

No SQL?



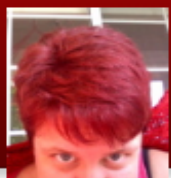
Neo4j

the benefits of
graph databases

Emil Eifrem
CEO, Neo Technology

#neo4j
@emileifrem
emil@neotechnology.com





Princess Polymath

mom
creator
diva
rogue

Graph Databases are the New Black

By Synedra on June 19, 2009 9:57 AM | [Permalink](#) | [Comments \(0\)](#) | [TrackBacks \(1\)](#)

When I started working on computers, the main thing people used databases for was storing tabular data. Sales reports, inventory records, lists of things. Relational databases were cool because you could associate a transaction in your sales table with an item in your inventory table. As the internet has grown, however, our database needs have outgrown the abilities of the standard relational database system. Social networks are a great example of this - the type of query needed to ask for [six degrees of Kevin Bacon](#) separation in a relational database is complex, resource intensive, and extremely time consuming. Many very smart people have come up with ways to work around this problem, but the real problem is that this type of question is not what relational databases are designed to handle.

[Graph databases](#), on the other hand, are optimized for exactly this type of question.

The problem with graph databases is that it's a whole different data structure that we've got to wrap our minds around, and it's difficult to do that without some starting place. [Freebase](#) is a publicly accessible, open graph database, available for people to use as the back end for their applications. They've even created a [suite of tools](#) to make it easier to build fun things on top of their extensive data. They will be giving a tutorial on [Semantic Technologies](#) at OSCON, and if you have any interest in learning about graph structure and what you can do with it, you should definitely check it out.

If you want a more data-agnostic view of graph technologies, or want to try installing a graph database on your own, you can learn about [Neo4J](#), an open source graph database, on Wednesday afternoon.

1 TrackBacks

Listed below are links to blogs that reference this entry: [Graph Databases are the New Black](#).

TrackBack URL for this entry: <http://www.princesspolymath.com/cgi-bin/mt/mt->

Search

About this Entry

[Open Source iPhone Development at OSCON](#) was the previous entry in this blog.

[Ignite! Talks at OSCON](#) is the next entry in this blog.

Find recent content on the [main index](#) or look in the [archives](#) to find all content.

Categories

[Cooking \(8\)](#)
[Fitness \(2\)](#)
[Food \(2\)](#)
[Geek Stuff \(21\)](#)
[Knitting \(38\)](#)
[Lap Band \(1\)](#)

Monthly Archives

[July 2009 \(1\)](#)
[June 2009 \(2\)](#)
[May 2009 \(1\)](#)
[March 2009 \(1\)](#)
[February 2009 \(3\)](#)
[January 2009 \(2\)](#)
[November 2008 \(2\)](#)
[October 2008 \(4\)](#)
[September 2008 \(5\)](#)
[April 2008 \(1\)](#)
[March 2008 \(2\)](#)
[February 2008 \(3\)](#)
[January 2008 \(1\)](#)
[November 2007 \(4\)](#)
[October 2007 \(4\)](#)
[September 2007 \(5\)](#)
[August 2007 \(10\)](#)
[October 2006 \(1\)](#)
[April 2006 \(1\)](#)
[March 2006 \(2\)](#)



- Questions
- Tags
- Users
- Badges
- Unanswered

Ask Question

Hype around graph databases... why ?

File Search

Search your intranet, file shares & more with solutions from Google
www.google.com/enterprise

Database Management

Free Whitepaper, What is Enterprise Open Source. Download Now!
www.ingres.com

Ads by Google

Hello World!
Stack Overflow is a **collaboratively edited question and answer site for programmers** – regardless of platform or language. It's 100% free, no registration required.
about » faq »

there is some hype around graph databases, I'm wondering why.

0 What are the possible problems that one can be confronted to in today's web environment that can be solved using graph databases? And are graph databases suitable for classical applications and can come as a drop-in replacement of classical Relational Databases. So in fact it's two questions in one.

graph database web neo4j applications

flag

asked 2 days ago
ephemerebis
15 ● 3

- tagged
- database × 3789
 - web × 1015
 - graph × 104
 - applications × 62
 - neo4j × 2

4 Answers

oldest newest votes

3
Many relational representations of graphs aren't particularly efficient for all operations you might want to perform.
For example, if one wants the connected set of all nodes where edges satisfy a given predicate, starting from a given node, there's no natural way in SQL to express that. Likely you'll either do a query for edges with the predicate, and then have to exclude disconnected edges locally, or have a very verbose conversation with the database server following one set of links to the next in iterated queries.
Graphs aren't a general replacement for relational databases. RDBs deal primarily in sets (tables), while

asked 2 days ago
viewed 122 times
latest activity yesterday

Death?

Community experimentation:

- CouchDB
- Redis
- Hypertable
- Cassandra
- Scalaris
- ...

Tuesday, February 5, 2008

The Death of the Relational Database

The relational database is becoming increasingly less useful in a web 2.0 world. The reason for this is that, while the relational database model is great for storing information, it is horrible for storing knowledge. By knowledge I mean information that has value beyond the narrow current conception of the given application. I mean information that can have enduring value. In this context, one might say knowledge is information in a disposable form.

The RDBMS is not enough.

Posted by [Sebastien Auvray](#) on Nov 26, 2007 09:30 AM

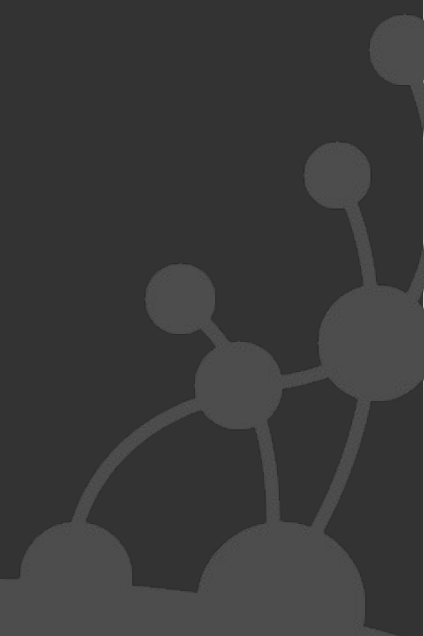
Community [Architecture](#), [Ruby](#) Topics [Performance & Scalability](#), [Data Access](#), [Database Design](#)

Tags: [CouchDB](#), [Database](#), [Database Management](#), [Distributed Document Oriented Database](#), [S3](#), [RDD](#), [Relational Databases](#), [Scalability](#)

