine)

ETL

ETL



Graph Database Cluster

Graph Mining & Aggregation

Ad-Hoc Analysis

Data Storage & Business Rules Execution

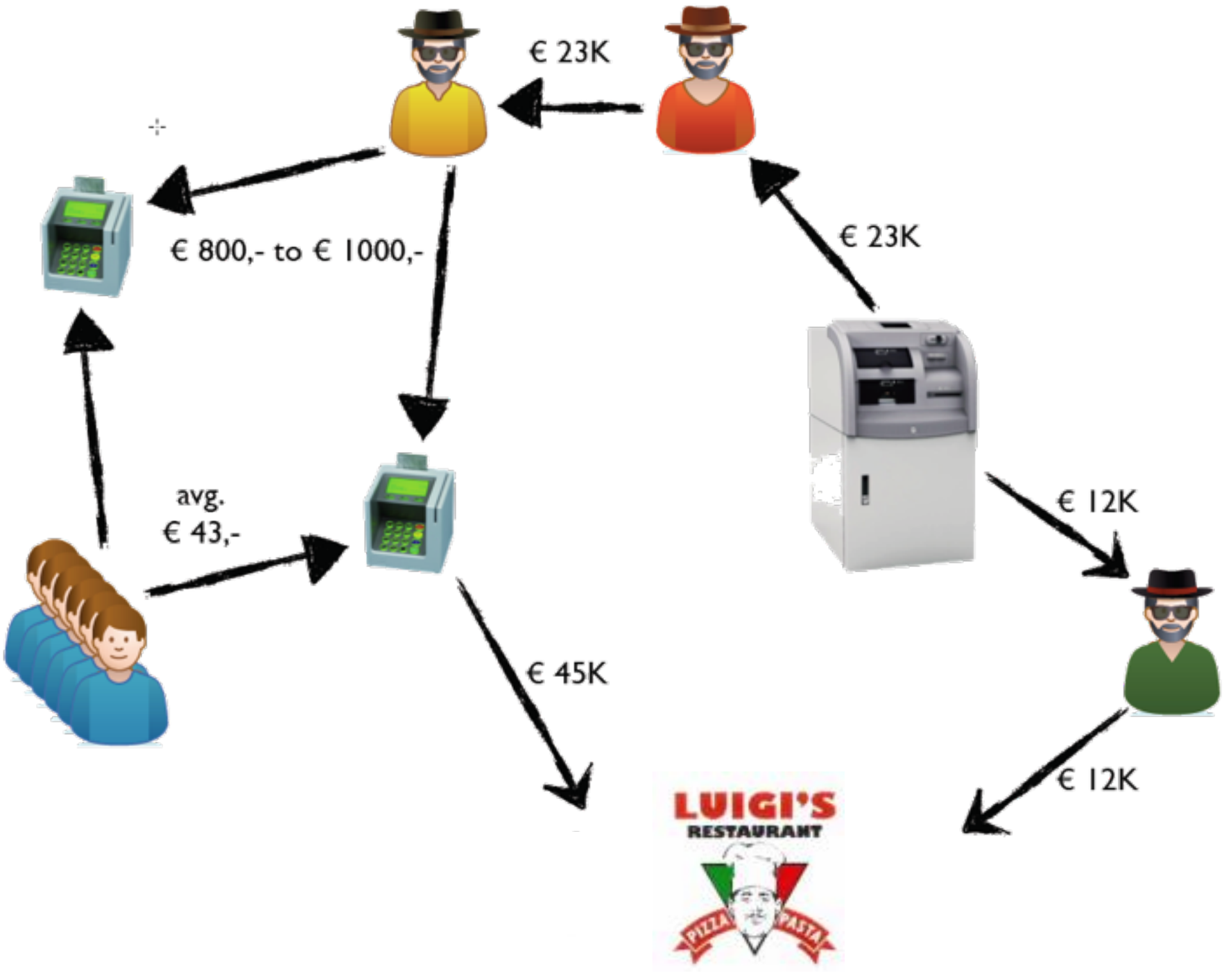
