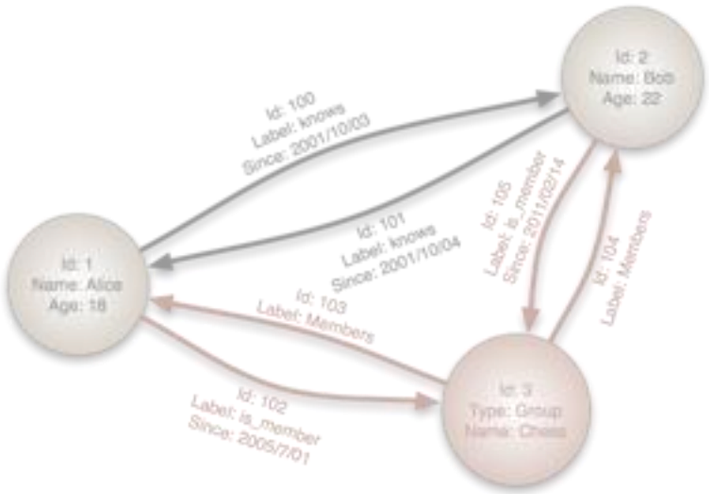
neo technology
graphs are everywhere

Intro to Neo4j

JUG Darmstadt
November 2015

Dirk Mahler
dirk.mahler@buschmais.com
@dirkmahler



Agenda

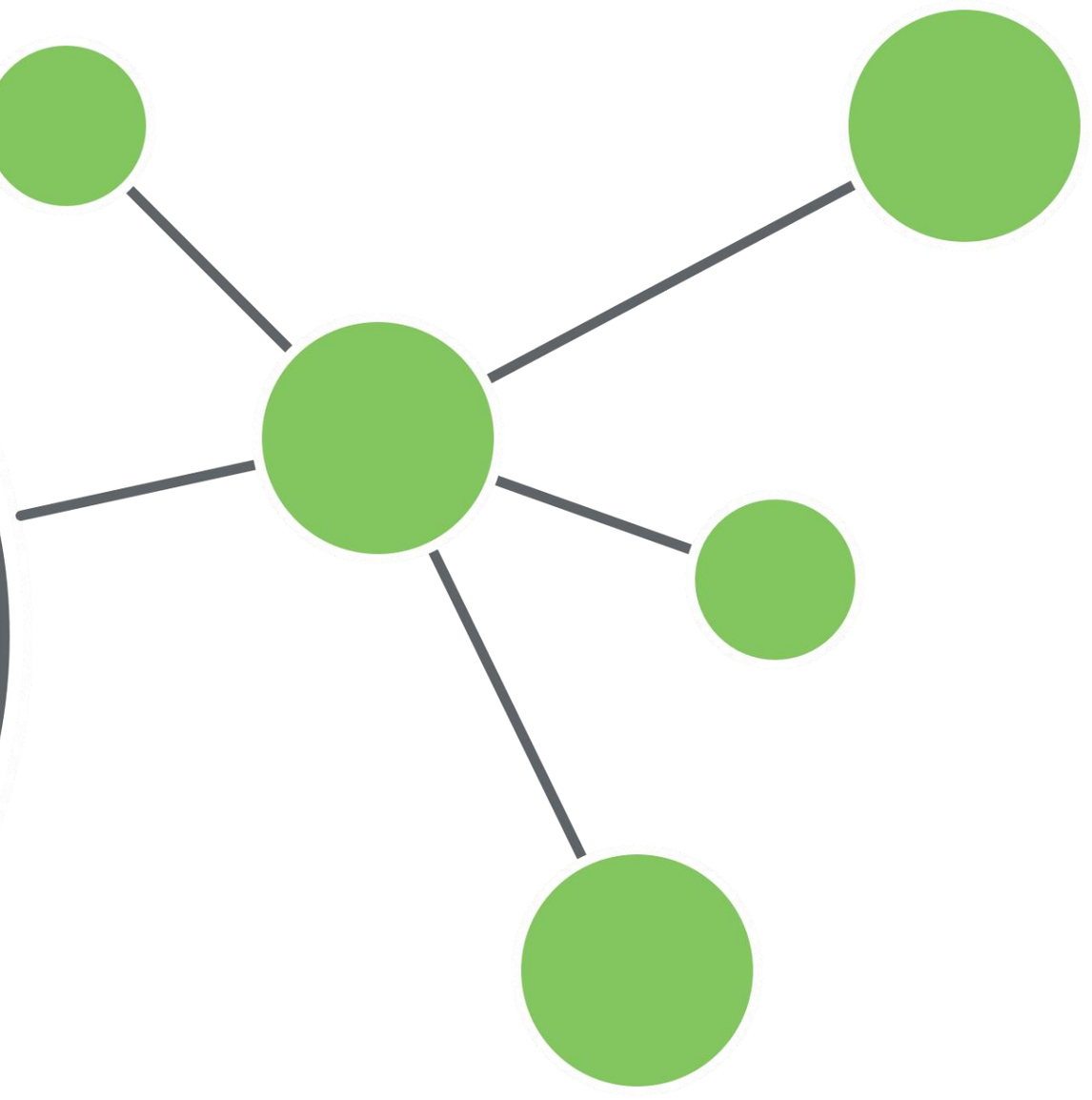
1. Why Graphs, Why Now?
2. What Is A Graph, Anyway?
3. Neo4j as a Graph Database
4. Graph Querying
    1. Cypher
    2. Examples

Why Graphs?