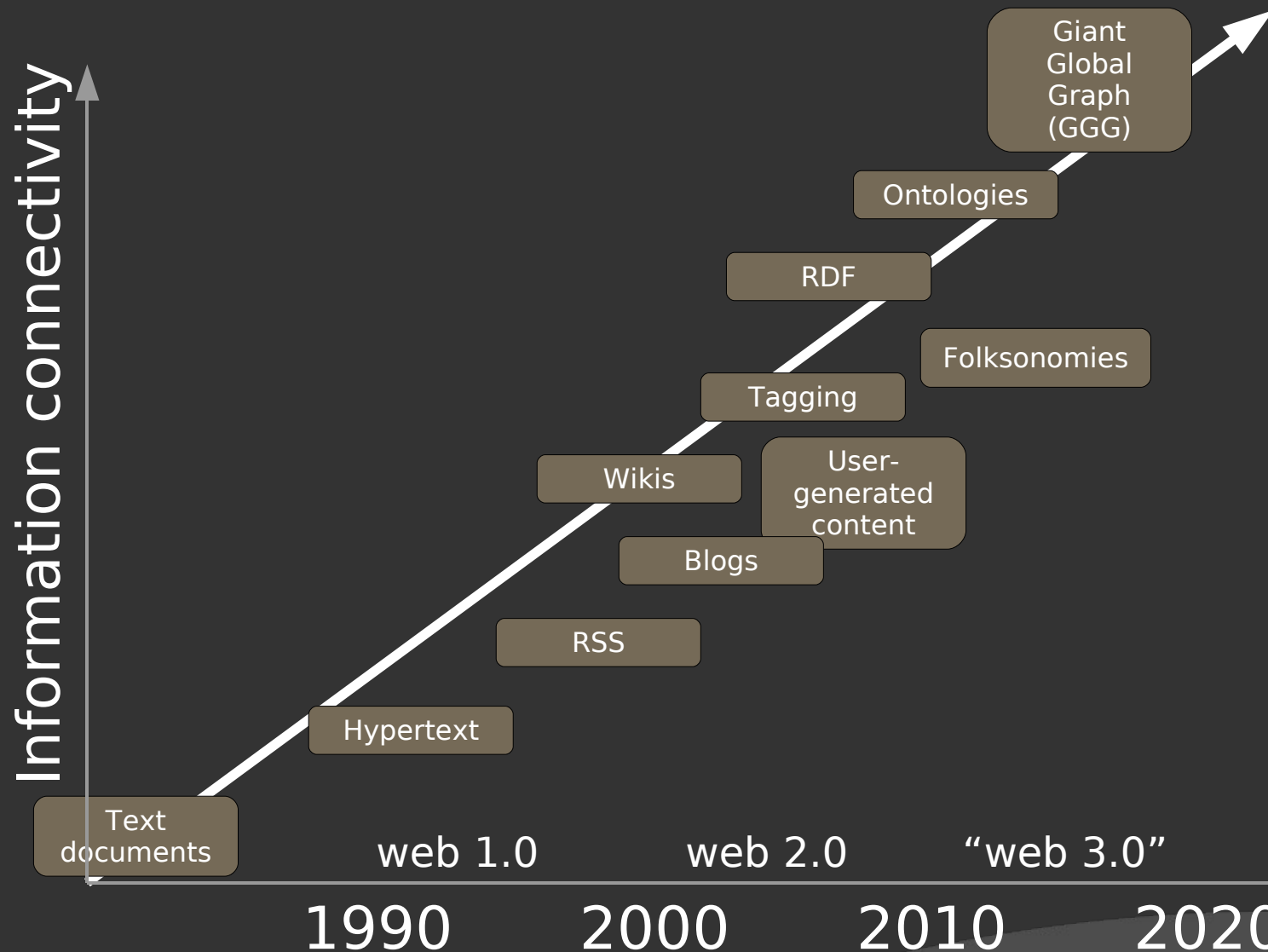
While Relational Databases fit a client-server model, in a [world of services new solutions are needed](#), RDBMS are subject to scalability issues: [How to create redundancy, parallelism ?](#)

CouchDB: Thinking beyond the RDBMS

September 2nd, 2007



Trend 1: data is getting more connected

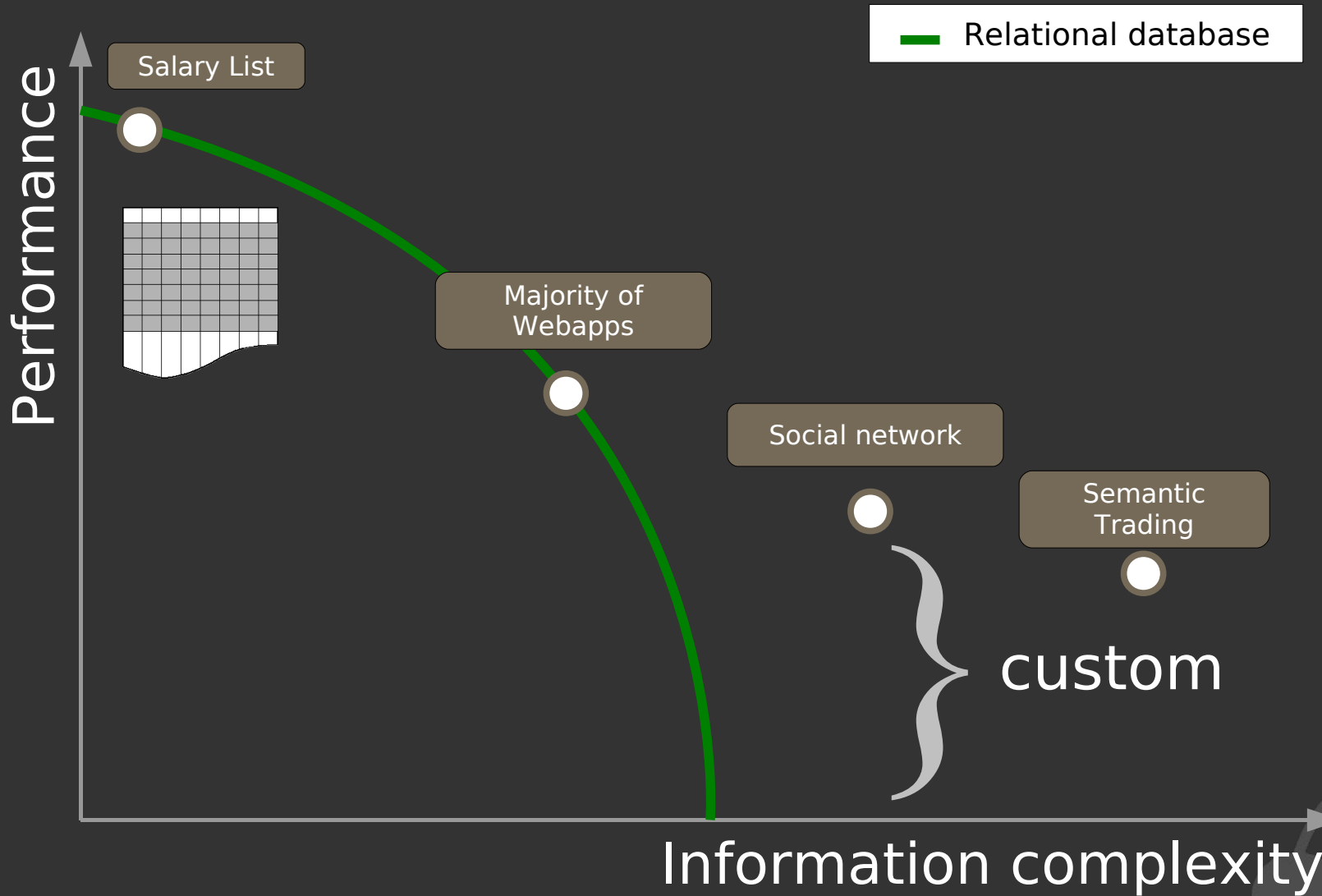


Trend 2: ... and more semi-structured

◎ Individualization of content!

- In the salary lists of the 1970s, all elements had exactly one job
- In the salary lists of the 2000s, we need 5 job columns! Or 8? Or 15?

◎ Trend accelerated by the decentralization of content generation that is the hallmark of the age of participation (“web 2.0”)

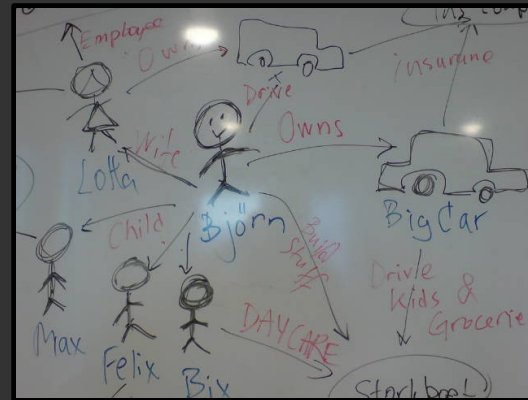


We = hackers!