Data Scientist

neotechnology
graphs are everywhere

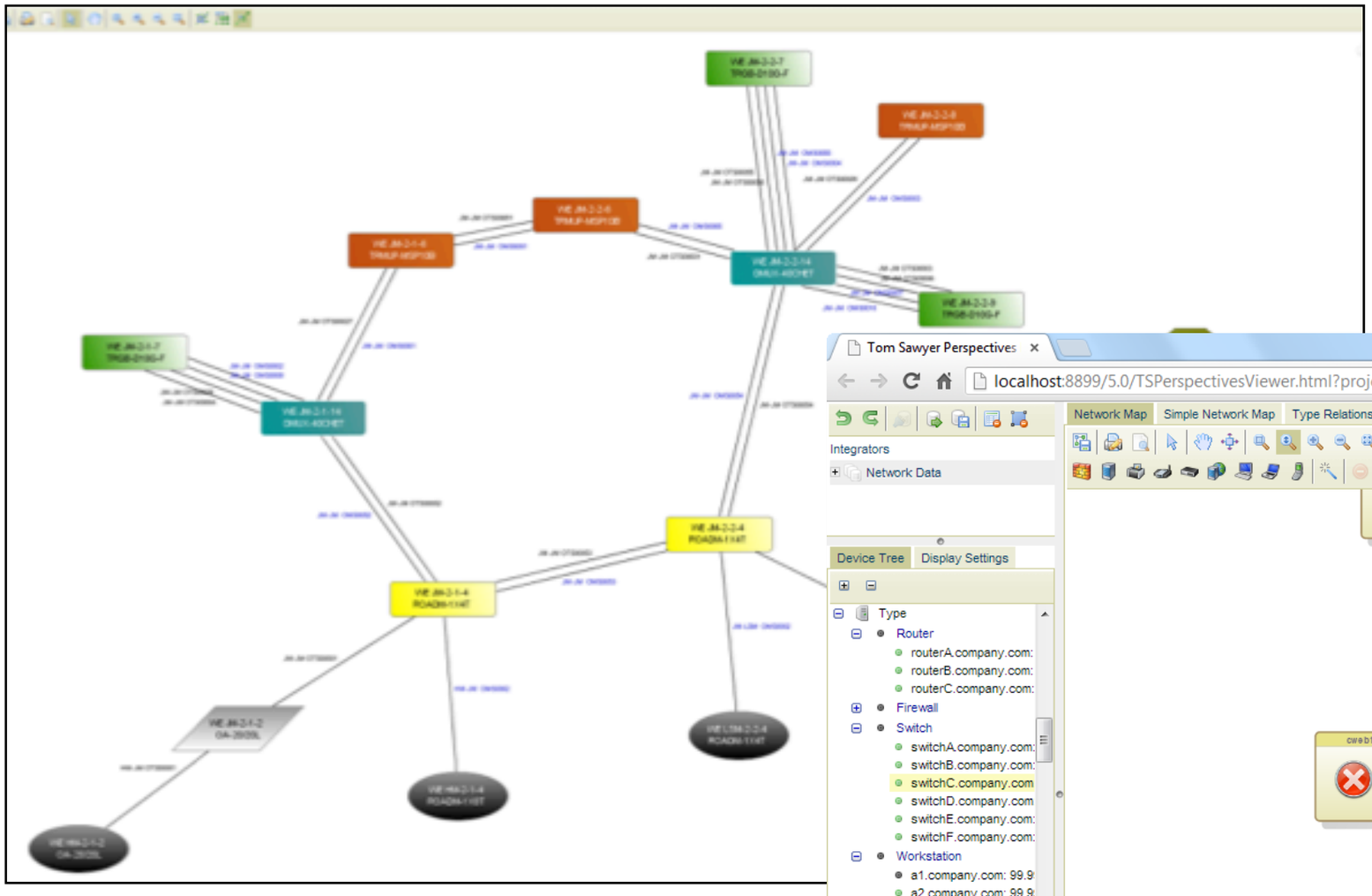
Graph Dashboards

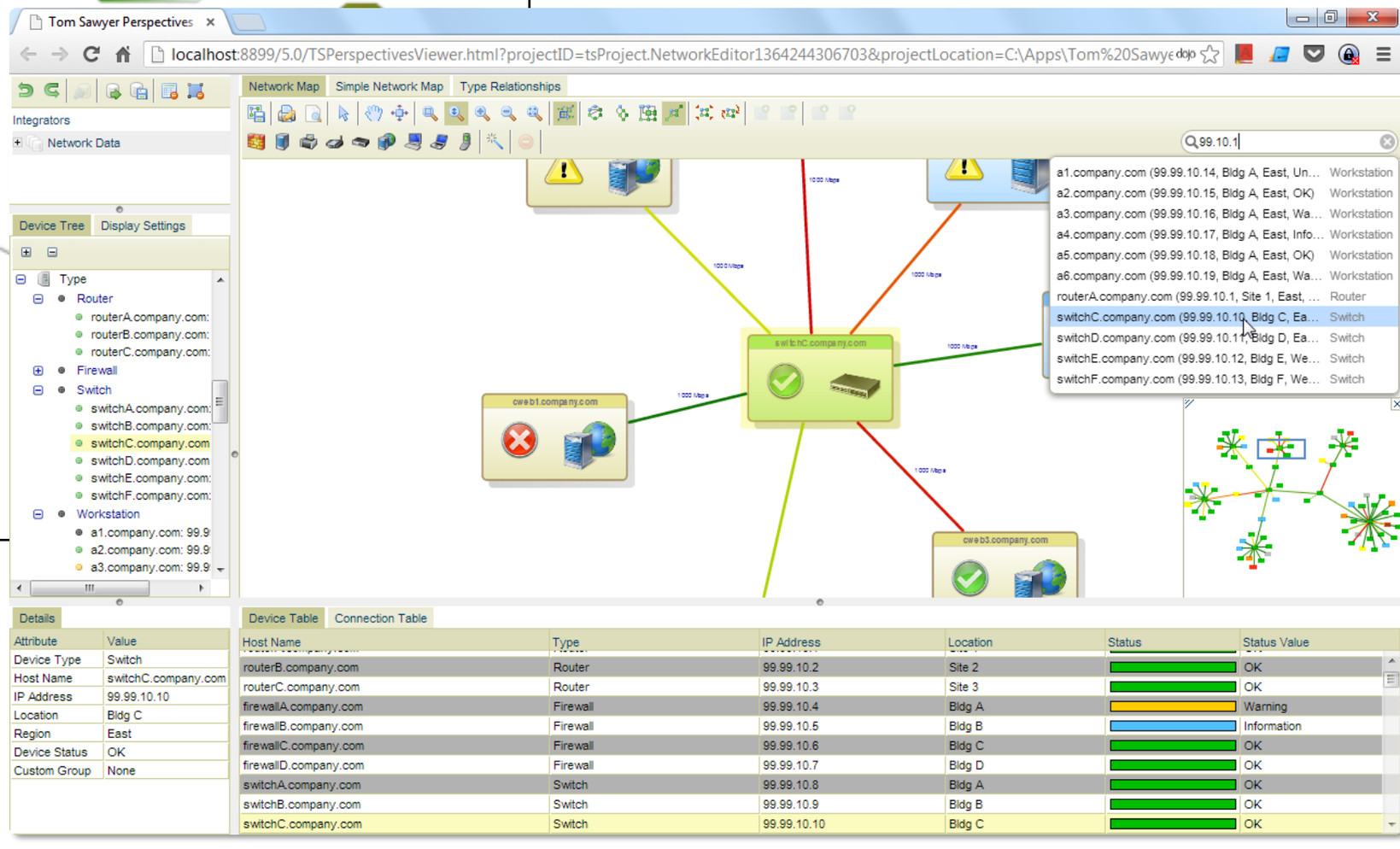
The Power of Visualization