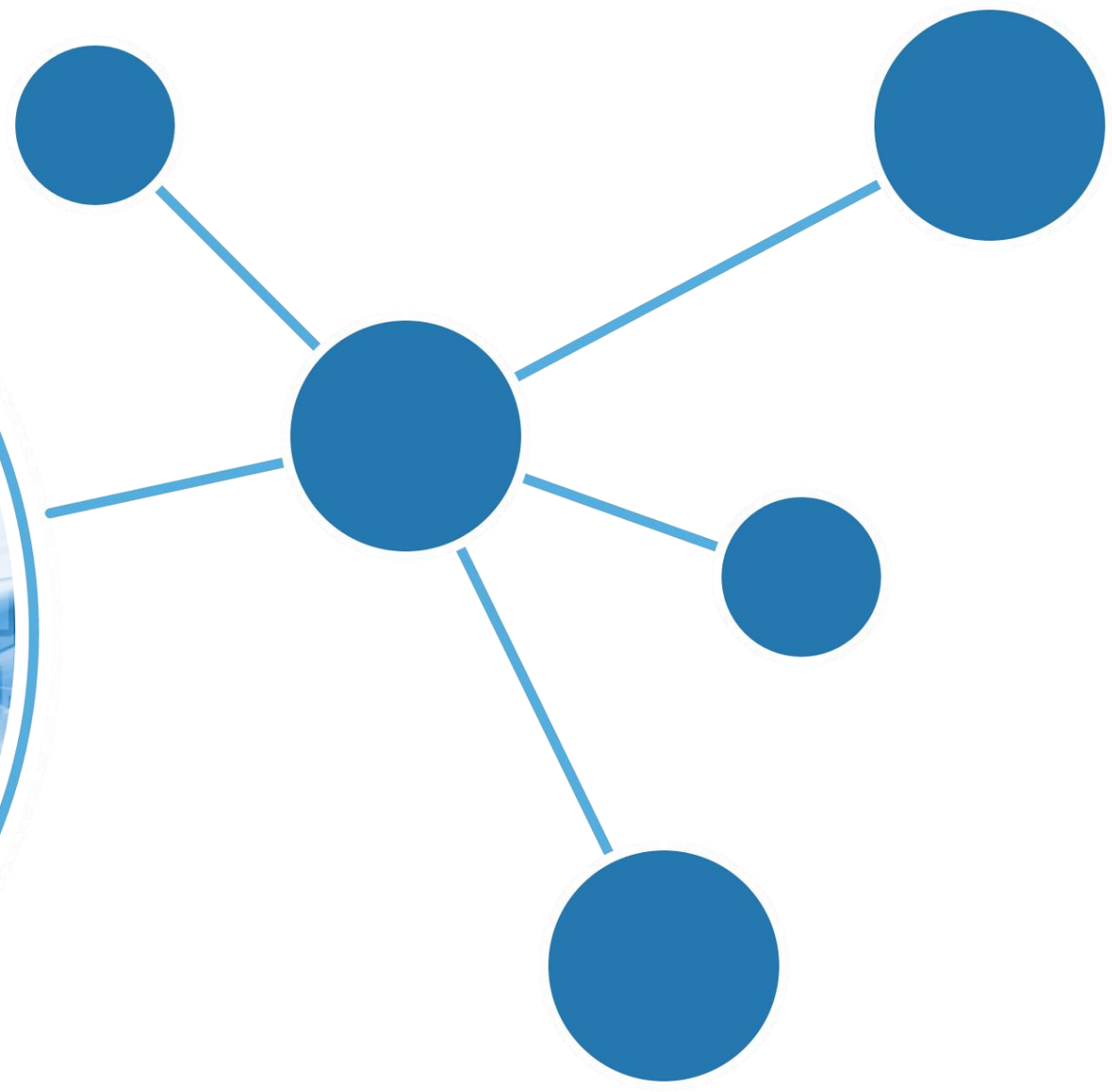
The World is a Graph

Some Use-Cases

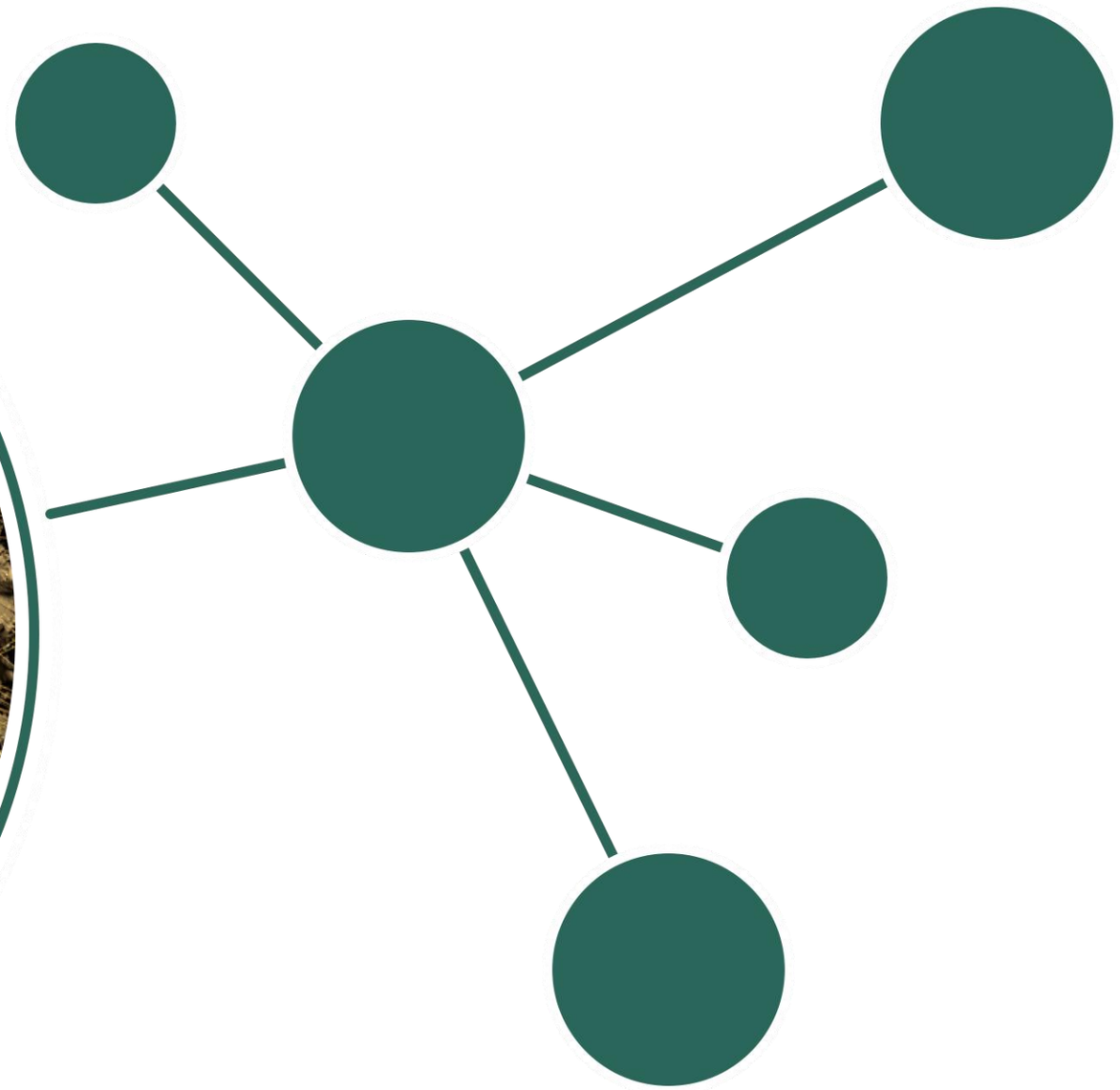
Social Network



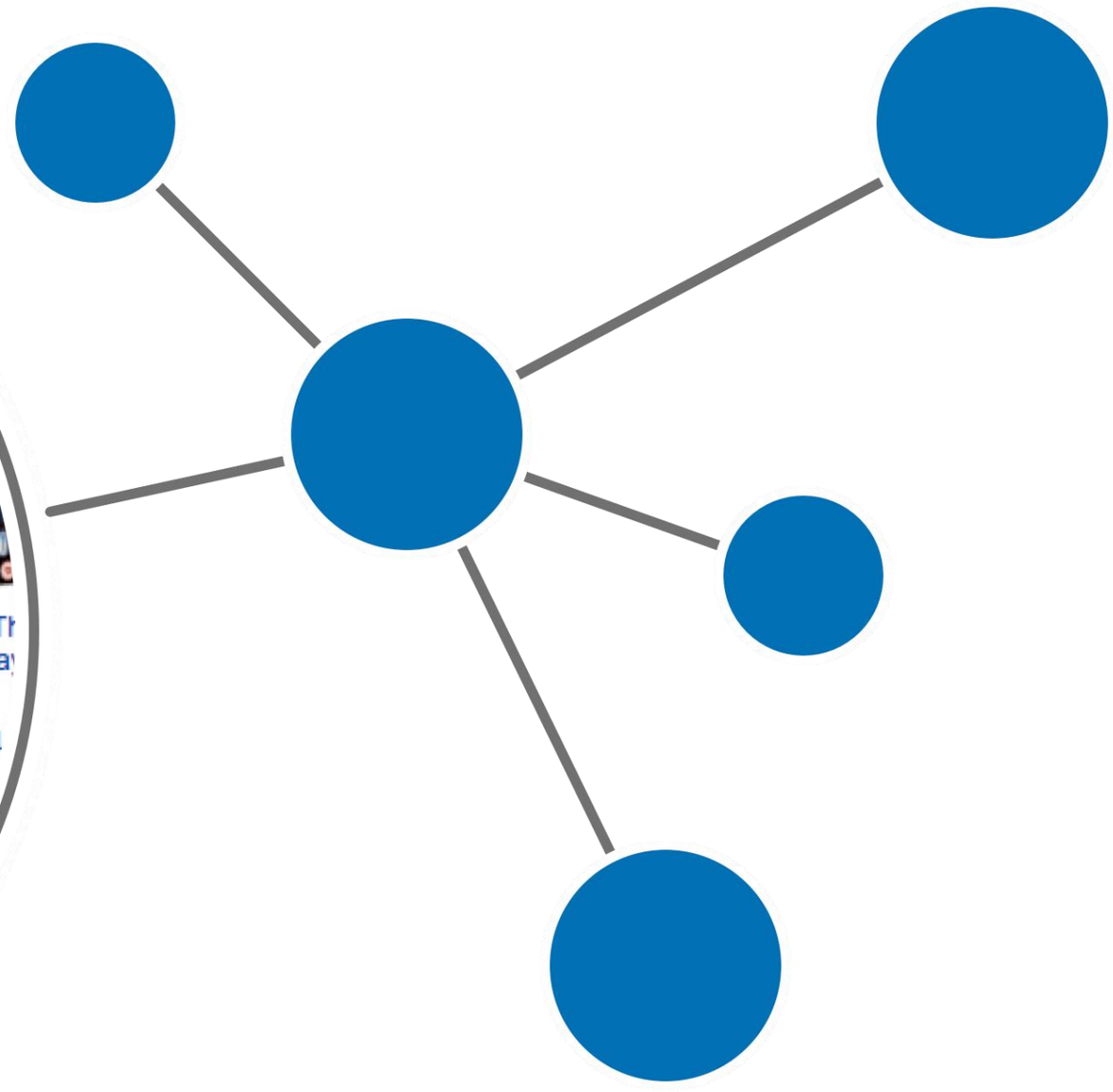
(Network) Impact Analysis



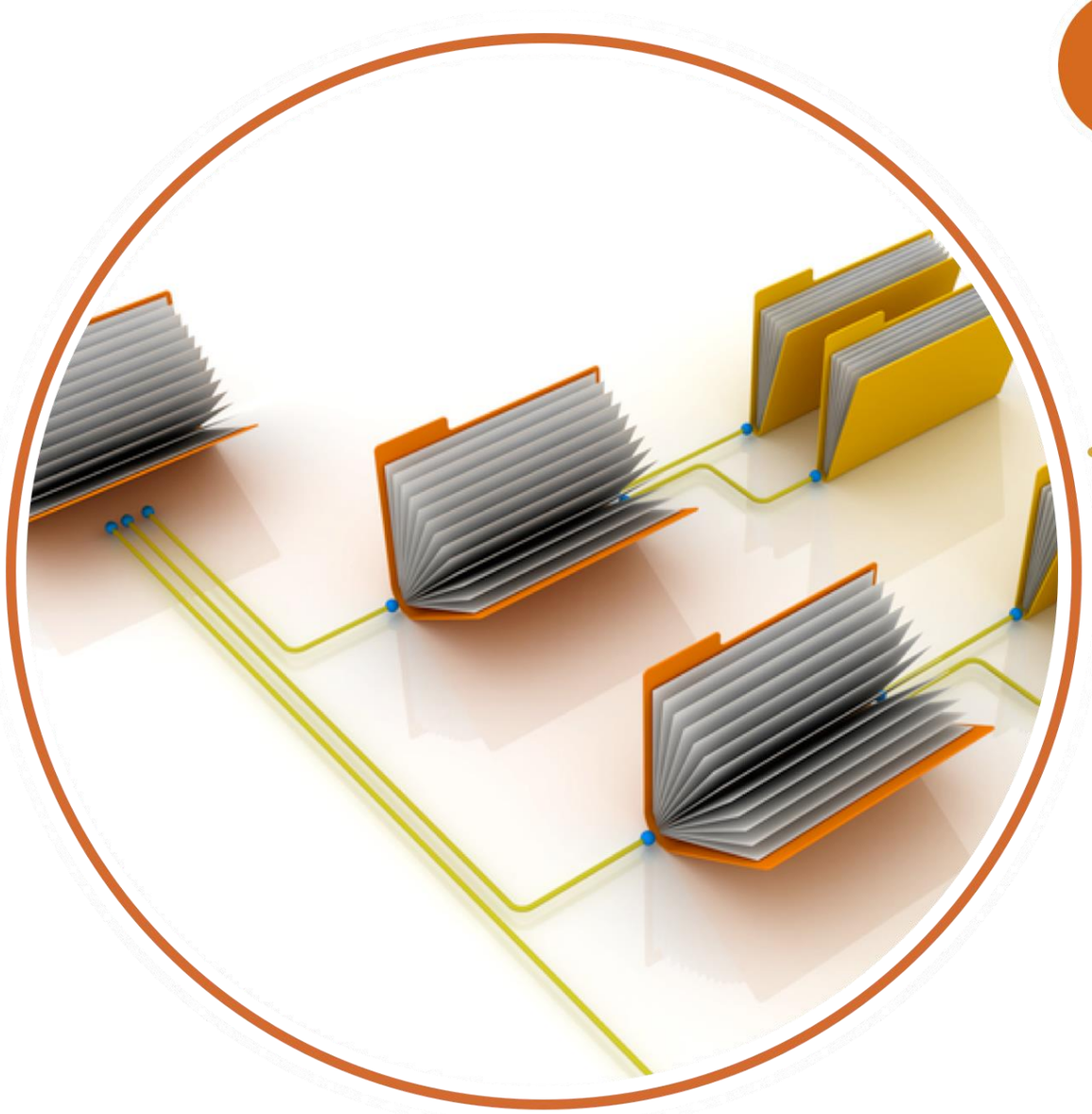