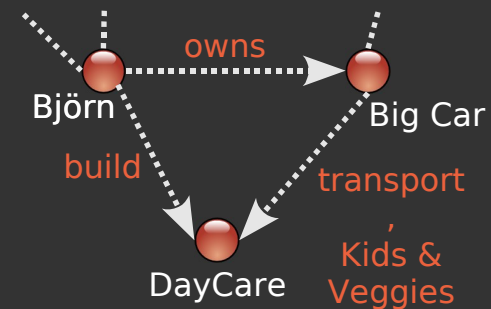
So that's v_{CPU} ...

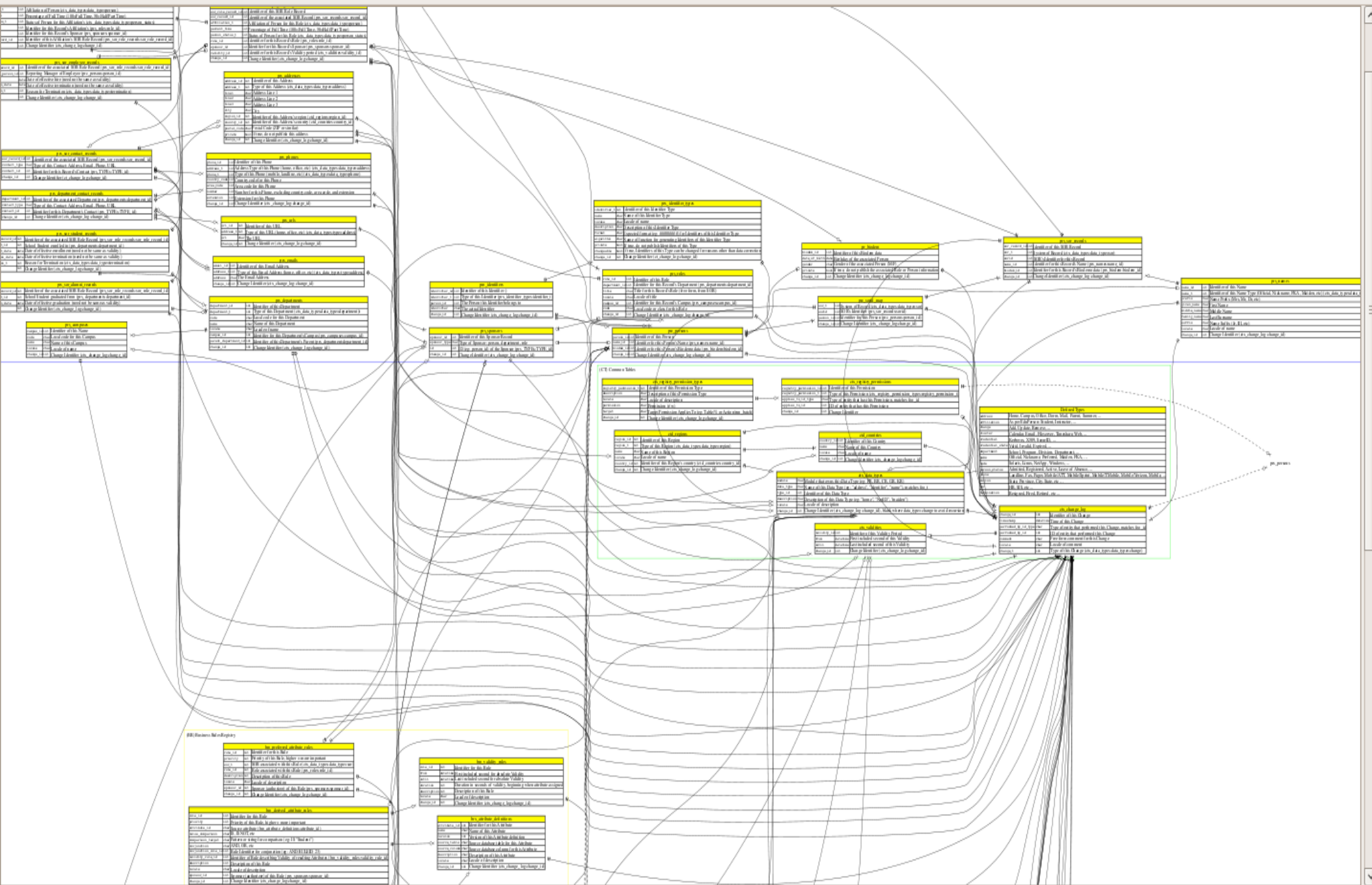
what about v_{hackers} ?

Whiteboard friendly?



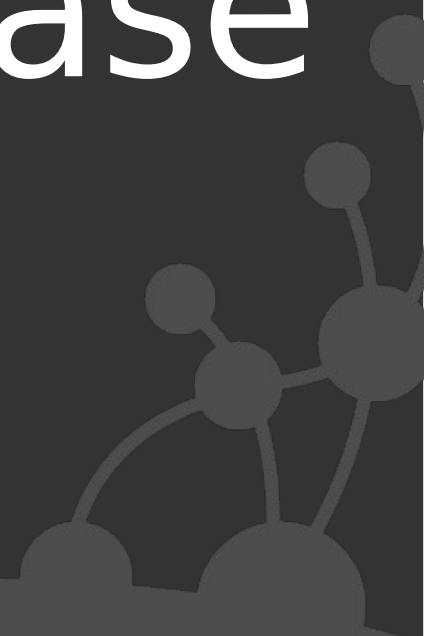
?





Alternative?