Fraud Detection & Money Laundering



IT Service Dependencies

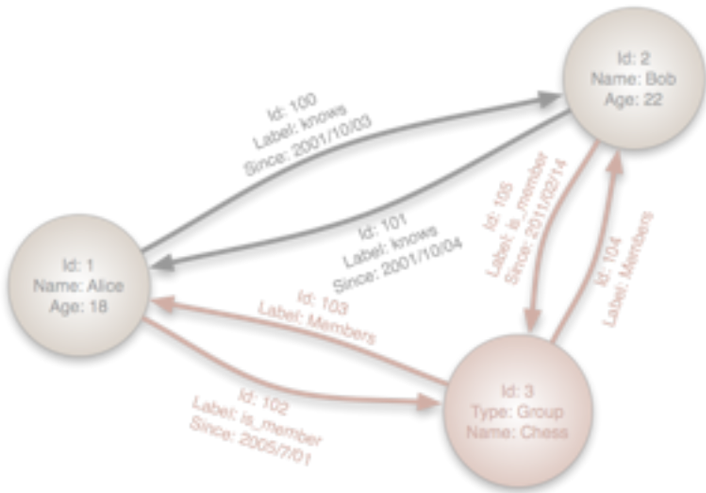




neo technology
graphs are everywhere

Working with Graphs