Logistics & Routing

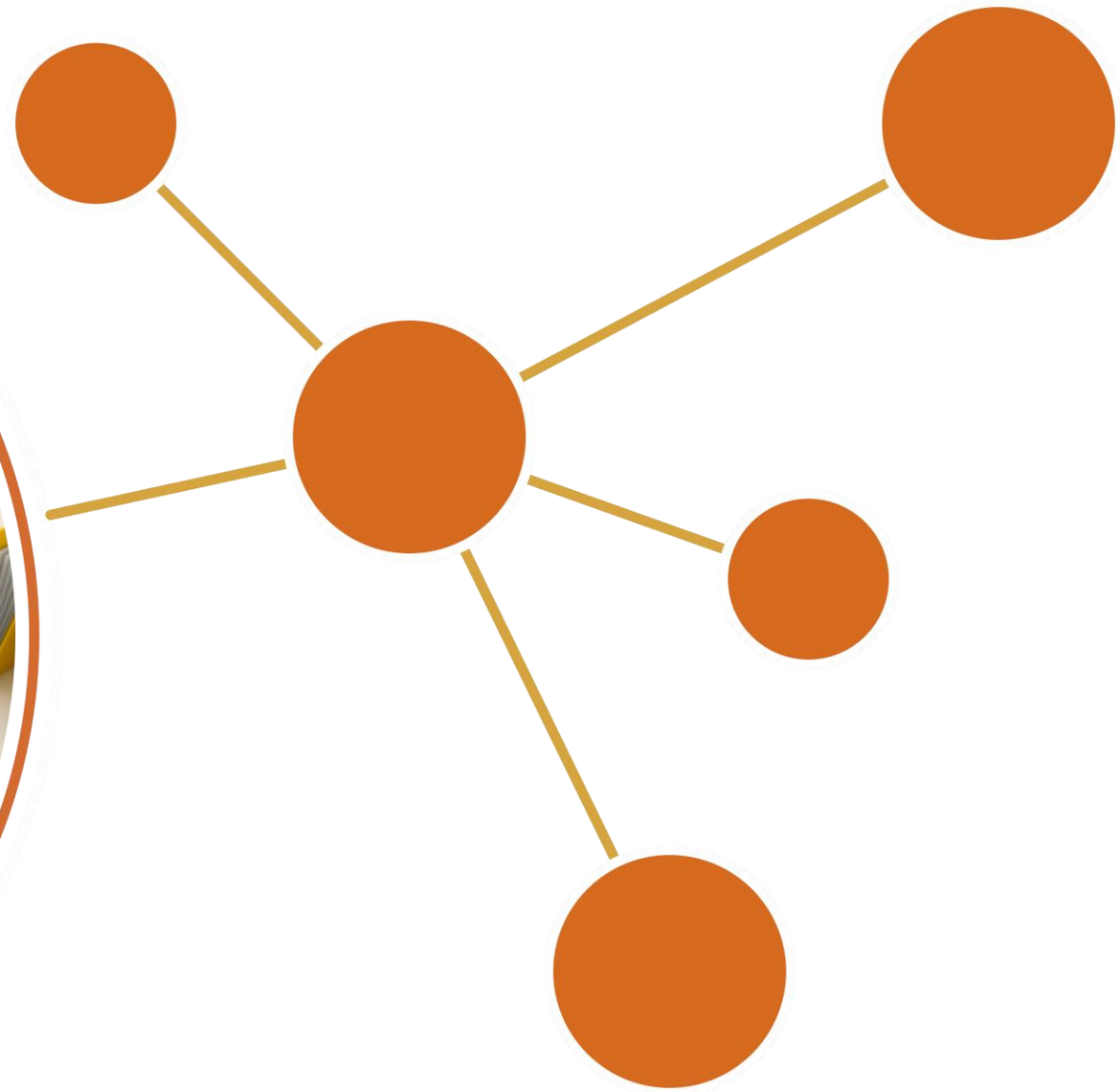


Recommendations

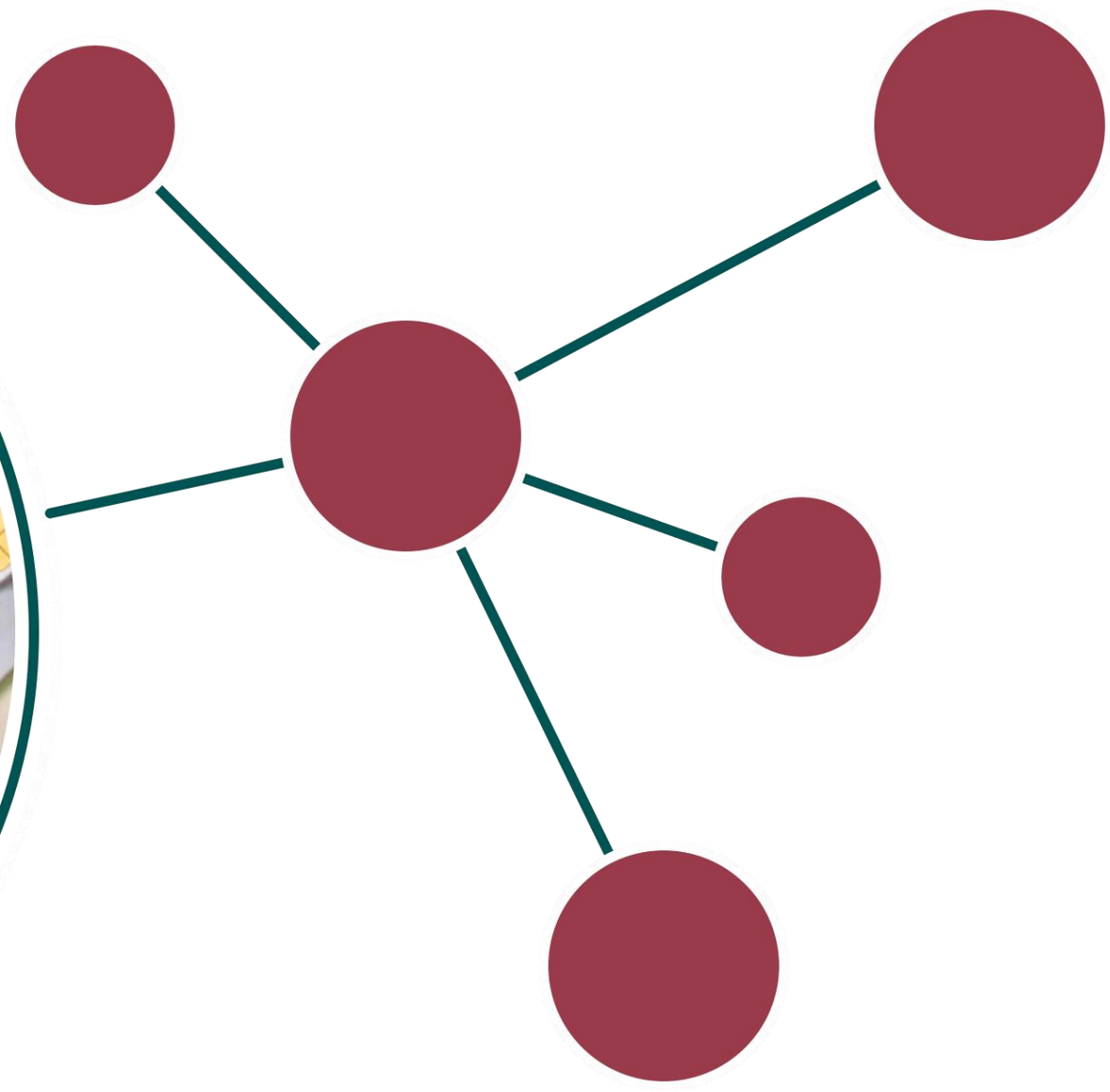


Access Control



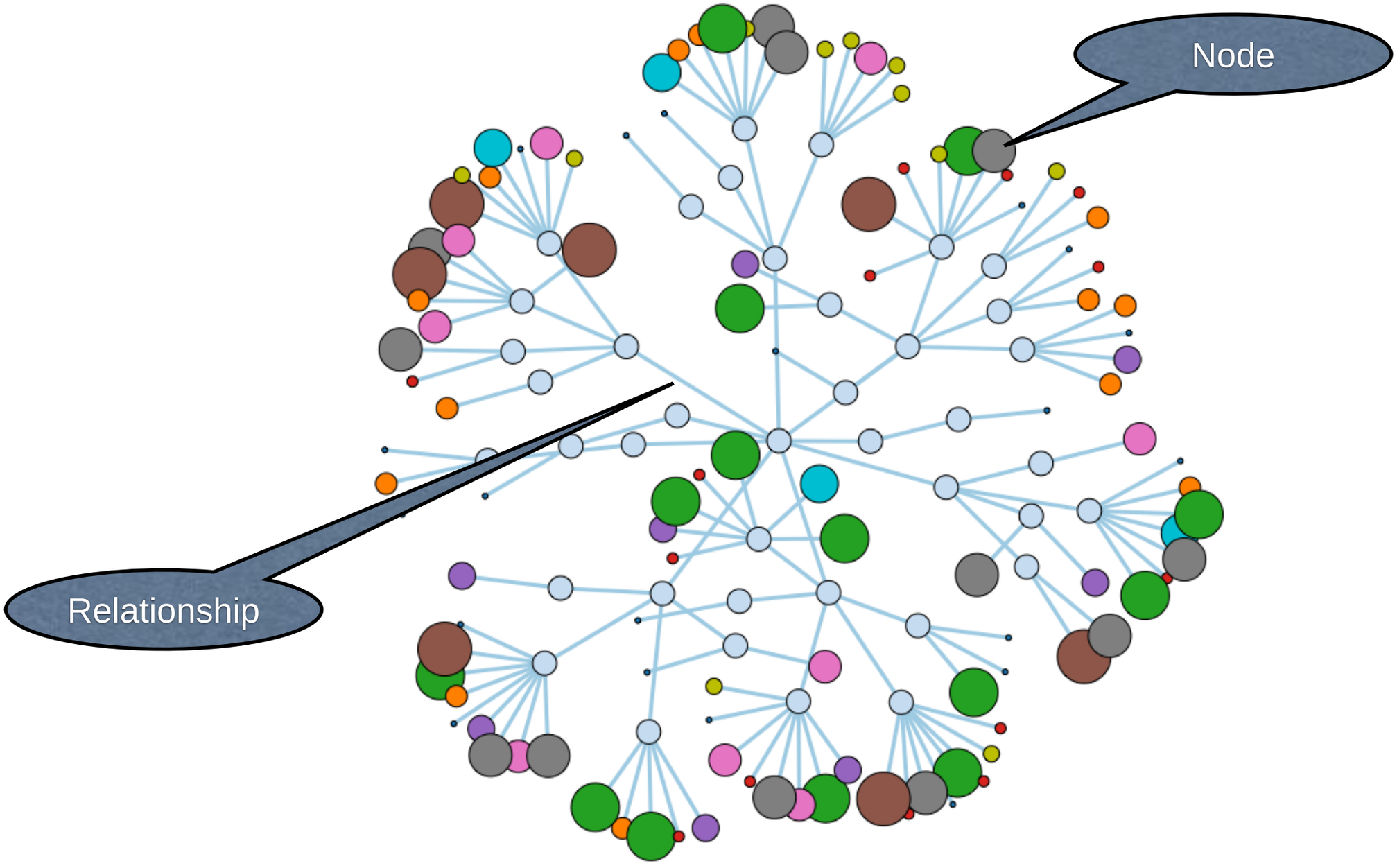


Fraud Analysis



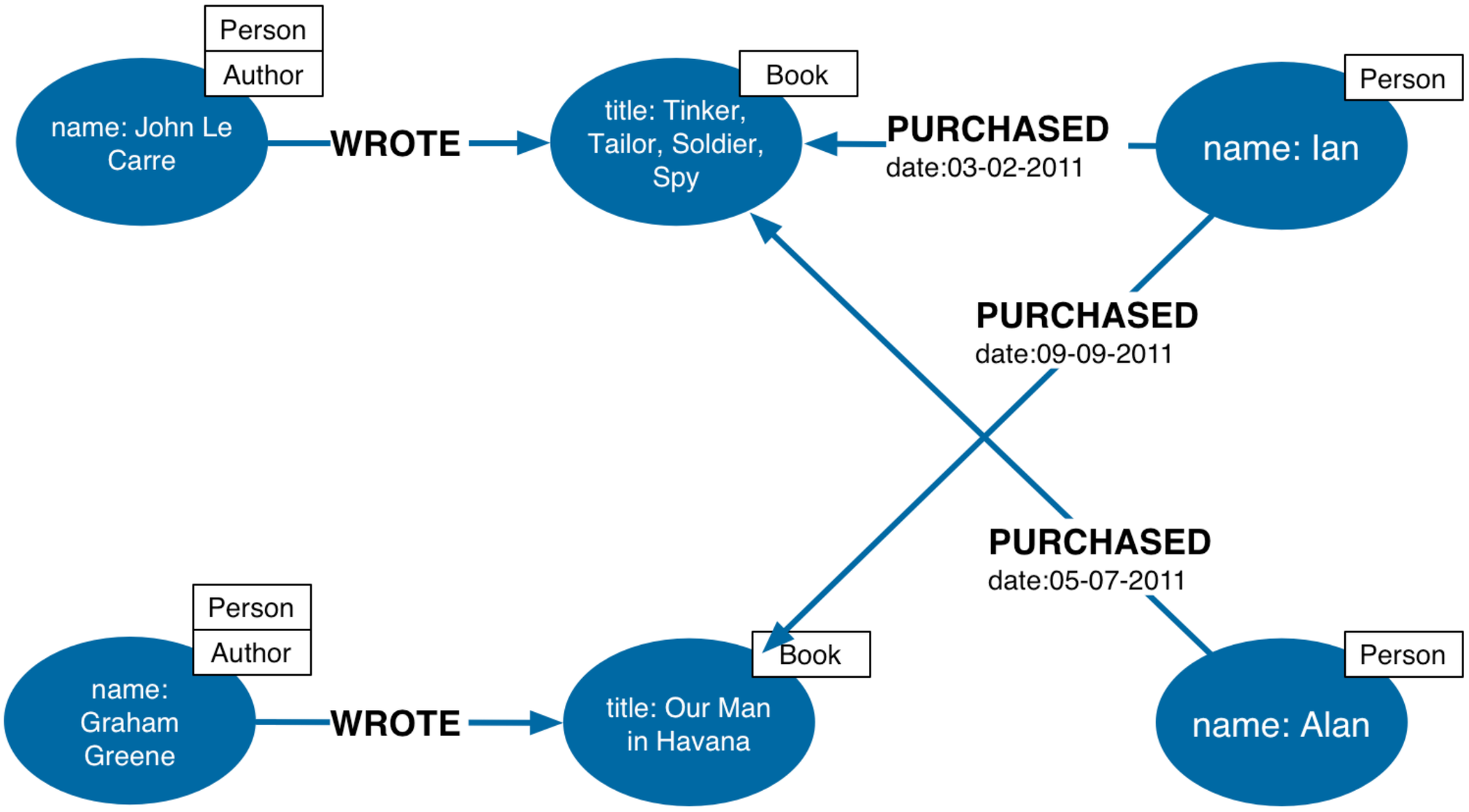
What Is A Graph, Anyway?