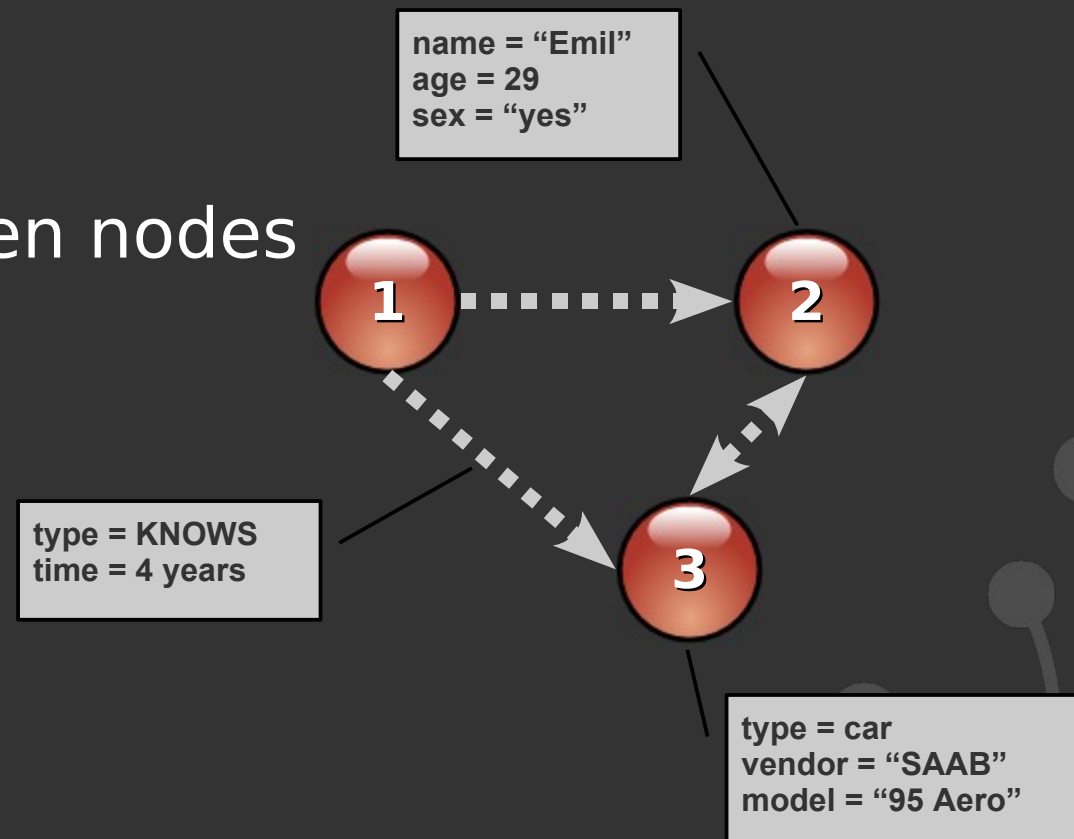
a graph database



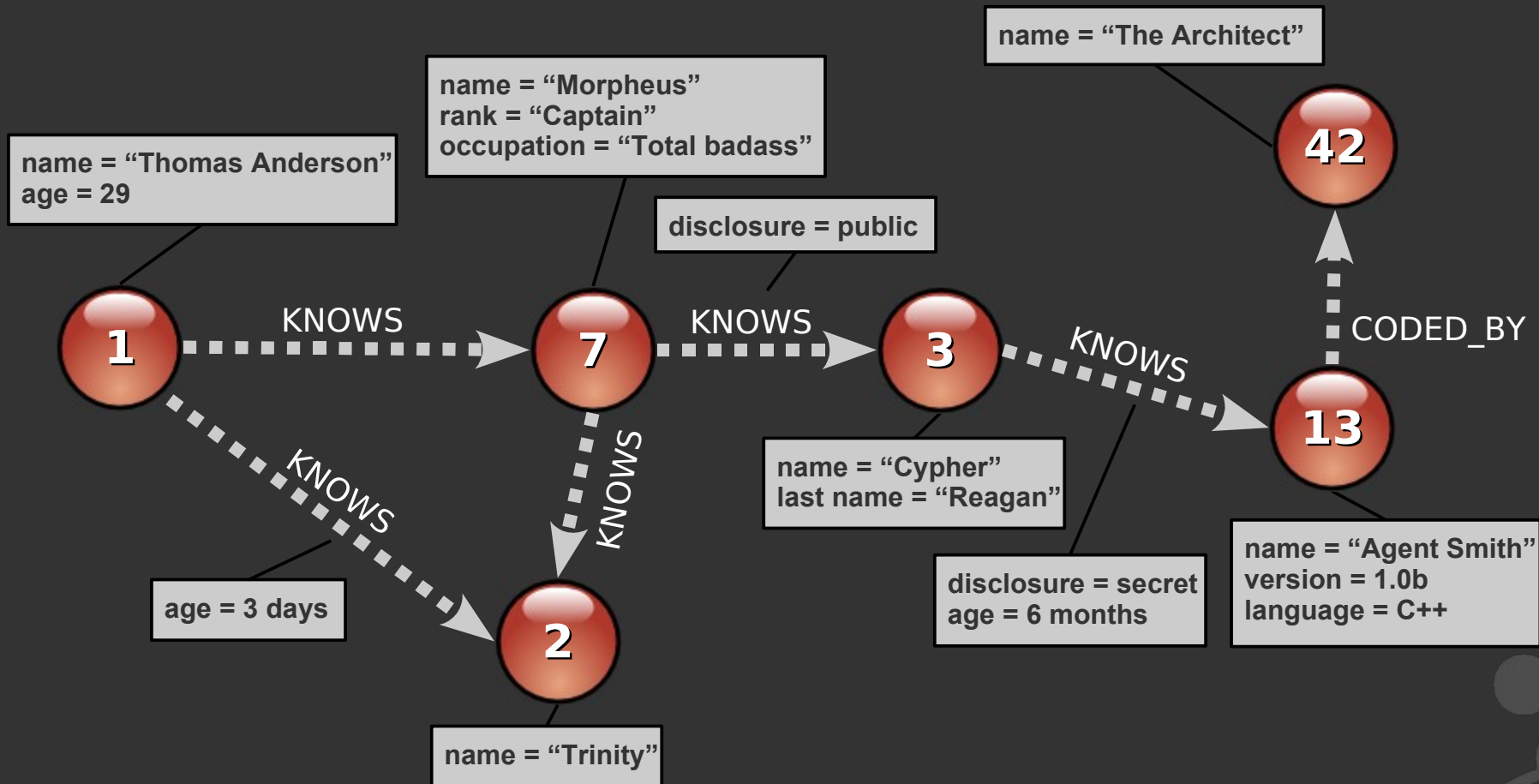
The Graph DB model: representation

Core abstractions:

- Nodes
- Relationships between nodes
- Properties on both



Example: The Matrix



Code (1): Building a node space

```
NeoService neo = ... // Get factory

// Create Thomas 'Neo' Anderson
Node mrAnderson = neo.createNode();
mrAnderson.setProperty( "name", "Thomas Anderson" );
mrAnderson.setProperty( "age", 29 );

// Create Morpheus
Node morpheus = neo.createNode();
morpheus.setProperty( "name", "Morpheus" );
morpheus.setProperty( "rank", "Captain" );
morpheus.setProperty( "occupation", "Total bad ass" );

// Create a relationship representing that they know each other
mrAnderson.createRelationshipTo( morpheus, RelTypes.KNOWS );
// ...create Trinity, Cypher, Agent Smith, Architect similarly
```


Code (1): Building a node space

```
NeoService neo = ... // Get factory
Transaction tx = neo.beginTransaction();

// Create Thomas 'Neo' Anderson
Node mrAnderson = neo.createNode();
mrAnderson.setProperty( "name", "Thomas Anderson" );
mrAnderson.setProperty( "age", 29 );

// Create Morpheus
Node morpheus = neo.createNode();
morpheus.setProperty( "name", "Morpheus" );
morpheus.setProperty( "rank", "Captain" );
morpheus.setProperty( "occupation", "Total bad ass" );

// Create a relationship representing that they know each other
mrAnderson.createRelationshipTo( morpheus, RelTypes.KNOWS );
// ...create Trinity, Cypher, Agent Smith, Architect similarly

tx.commit();
```