Case Studies & Working Examples



Cypher

ASCII art Graph Patterns

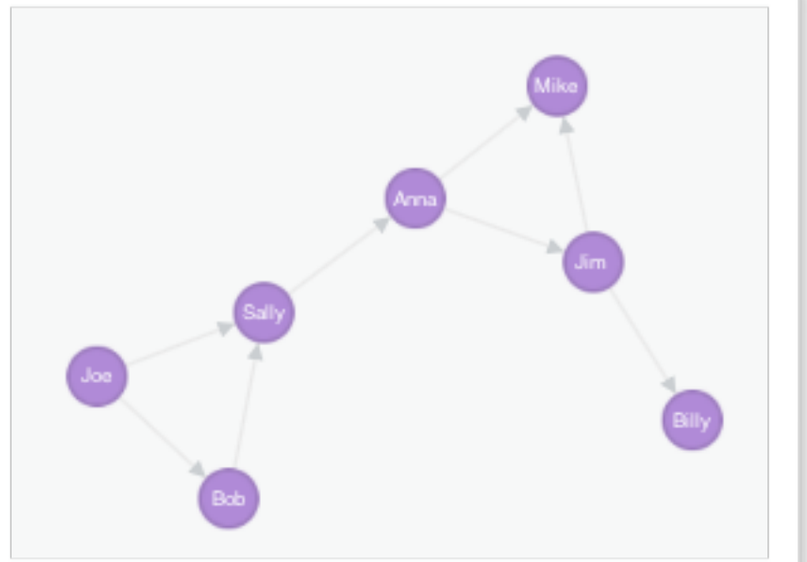

```
MATCH (A) -[:LOVES]-> (B)
  WHERE A.name = "A"
  RETURN B as lover
```