A Graph

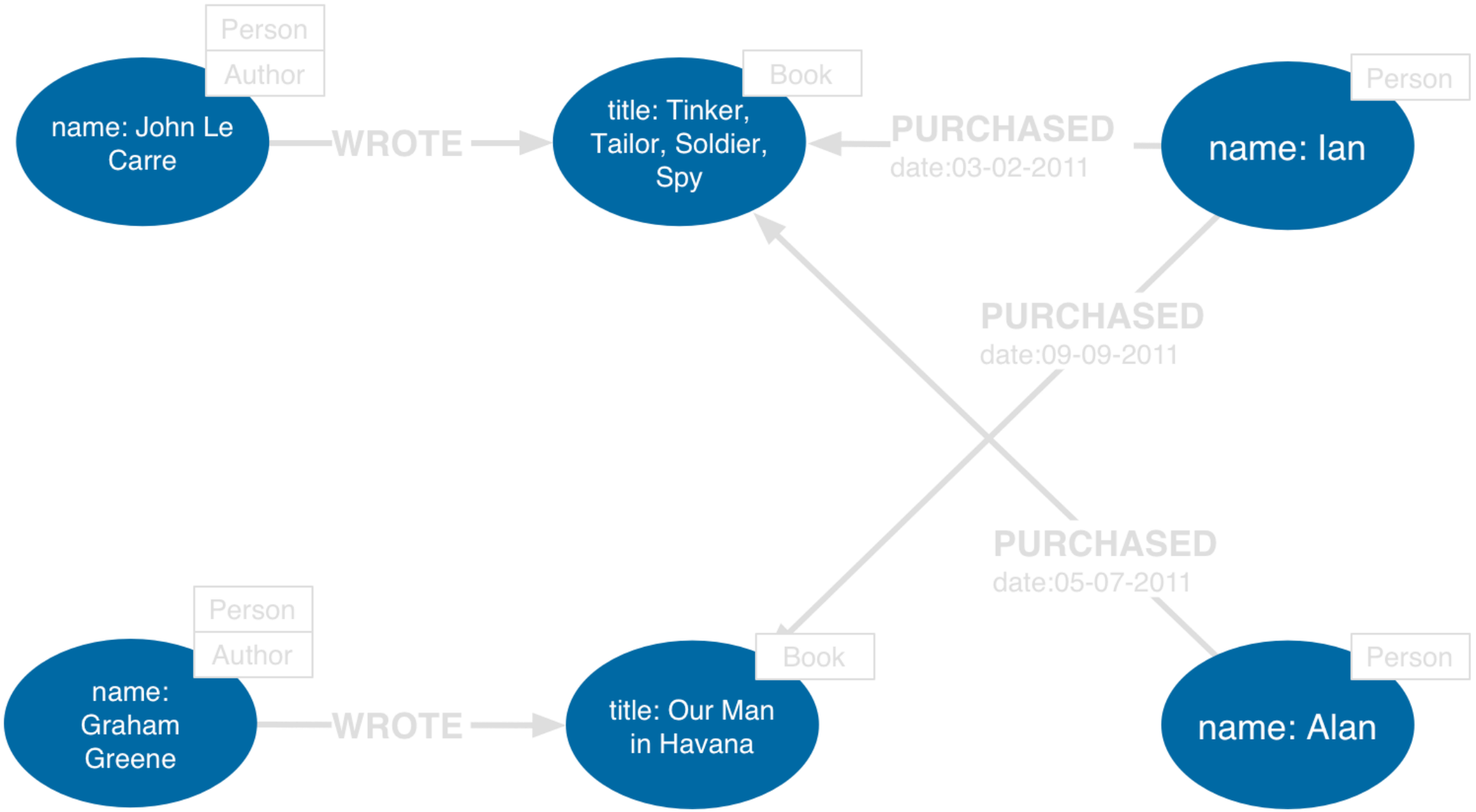


Four Graph Model Building Blocks

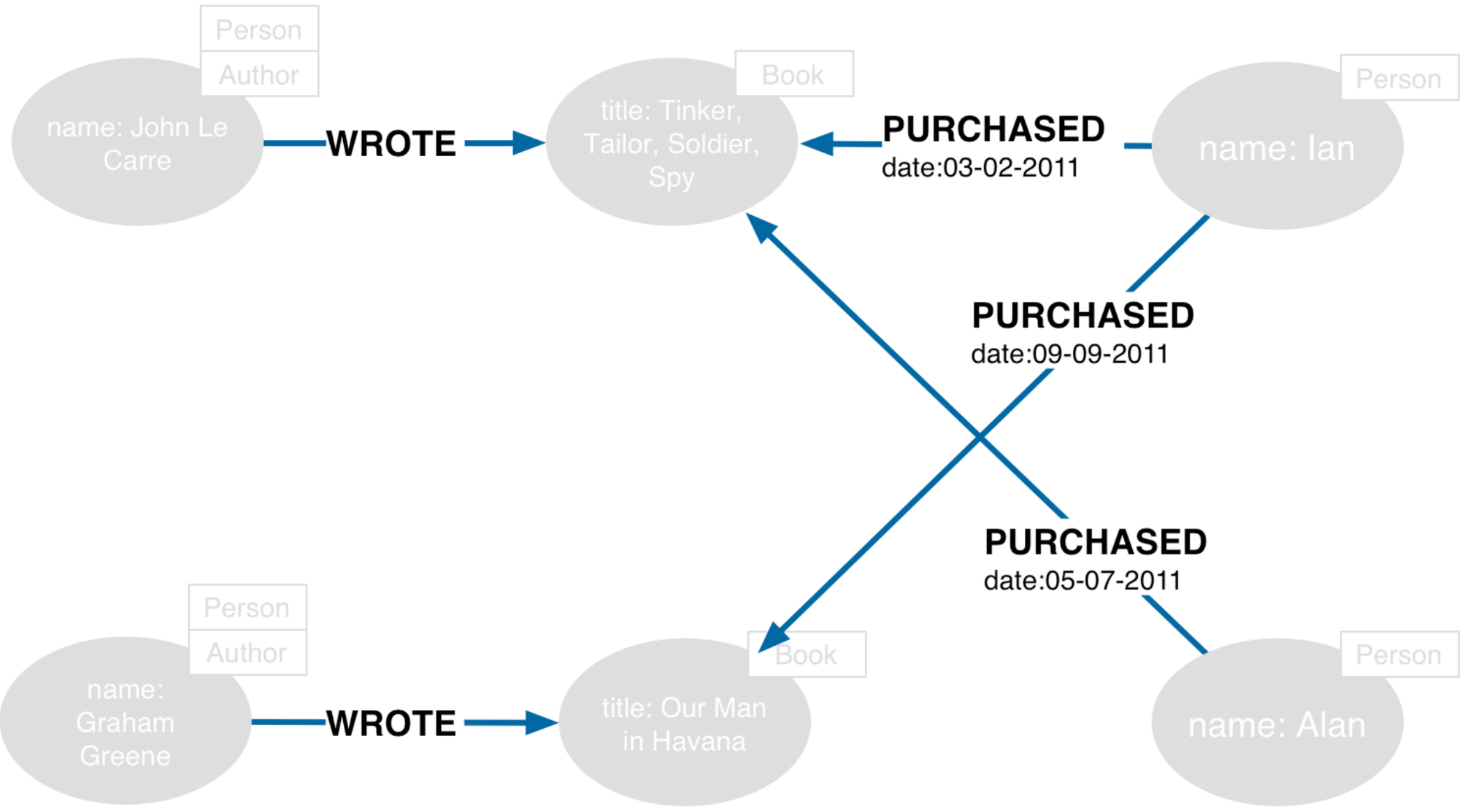
Property Graph Data Model



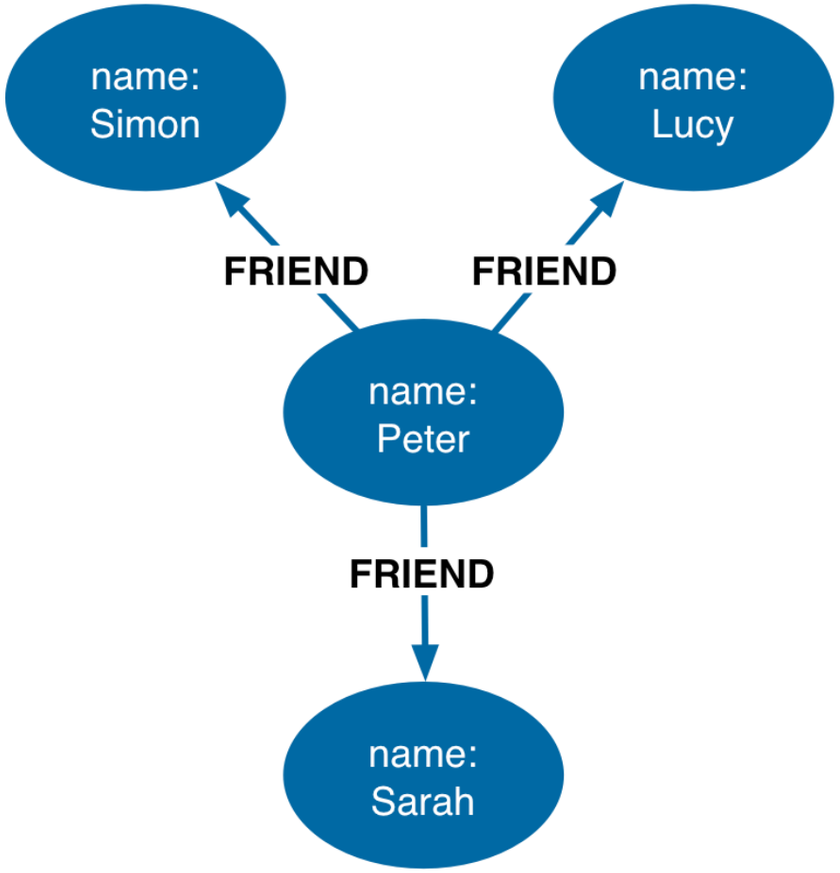
Nodes



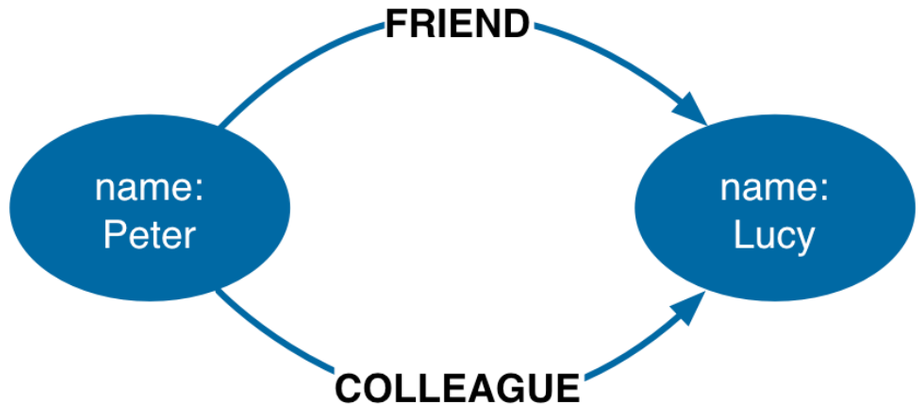
Relationships



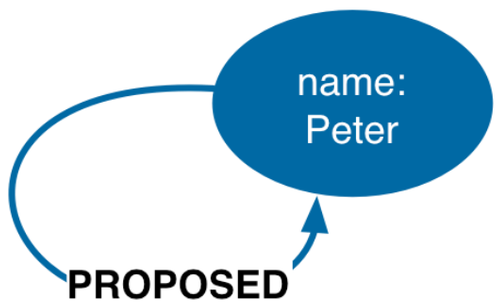
Relationships (continued)