Code (1b): Defining RelationshipTypes

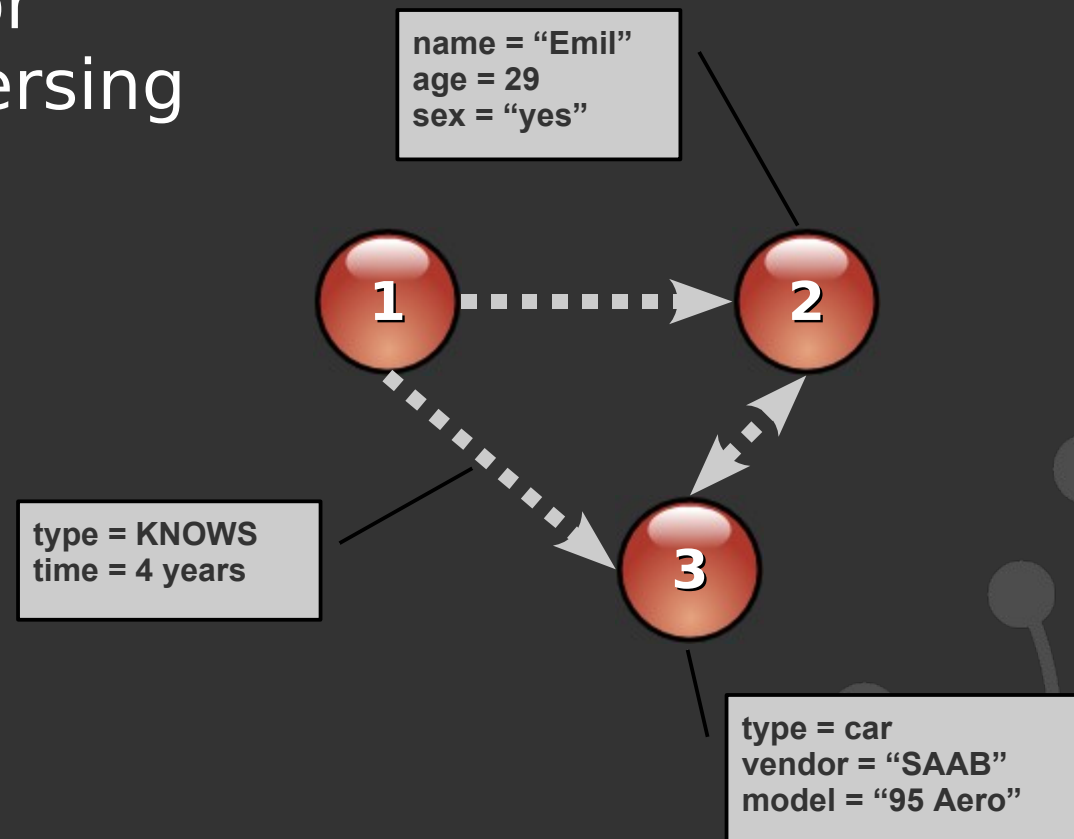
```
// In package org.neo4j.api.core
public interface RelationshipType
{
    String name();
}

// In package org.yourdomain.yourapp
// Example on how to roll dynamic RelationshipTypes
class MyDynamicRelType implements RelationshipType
{
    private final String name;
    MyDynamicRelType( String name ){ this.name = name; }
    public String name() { return this.name; }
}

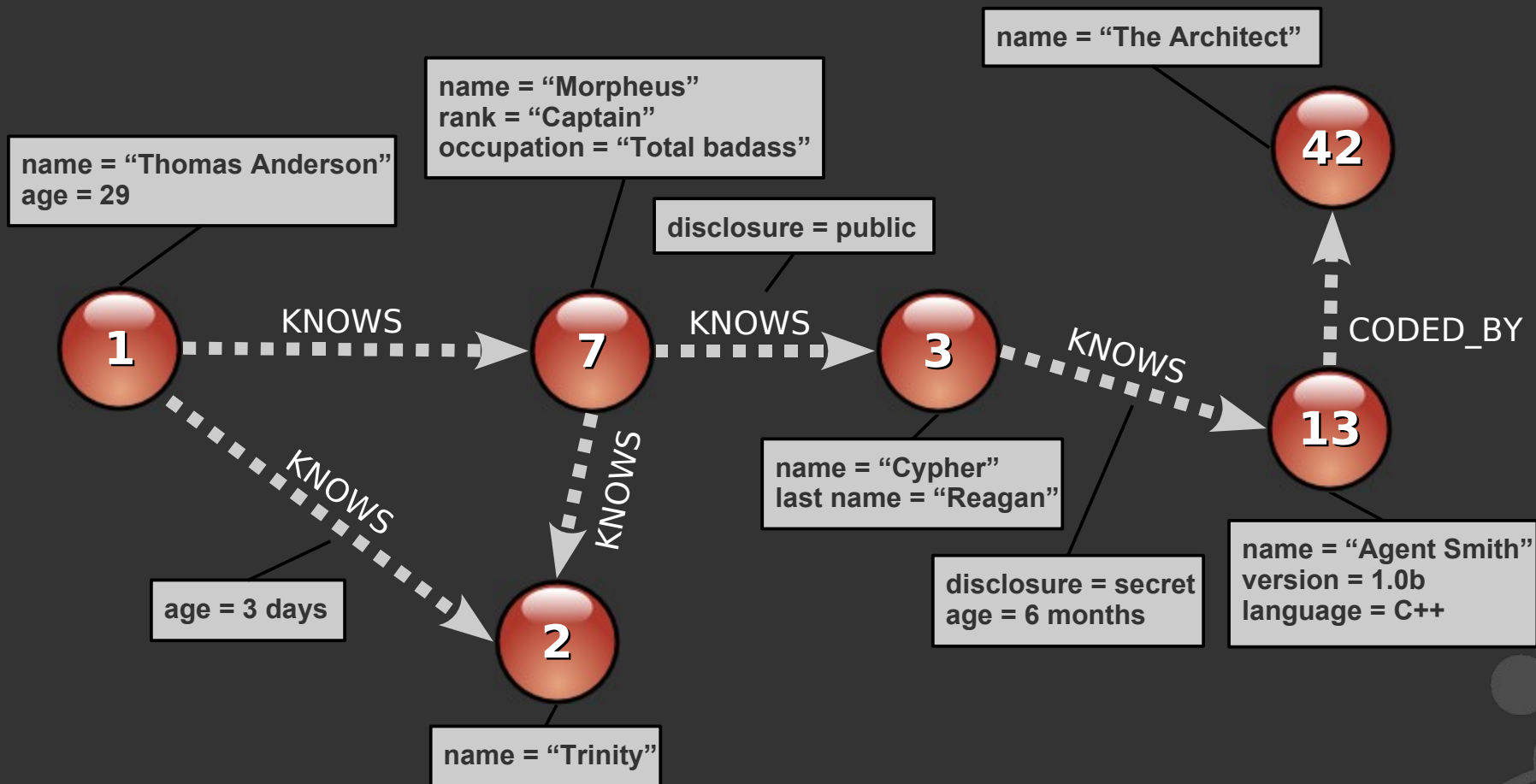
// Example on how to kick it, static-RelationshipType-like
enum MyStaticRelTypes implements RelationshipType
{
    KNOWS,
    WORKS_FOR,
}
```