Social Example

Practical Cypher

Social Graph - Create

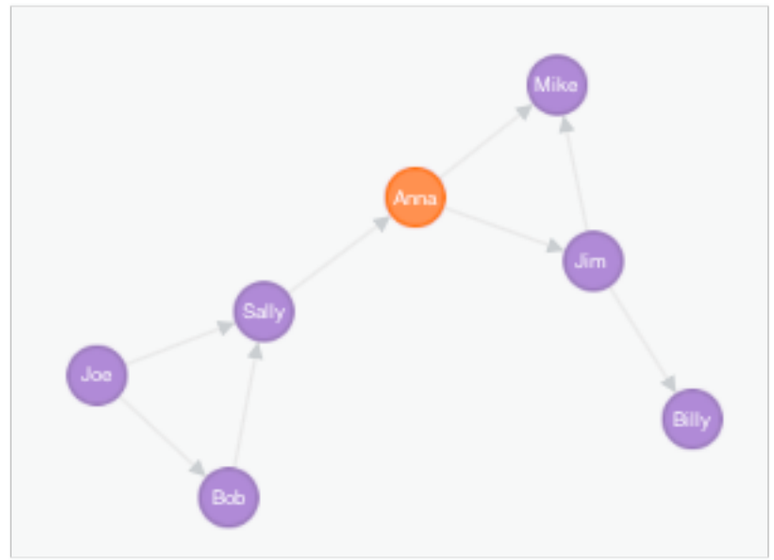
```
CREATE
    (joe:Person {name:"Joe"}),
    (bob:Person {name:"Bob"}),
    (sally:Person {name:"Sally"}),
    (anna:Person {name:"Anna"}),
    (jim:Person {name:"Jim"}),
    (mike:Person {name:"Mike"}),
    (billy:Person {name:"Billy"}),

    (joe)-[:KNOWS]->(bob),
    (joe)-[:KNOWS]->(sally),
    (bob)-[:KNOWS]->(sally),
    (sally)-[:KNOWS]->(anna),
    (anna)-[:KNOWS]->(jim),
    (anna)-[:KNOWS]->(mike),
    (jim)-[:KNOWS]->(mike),
    (jim)-[:KNOWS]->(billy)
```



Practical Cypher

Social Graph - Friends of Joe's Friends

```
MATCH (person)-[:KNOWS]-(friend),
      (friend)-[:KNOWS]-(foaf)
WHERE person.name = "Joe"
  AND NOT(person-[:KNOWS]-foaf)
RETURN foaf
```

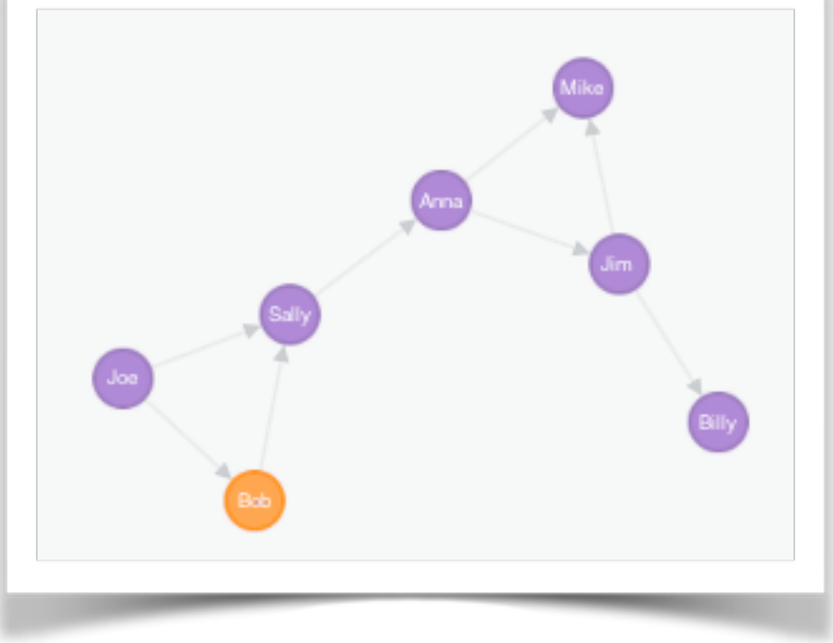


foaf
{name:"Anna"}

Practical Cypher

Social Graph - Common Friends

```
MATCH (person1)-[:KNOWS]-(friend),
  (person2)-[:KNOWS]-(friend)
WHERE person1.name = "Joe"
  AND person2.name = "Sally"
RETURN friend
```