Nodes can have more than one relationship

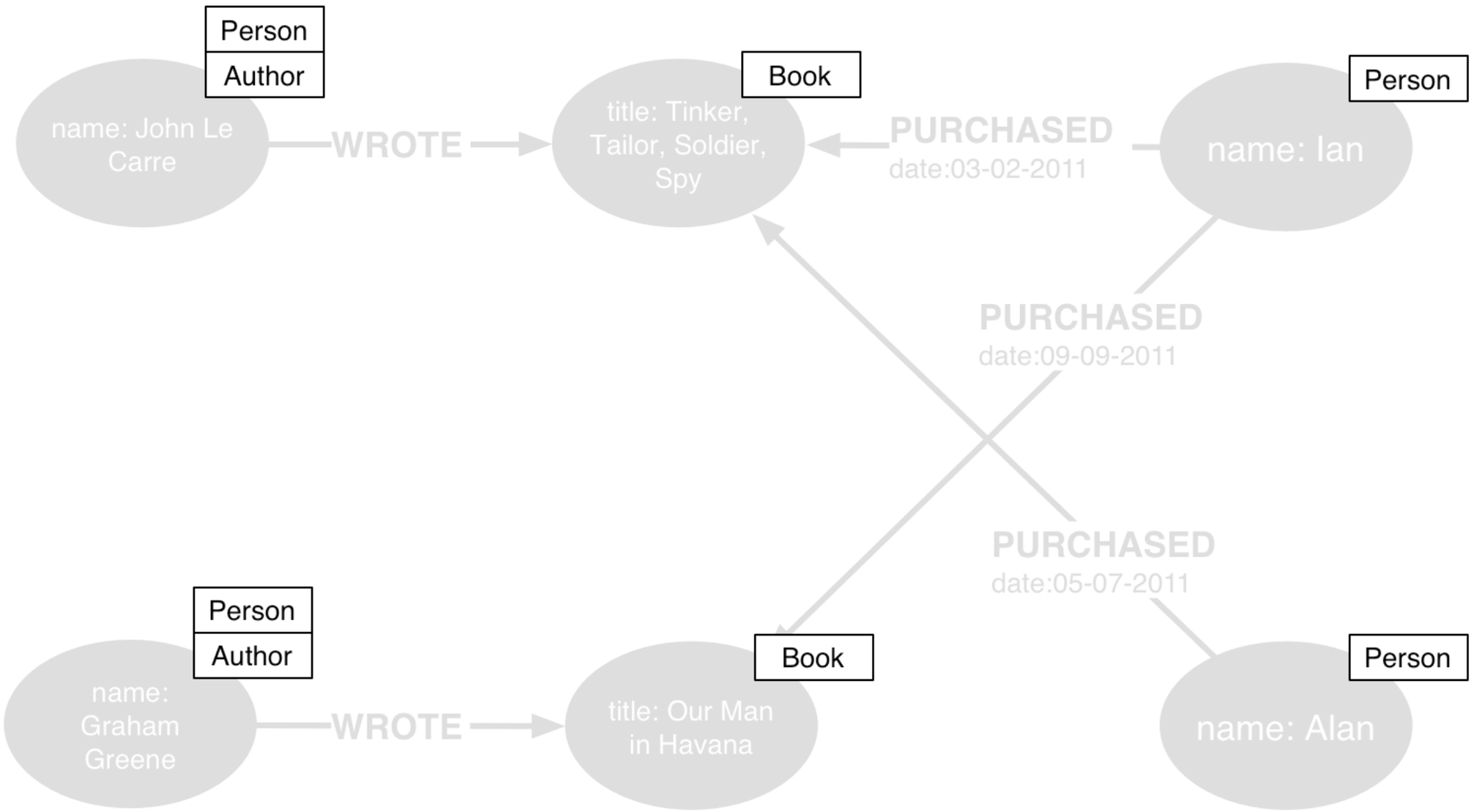


Nodes can be connected by more than one relationship



Self relationships are allowed

Labels

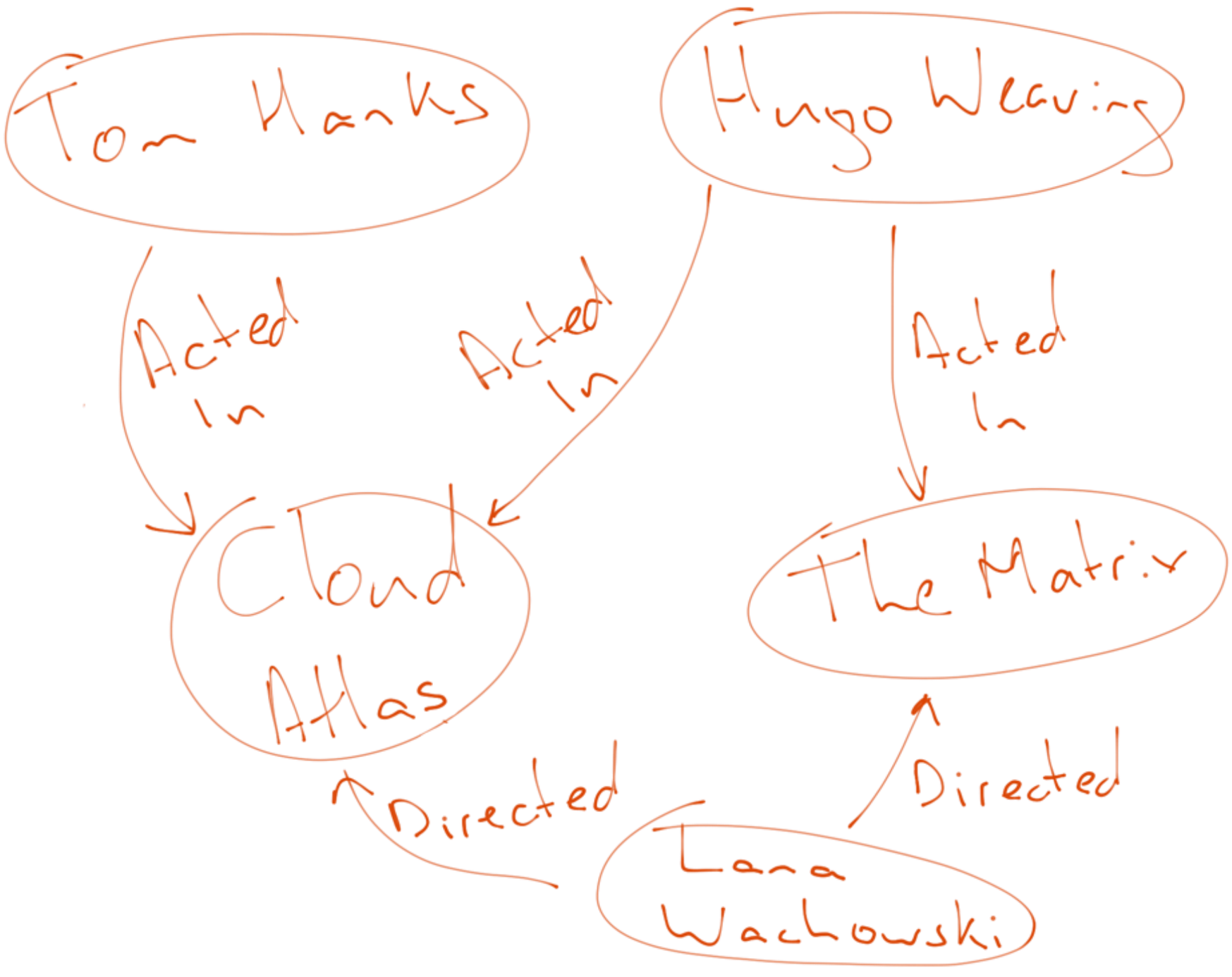


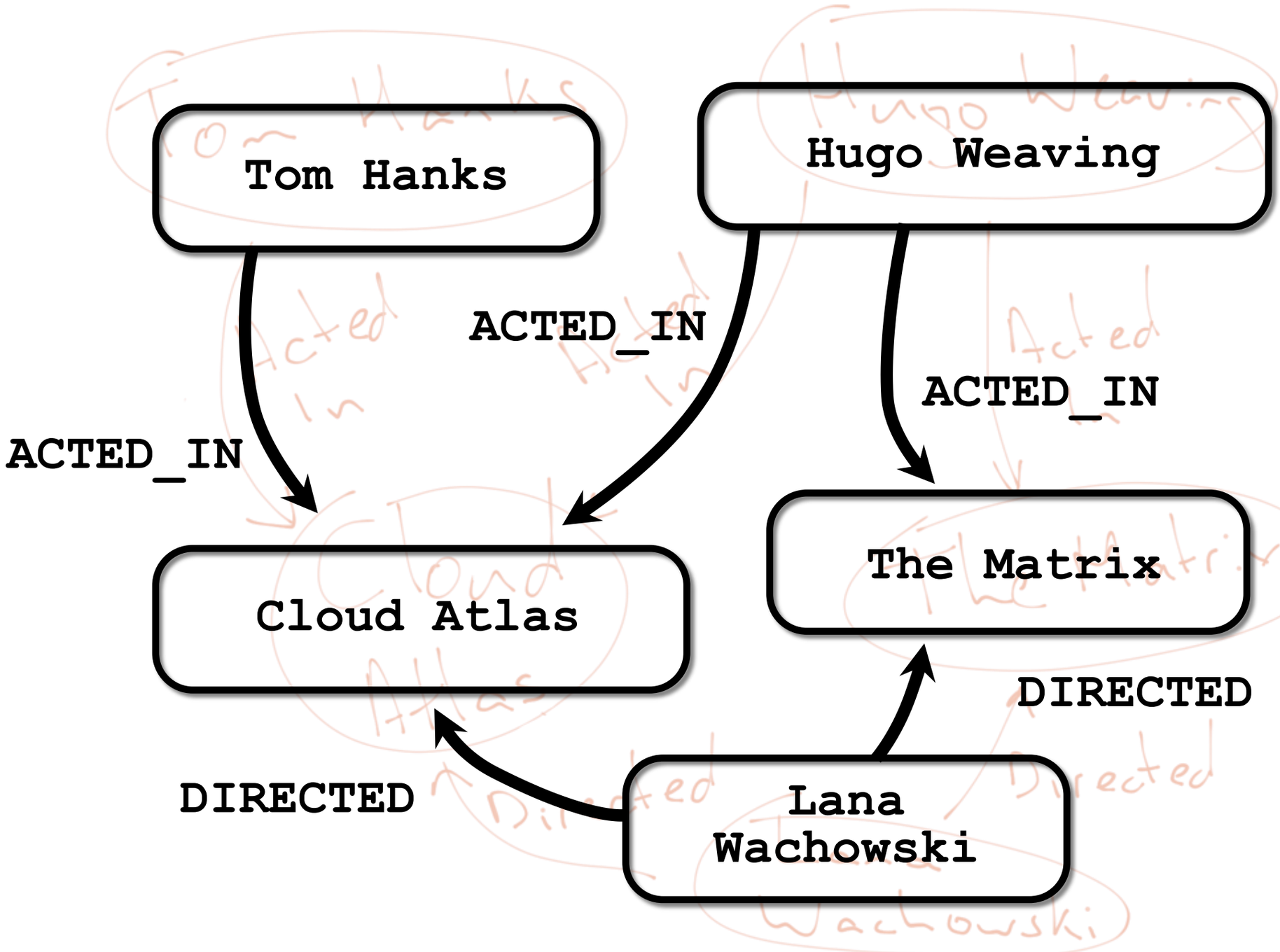
Four Building Blocks

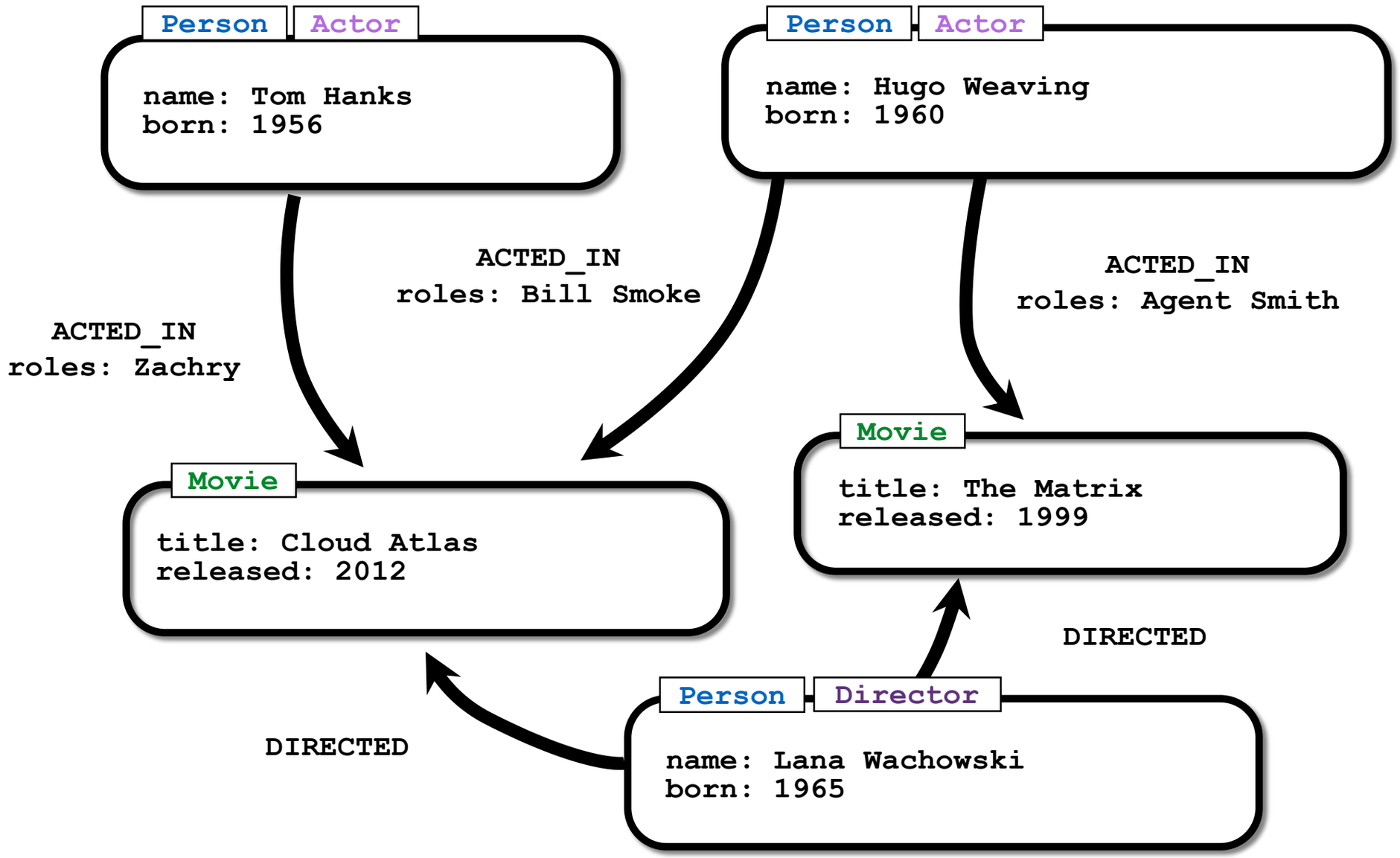
- Nodes
  - Entities
- Relationships
  - Connect entities and structure domain
- Properties
  - Attributes and metadata
- Labels
  - Group nodes by role