The Graph DB model: traversal

- ◎ Traverser framework for high-performance traversing across the node space



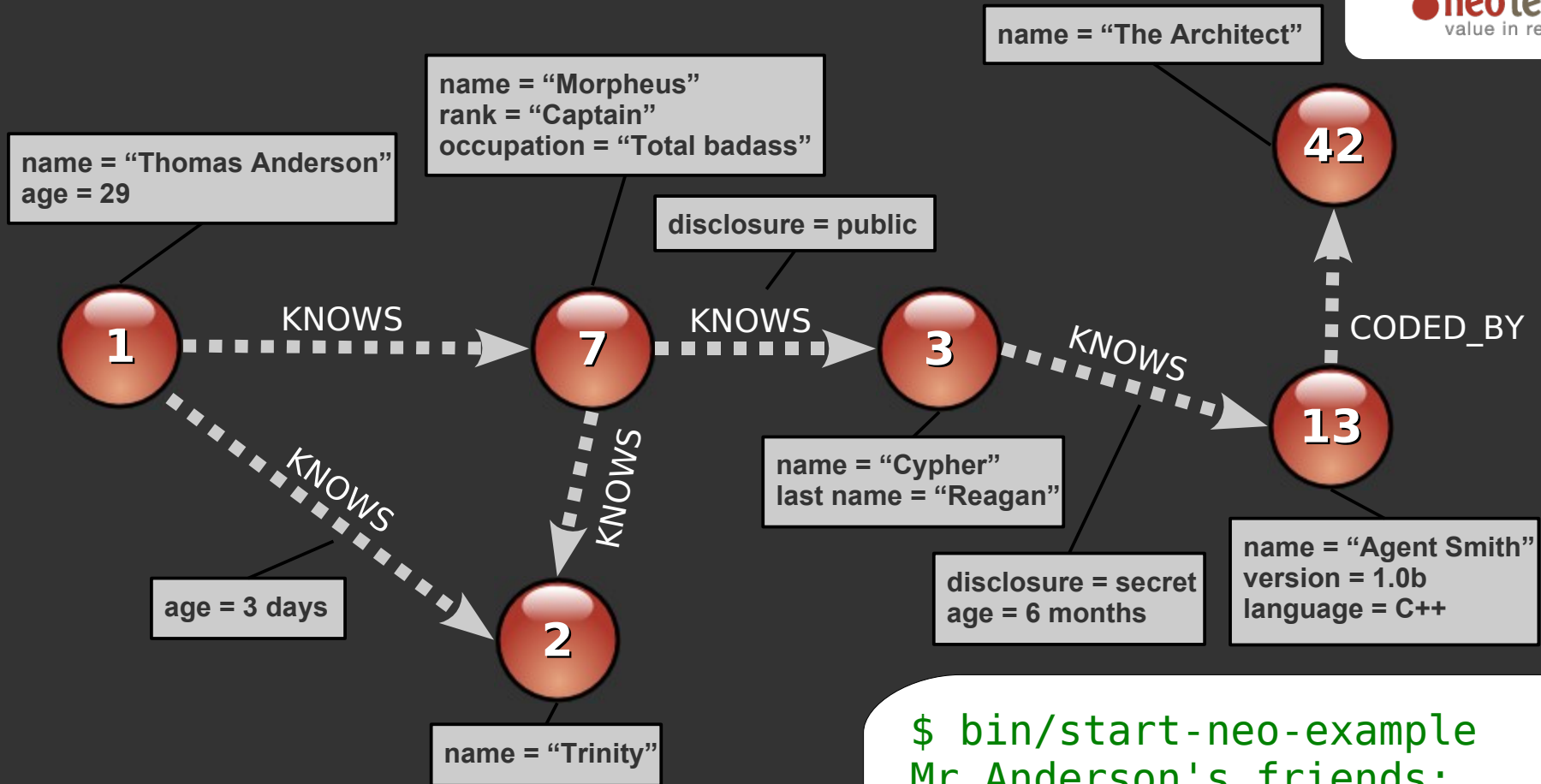
Example: Mr Anderson's friends



Code (2): Traversing a node space

```
// Instantiate a traverser that returns Mr Anderson's friends
Traverser friendsTraverser = mrAnderson.traverse(
    Traverser.Order.BREADTH_FIRST,
    StopEvaluator.END_OF_GRAPH,
    ReturnableEvaluator.ALL_BUT_START_NODE,
    RelTypes.KNOWS,
    Direction.OUTGOING );

// Traverse the node space and print out the result
System.out.println( "Mr Anderson's friends:" );
for ( Node friend : friendsTraverser )
{
    System.out.printf( "At depth %d => %s%n",
        friendsTraverser.currentPosition().getDepth(),
        friend.getProperty( "name" ) );
}
```



```

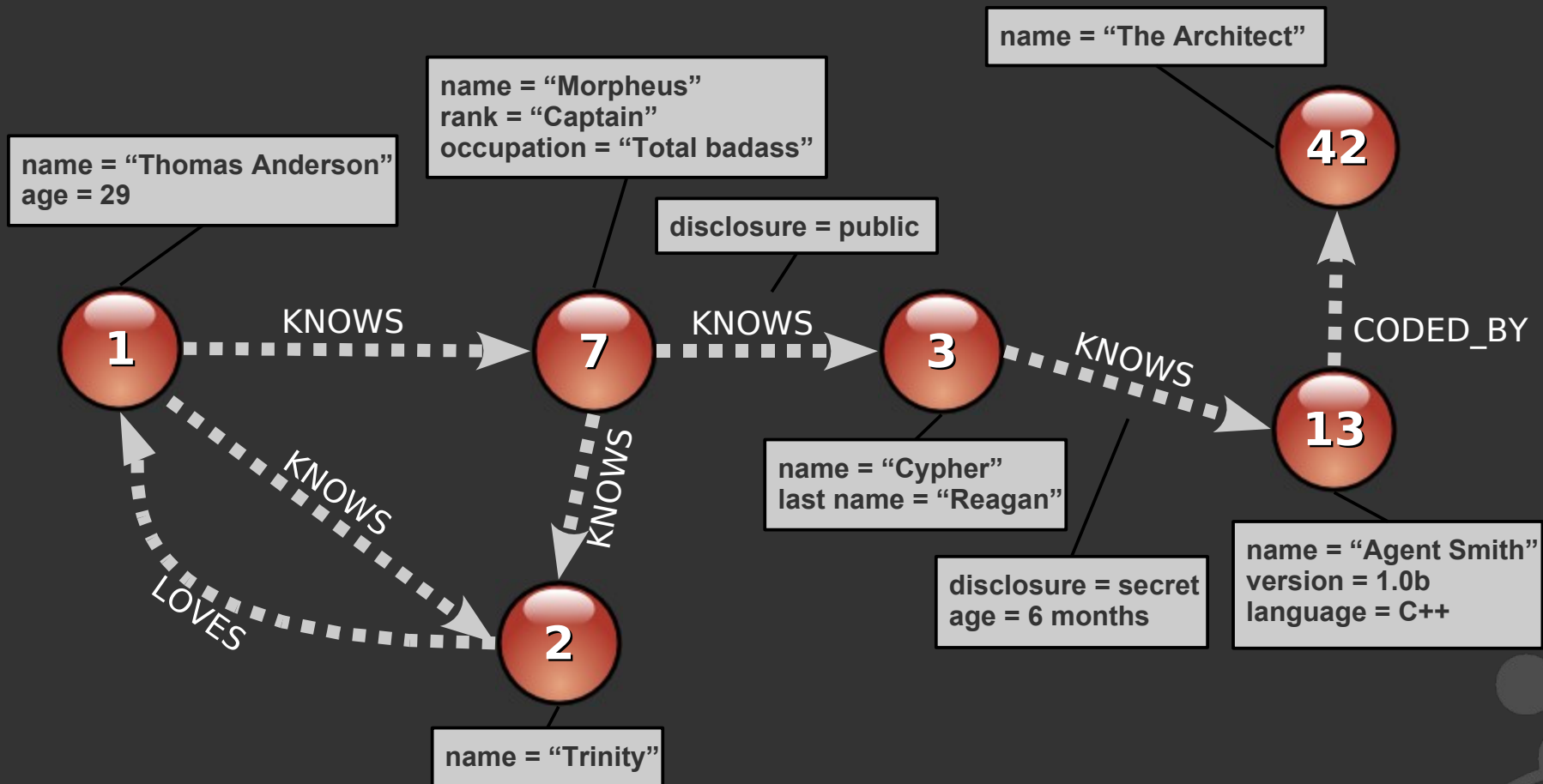
friendsTraverser = mrAnderson.traverse(
  Traverser.Order.BREADTH_FIRST,
  StopEvaluator.END_OF_GRAPH,
  ReturnableEvaluator.ALL_BUT_START_NODE,
  RelTypes.KNOWS,
  Direction.OUTGOING );
  
```

```

$ bin/start-neo-example
Mr Anderson's friends:

At depth 1 => Morpheus
At depth 1 => Trinity
At depth 2 => Cypher
At depth 3 => Agent Smith
$
  
```

Example: Friends in love?