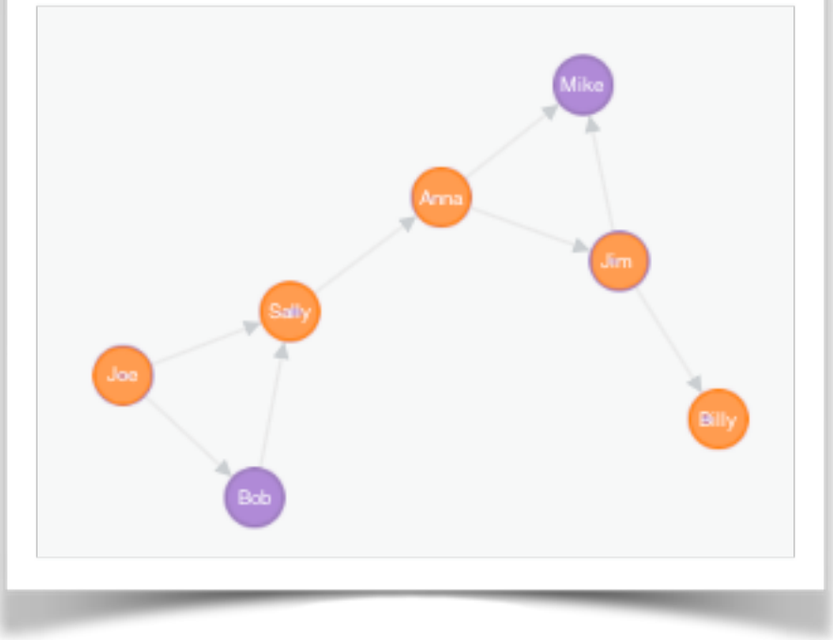


friend
{name:"Bob"}

Practical Cypher

Social Graph - Shortest Path

```
MATCH path = shortestPath(
  (person1)-[:KNOWS*..6]-(person2)
)
WHERE person1.name = "Joe"
   AND person2.name = "Billy"
RETURN path
```



path
{start:"13759", nodes:["13759","13757","13756","13755","13753"], length:4, relationships:["101407","101409","101410","101413"], end:"13753"}

Network Management Example

Practical Cypher

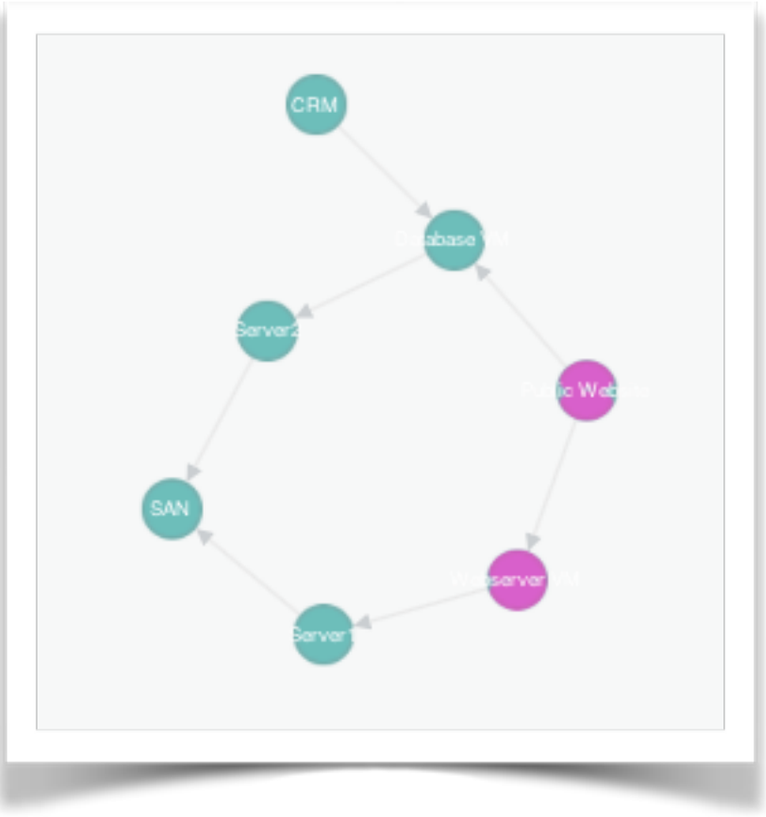
Network Management - Create

```
CREATE
    (crm {name:"CRM"}),
    (dbvm {name:"Database VM"}),
    (www {name:"Public Website"}),
    (wwwvm {name:"Webserver VM"}),
    (srv1 {name:"Server 1"}),
    (san {name:"SAN"}),
    (srv2 {name:"Server 2"}),

    (crm)-[:DEPENDS_ON]->(dbvm),
    (dbvm)-[:DEPENDS_ON]->(srv2),
    (srv2)-[:DEPENDS_ON]->(san),
    (www)-[:DEPENDS_ON]->(dbvm),
    (www)-[:DEPENDS_ON]->(wwwvm),
    (wwwvm)-[:DEPENDS_ON]->(srv1),
    (srv1)-[:DEPENDS_ON]->(san)
```