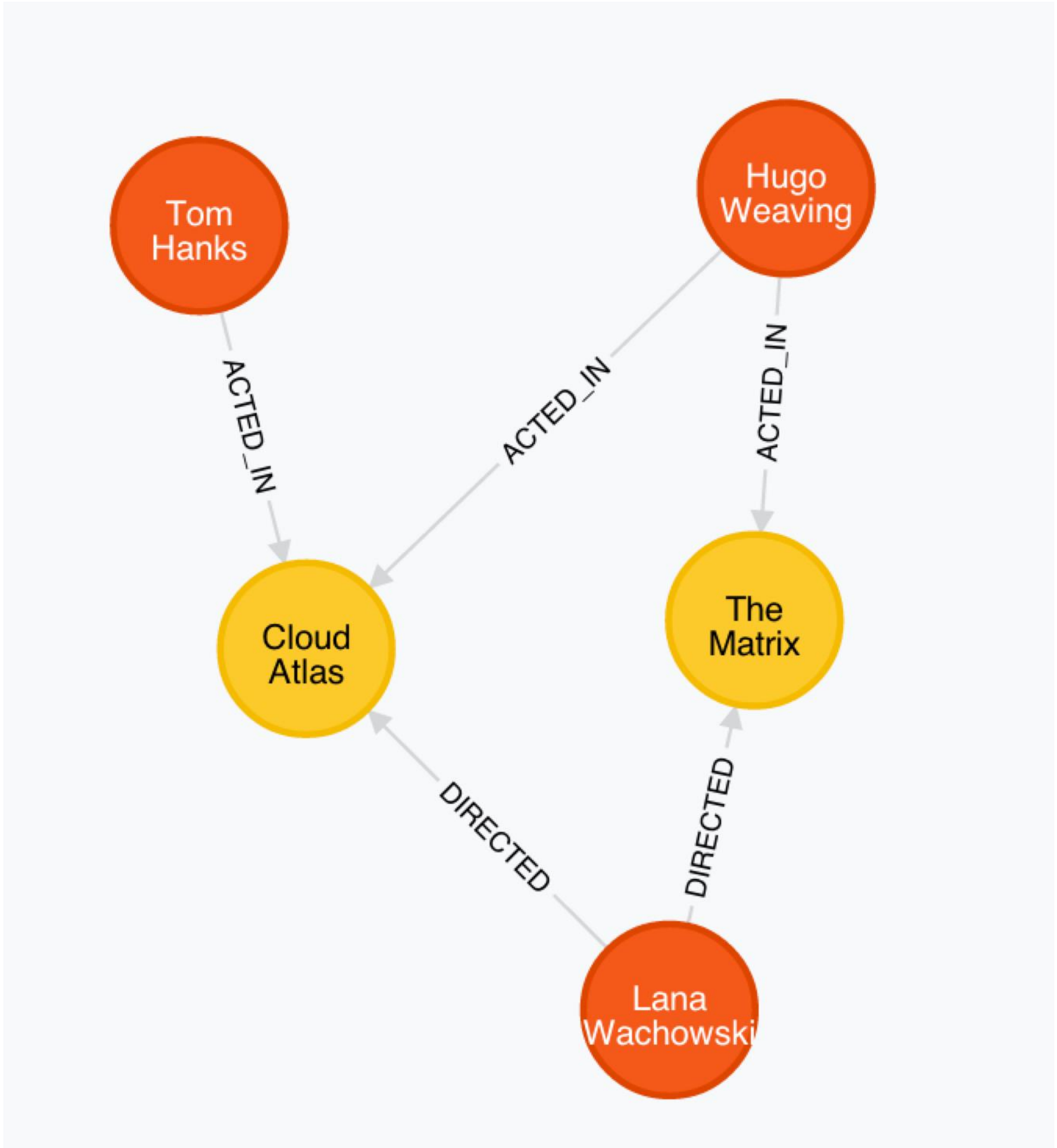
Whiteboard Friendlyness

Easy to design and model
direct representation of the model

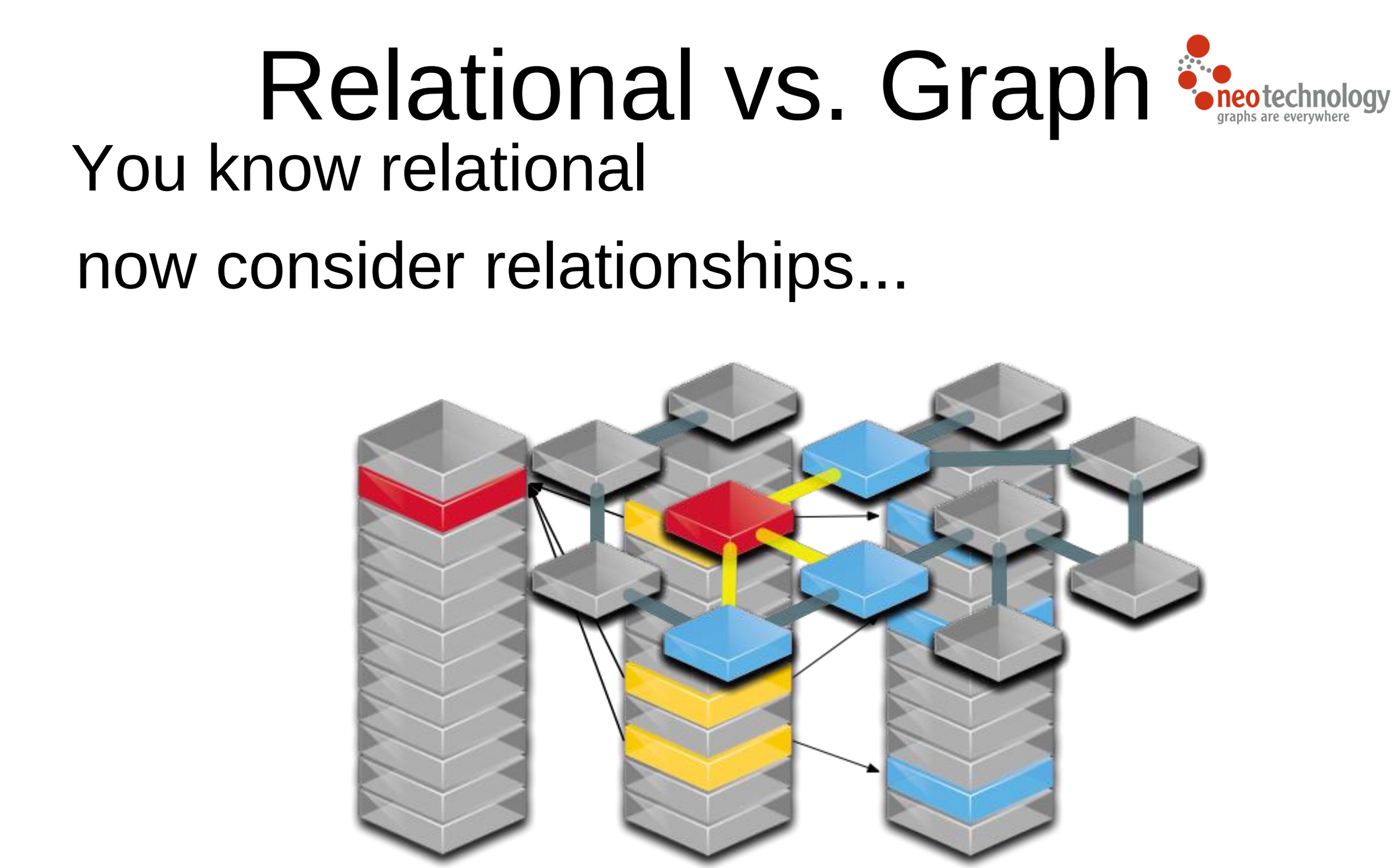







Tom Hanks
Hugo Weaving
ACTED_IN
ACTED_IN
ACTED_IN
Cloud Atlas
The Matrix
DIRECTED
DIRECTED
Lana Wachowski

Relational vs. Graph



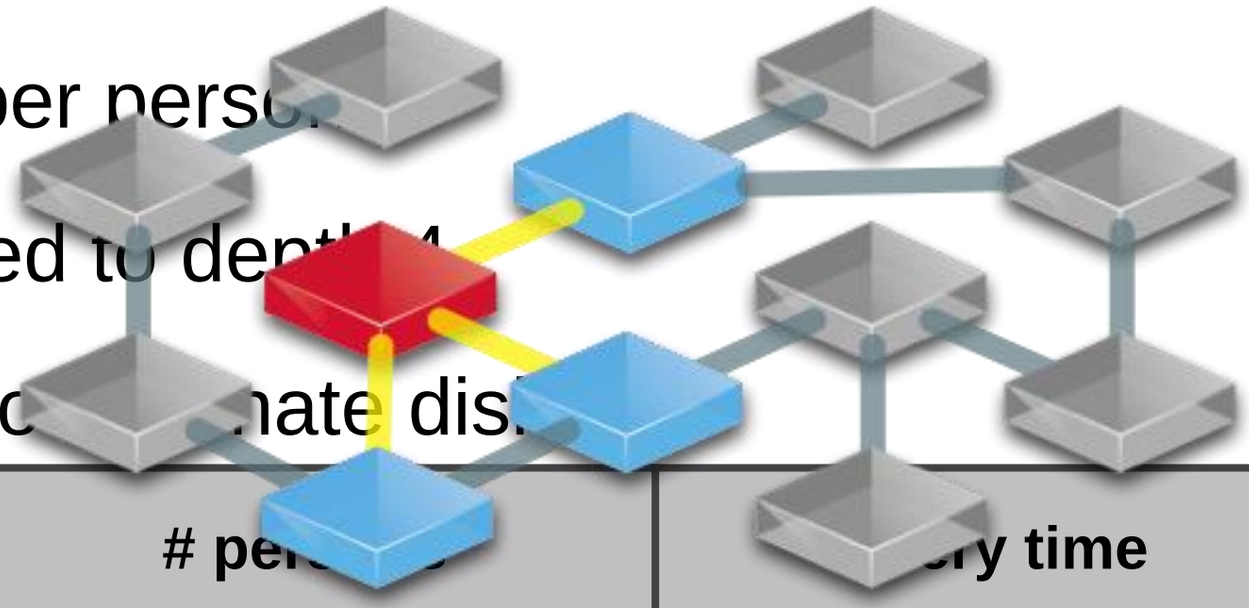
You know relational

now consider relationships...

users user_skill skills

Looks different, fine. Who cares?