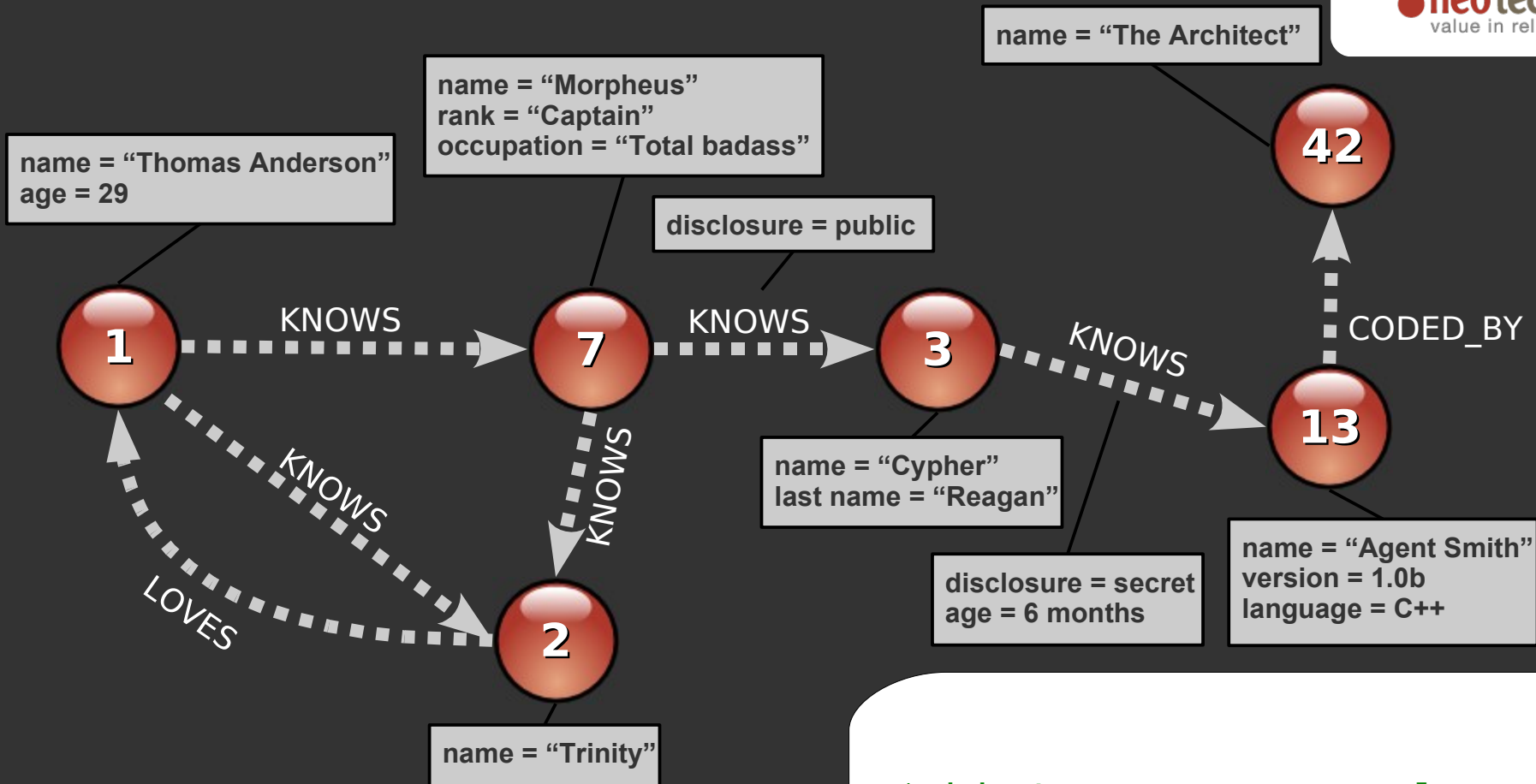


Code (3a): Custom traverser

```
// Create a traverser that returns all "friends in love"
Traverser loveTraverser = mrAnderson.traverse(
    Traverser.Order.BREADTH_FIRST,
    StopEvaluator.END_OF_GRAPH,
    new ReturnableEvaluator()
    {
        public boolean isReturnableNode( TraversalPosition pos )
        {
            return pos.currentNode().hasRelationship(
                RelTypes.LOVES, Direction.OUTGOING );
        }
    },
    RelTypes.KNOWS,
    Direction.OUTGOING );
```


Code (3a): Custom traverser

```
// Traverse the node space and print out the result  
System.out.println( "Who's a lover?" );  
for ( Node person : loveTraverser )  
{  
    System.out.printf( "At depth %d => %s%n",  
        loveTraverser.currentPosition().getDepth(),  
        person.getProperty( "name" ) );  
}
```



```

new ReturnableEvaluator()
{
  public boolean isReturnableNode(
    TraversalPosition pos)
  {
    return pos.currentNode().
      hasRelationship( RelTypes.LOVES,
        Direction.OUTGOING );
  }
},

```

```

$ bin/start-neo-example
Who's a lover?

```

```

At depth 1 => Trinity
$

```

Bonus code: domain model

- ◎ How do you implement your domain model?
- ◎ Use the delegator pattern, i.e. every domain entity wraps a Neo4j primitive:

```
// In package org.yourdomain.yourapp
class PersonImpl implements Person
{
    private final Node underlyingNode;
    PersonImpl( Node node ){ this.underlyingNode = node; }

    public String getName()
    {
        return this.underlyingNode.getProperty( "name" );
    }
    public void setName( String name )
    {
        this.underlyingNode.setProperty( "name", name );
    }
}
```

Domain layer frameworks

◎ Qi4j (www.qi4j.org)



- Framework for doing DDD in pure Java5
- Defines Entities / Associations / Properties
 - Sound familiar? Nodes / Rel's / Properties!
- Neo4j is an “EntityStore” backend

◎ NeoWeaver (<http://components.neo4j.org/neo-weaver>)

- Weaves Neo4j-backed persistence into domain objects in runtime (dynamic proxy / cglib based)
- Veeeery alpha