Practical Cypher

Network Management - Impact Analysis

```
// Server 1 Outage
MATCH (n)<-[:DEPENDS_ON*]-(upstream)
WHERE n.name = "Server 1"
RETURN upstream
```

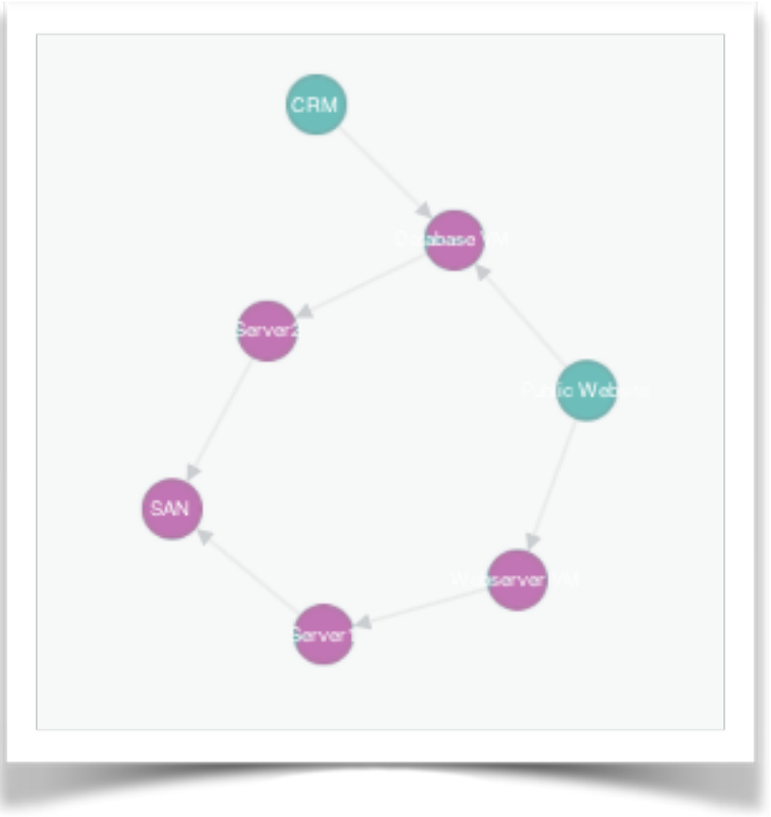


upstream
{name:"Webserver VM"}
{name:"Public Website"}

Practical Cypher

Network Management - Dependency Analysis

```
// Public website dependencies
MATCH (n)-[:DEPENDS_ON*]->(downstream)
WHERE n.name = "Public Website"
RETURN downstream
```



downstream
{name:"Database VM"}
{name:"Server 2"}
{name:"SAN"}
{name:"Webserver VM"}
{name:"Server 1"}

Practical Cypher

Network Management - Statistics

```
// Most depended on component
MATCH (n)<-[:DEPENDS_ON*]-(dependent)
RETURN n,
  count(DISTINCT dependent)
    AS dependents
ORDER BY dependents DESC
LIMIT 1
```

n	dependents
{name:"SAN"}	6

neo technology
graphs are everywhere

Questions ?