- a sample social graph
  - with ~1,000 persons
- average 50 friends per perso
- pathExists(a,b) limited to dept
- caches warmed up to



	# pe	ry time
Relational database	1.000	2000ms
Neo4j	1.000	2ms
Neo4j	1.000.000	2ms

Neo4j is a Graph Database

Neo4j is a Graph Database

- A Graph Database:
  - a schema-free labeled Property Graph
  - perfect for complex, highly connected data
- A Graph Database:
  - reliable with real ACID Transactions
  - scalable: Billions of Nodes and Relationships, Scale out with highly available Neo4j-Cluster
  - fast with more than 2-4M traversals / second
  - Server with HTTP API, or Embeddable on the JVM
  - Declarative Query Language

Graph Database: Pros & Cons

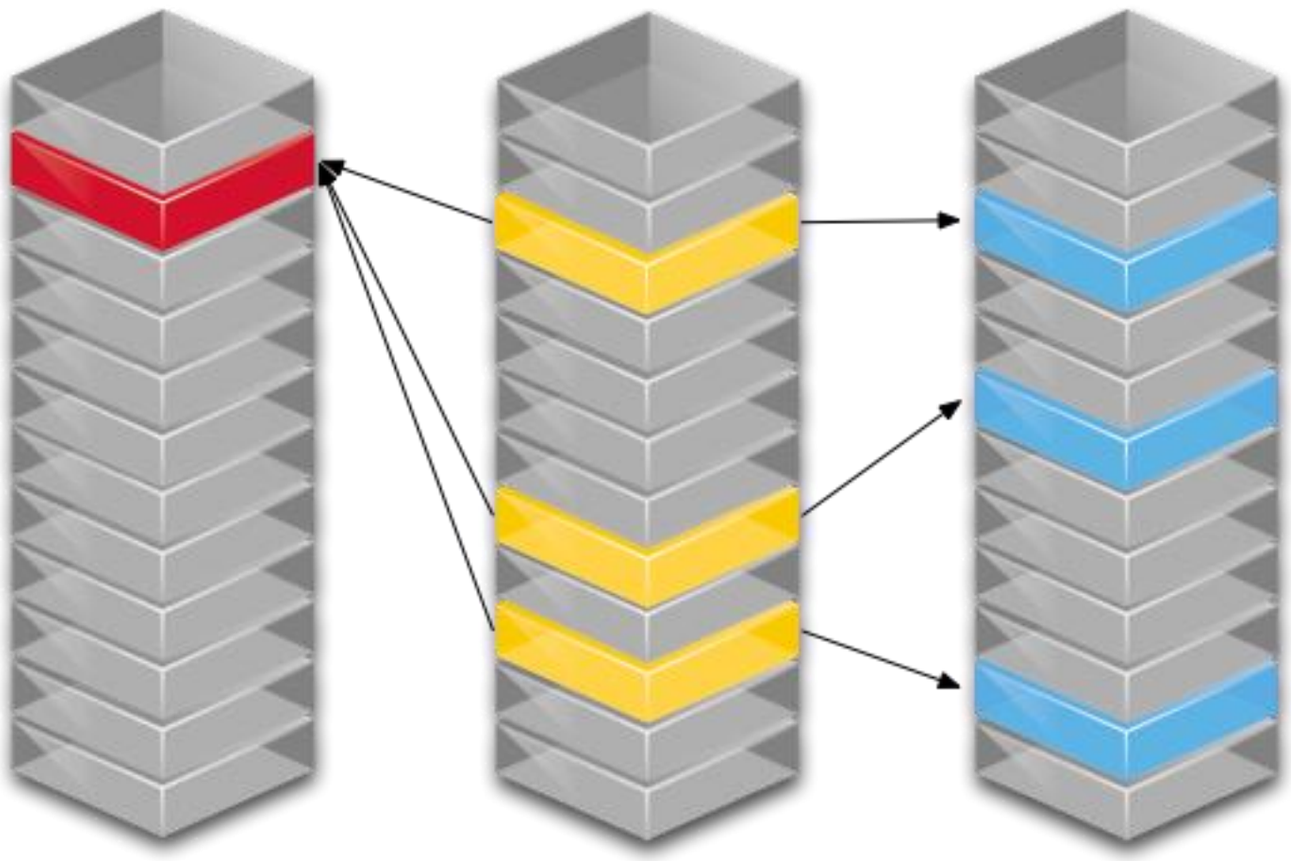
- Strengths
  - Powerful data model, as general as RDBMS
  - Whiteboard friendly, agile development
  - Fast, for connected data
    - Easy to query
- Weaknesses:
  - Sharding
  - Global Queries / Number Crunching
  - Binary Data / Blobs
  - Requires conceptual shift
    - graph-like thinking becomes addictive

Graph Querying

You know how to query a relational database!

Just use SQL

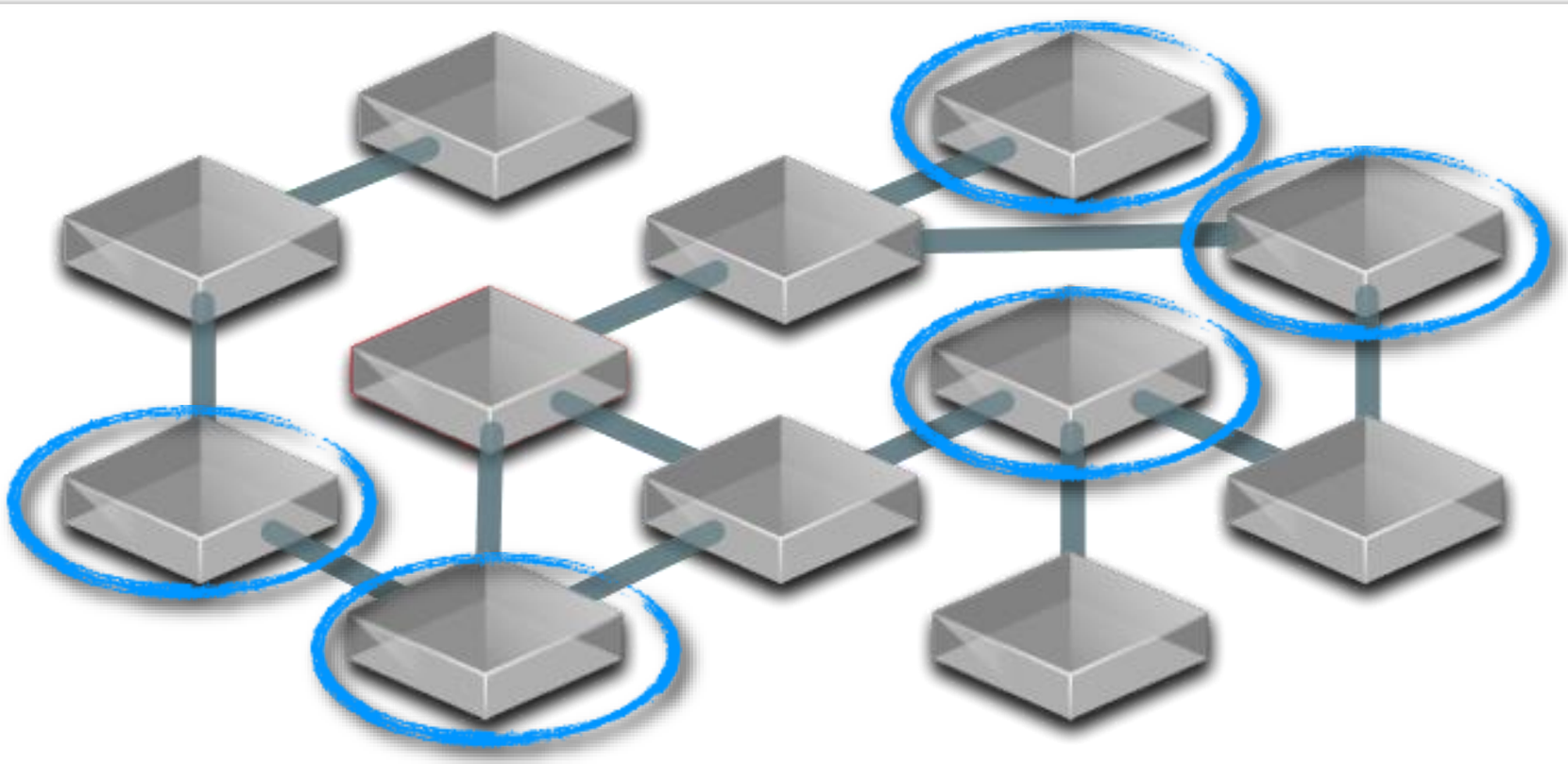
```
select skills.name
from users join user_skills on (...) join skills on (...)
where users.name = "Michael"
```



How to query a graph?

You traverse the graph

```
// then traverse the relationships
MATCH (me:Person {name:'Andreas'})-[:FRIEND]->(friend)
                                 -[:FRIEND]->(friend2)
RETURN friend2
```



Cypher

a pattern-matching
query language for graphs

Cypher attributes

#1 Declarative

You tell Cypher what you want, not how to get it

Cypher attributes

#2 Expressive

Optimize syntax for reading

```
MATCH (a:Actor)-[r:ACTS_IN]->(m:Movie)
RETURN a.name, r.role, m.title
```

Cypher attributes

#3 Pattern Matching

Patterns are easy for your human brain

Cypher attributes

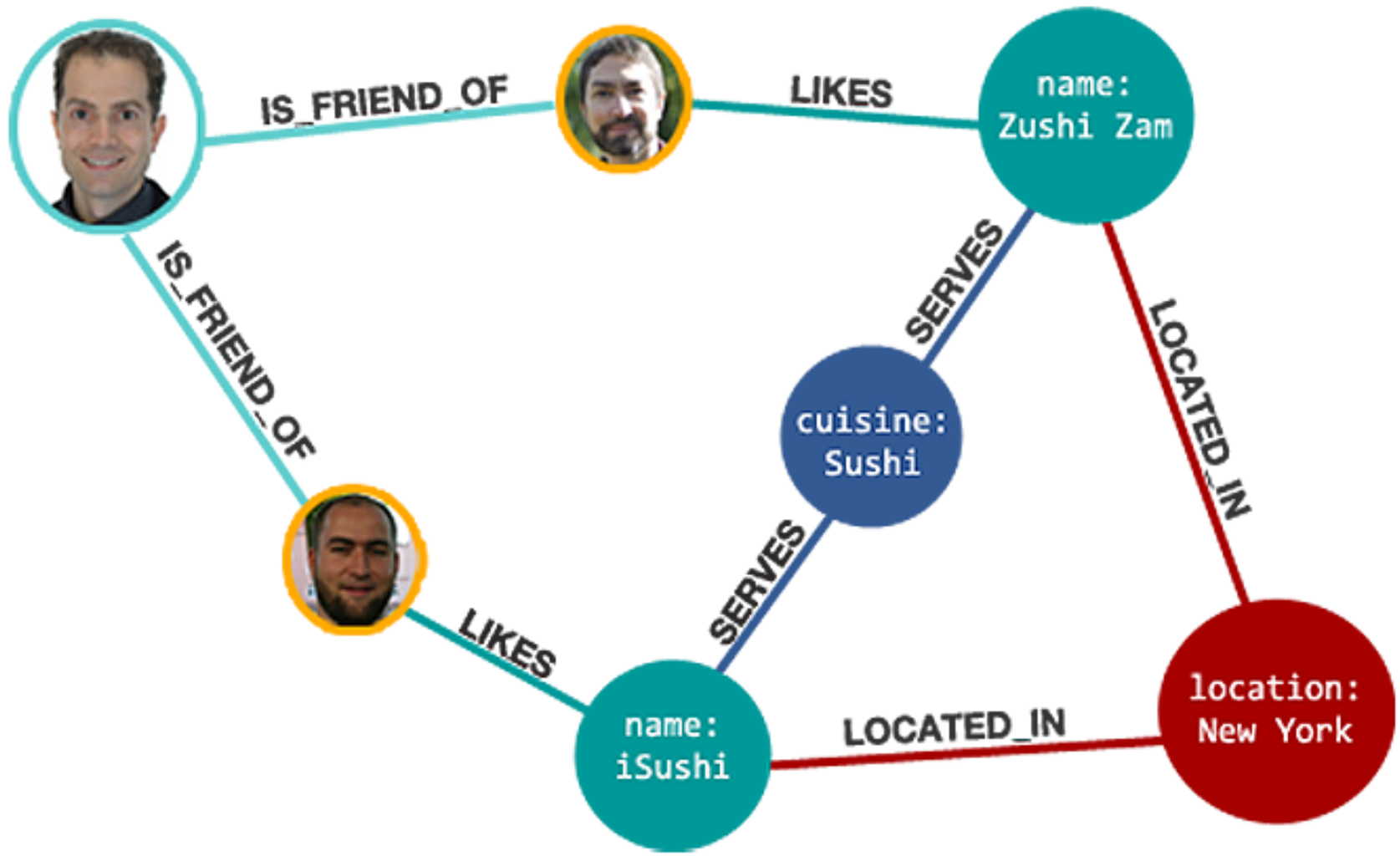
#4 Idempotent

State change should be expressed idempotently

Graph Query Examples

Social

Recommendation



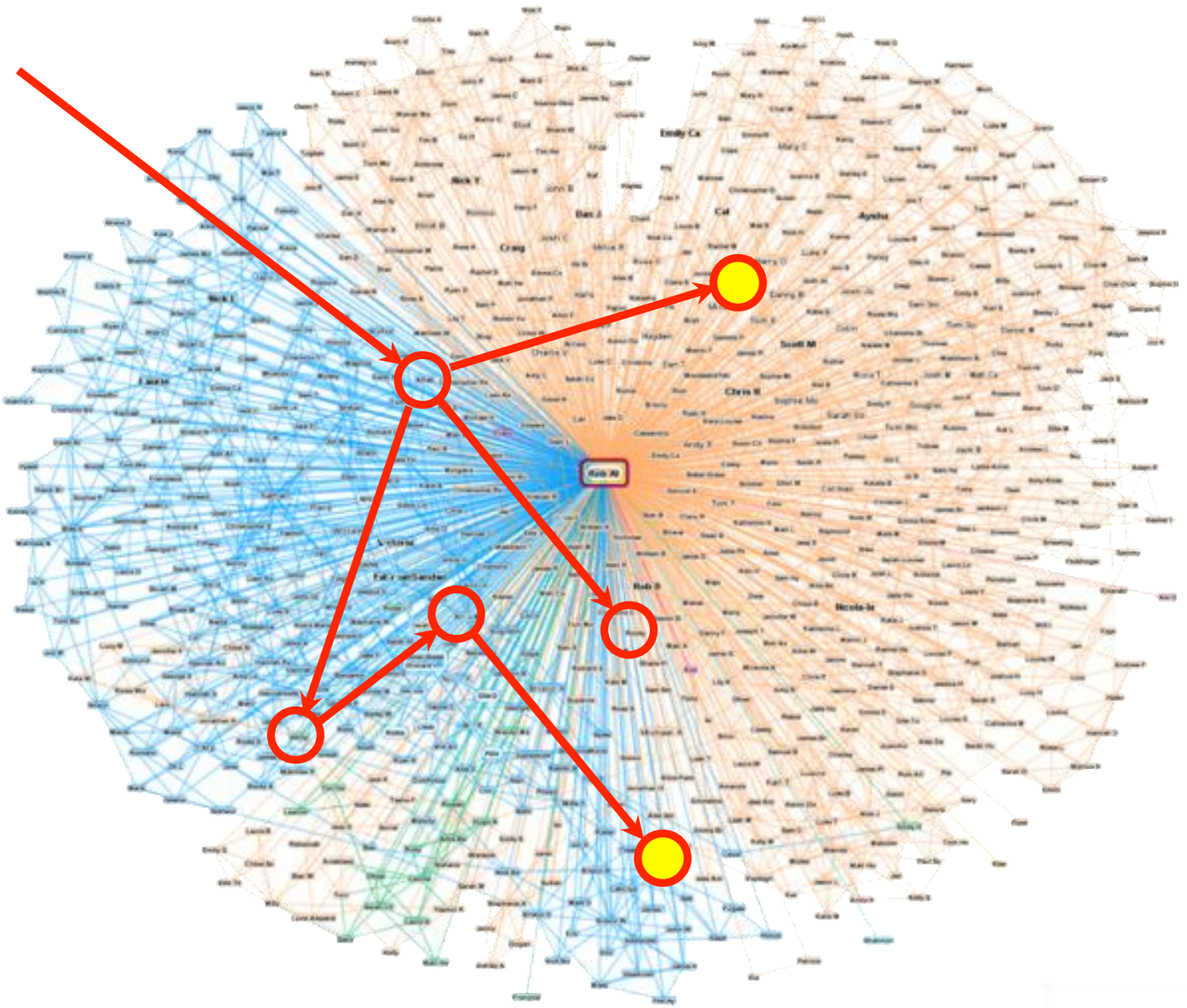

```
MATCH (person:Person)-[:IS_FRIEND_OF]->(friend),
 (friend)-[:LIKES]->(restaurant),
(restaurant)-[:LOCATED_IN]->(loc:Location),
(restaurant)-[:SERVES]->(type:Cuisine)

WHERE person.name = 'Philip' AND loc.location='New York' AND
      type.cuisine='Sushi'

RETURN restaurant.name
```

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

** Cypher query language example*



Network Management Example

Practical Cypher

Network Management - Create

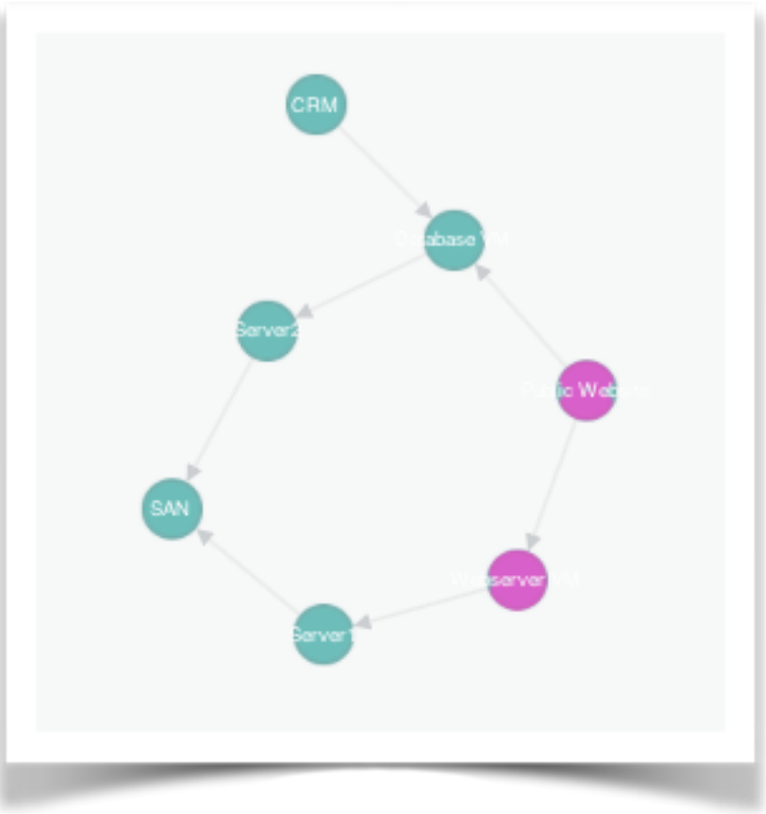
```
CREATE
    (crm {name:"CRM"}),
    (dbvm {name:"Database VM"}),
    (www {name:"Public Website"}),
    (wwwvm {name:"Webserver VM"}),
    (srv1 {name:"Server 1"}),
    (san {name:"SAN"}),
    (srv2 {name:"Server 2"}),

    (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
```

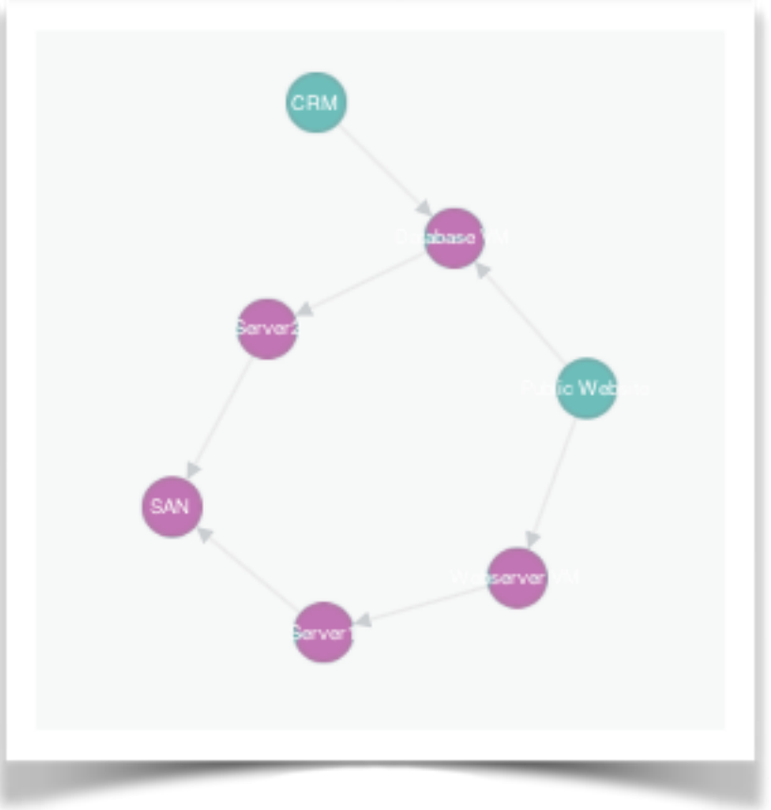


upstream
{name:"Webserver VM"}
{name:"Public Website"}

Practical Cypher

Network Management - Dependency Analysis

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

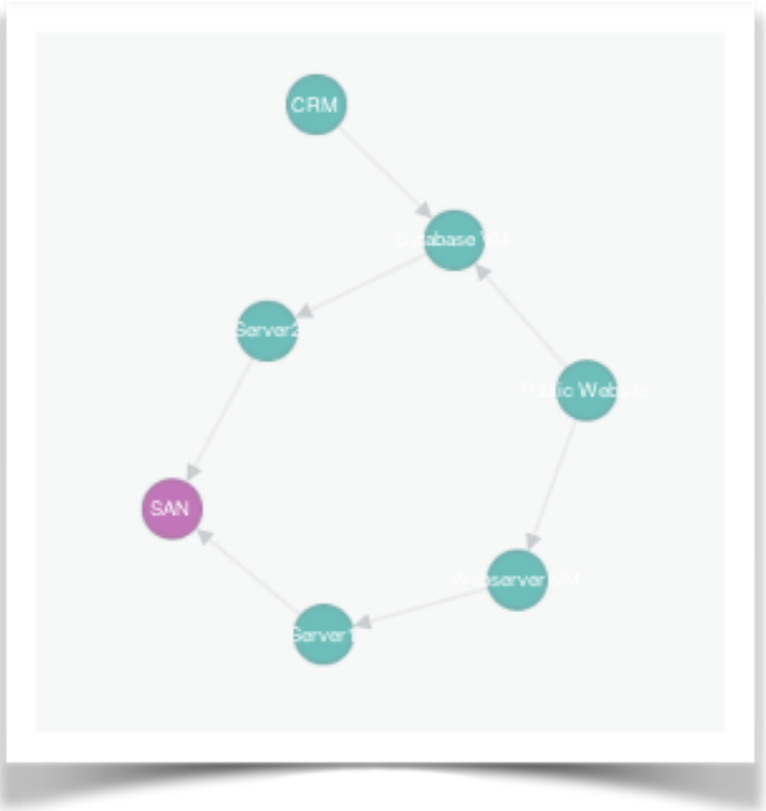


downstream
{name:"Database VM"}
{name:"Server 2"}
{name:"SAN"}
{name:"Webserver VM"}
{name:"Server 1"}

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

How to get started?

- Full day Neo4j Training & Online Training
- Free e-Books
  - Graph Databases, Learning Neo4j
- http://neo4j.com
- http://neo4j.com/developer
- http://neo4j.com/docs
- http://gist.neo4j.org
- Get Neo4j
  - http://neo4j.com/download
- Participate
  - http://groups.google.com/group/neo4j
  - http://neo4j.meetup.com
  - a session like this one ;)

Thank You

Time for Questions!