Neo4j system characteristics

◎ Disk-based

- Native graph storage engine with custom (“SSD-ready”) binary on-disk format

◎ Transactional

- JTA/JTS, XA, 2PC, Tx recovery, deadlock detection, etc

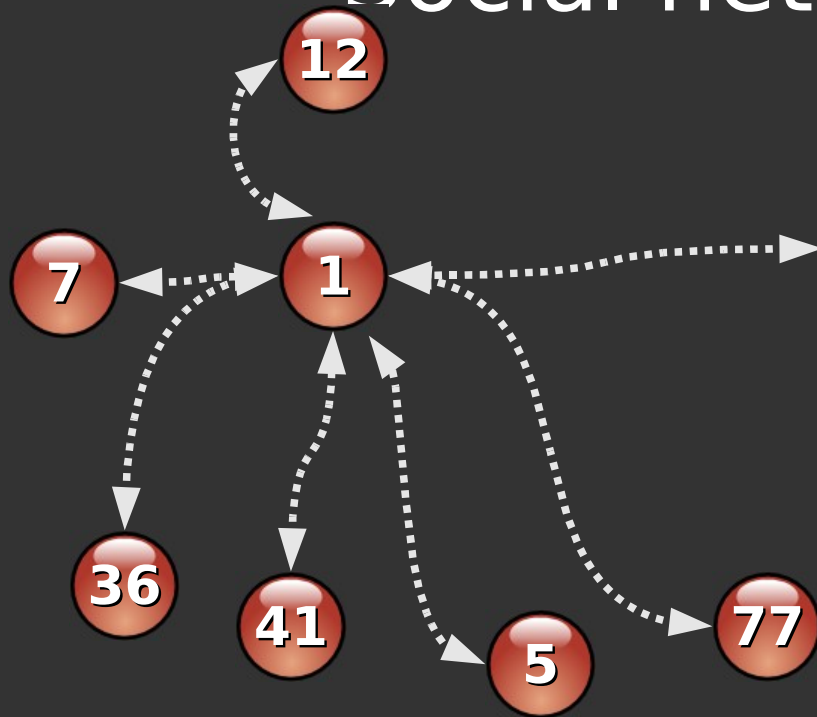
◎ Scalable

- Several billions of nodes/rels/props on single JVM

◎ Robust

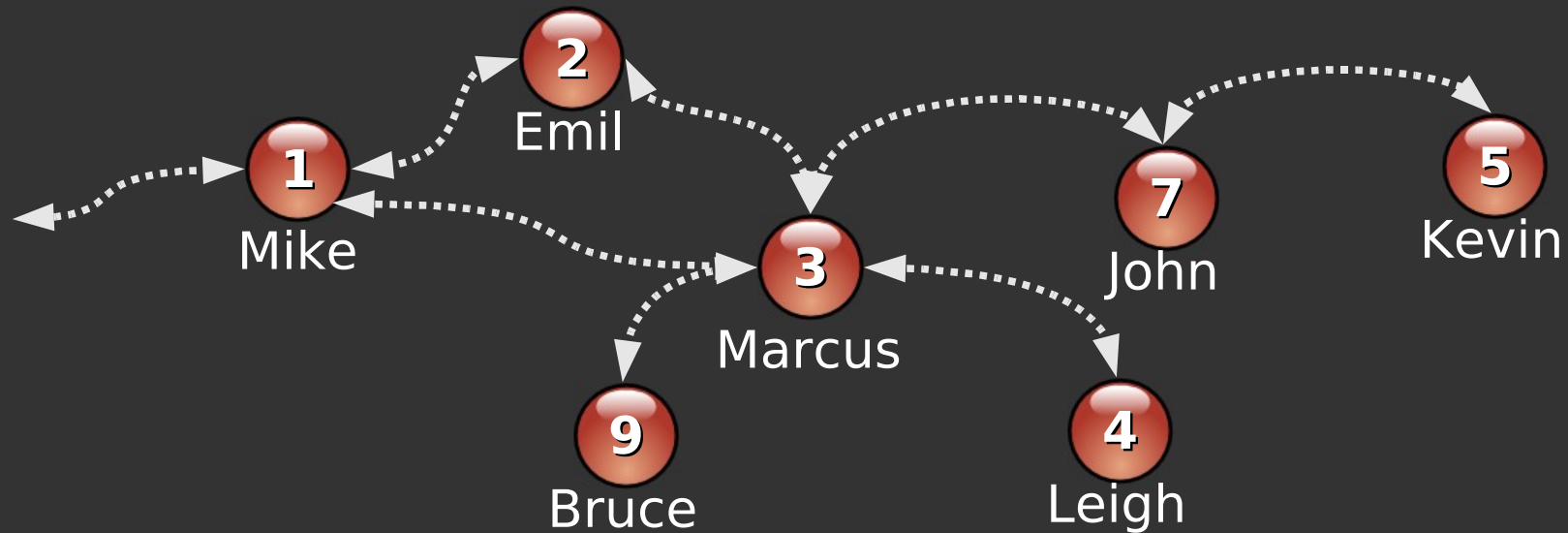
- 6+ years in 24/7 production

Social network *pathExists()*



- ~1k persons
- Avg 50 friends per person
- *pathExists(a, b)* limit depth 4
- Two backends
- Eliminate disk IO so warm up caches

Social network *pathExists()*



Relational database
Graph database (Neo4j)
Graph database (Neo4j)

persons *query time*



Got neo4j to do a do a lookup in 2 seconds, that sql server did in 45 minutes. neo4j rocks!



6:28 AM Jun 30th from web



turboCodr

John Conwell





Home Profile Find People Settings Help Sign out

Getting 40Mb/s write speeds out of Neo4J+Lucene, go Neo go!



9:12 PM Jun 9th from twhirl



peepwl
peepwl.com



View Twitter in: Standard | Mobile

© 2009 Twitter About Us Contact Blog Status Apps API Search Help Jobs Terms Privacy



Pros & Cons compared to RDBMS

- + No O/R impedance mismatch (*whiteboard friendly*)
- + Can easily evolve schemas
- + Can represent semi-structured info
- + Can represent graphs/networks (*with* performance)
- Lacks in tool and framework support
- Few other implementations => potential lock in
- + ~~No support for ad-hoc queries~~

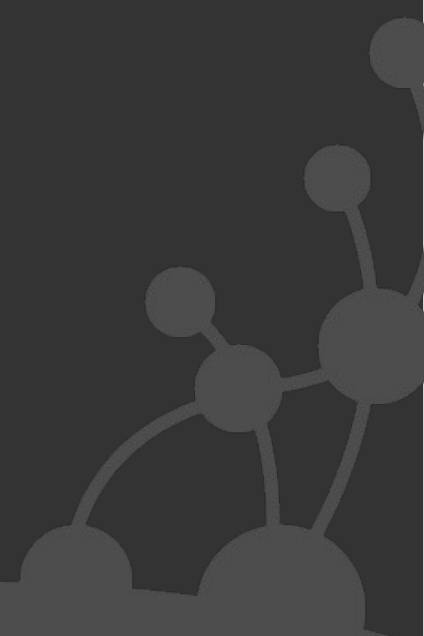
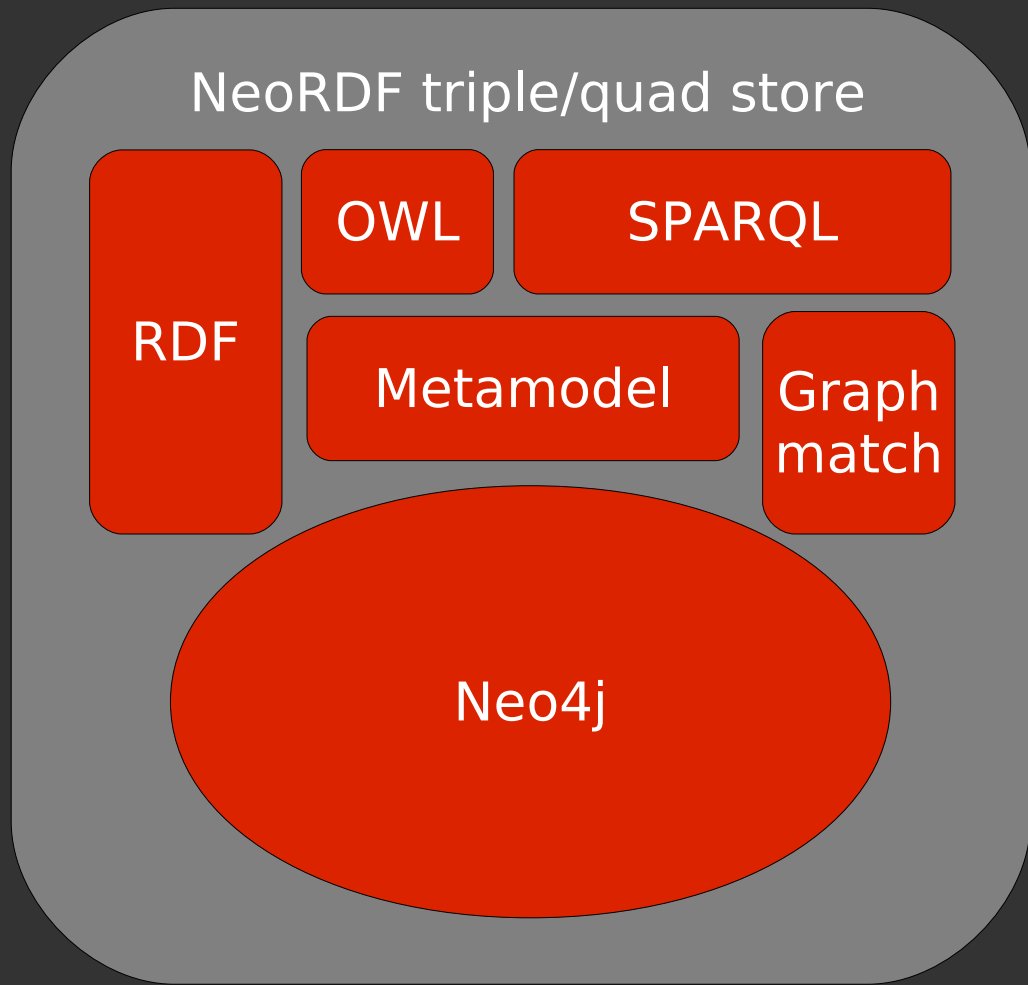
More consequences

- ◎ Ability to capture semi-structured information
 - => allowing individualization of content
- ◎ No predefined schema
 - => easier to evolve model
 - => can capture ad-hoc relationships
- ◎ Can capture non-normative relations
 - => easy to model specific links to specific sets
- ◎ All state is kept in transactional memory
 - => improves application concurrency

The Neo4j ecosystem

- ◎ Neo4j is an embedded database
 - Tiny teeny lil jar file
- ◎ Component ecosystem
 - index-util
 - neo-meta
 - neo-utils
 - owl2neo
 - sparql-engine
 - ...
- ◎ See <http://components.neo4j.org>

Example: NeoRDF



Language bindings

- ◎ Neo4j.py – bindings for Jython and CPython
 - <http://components.neo4j.org/neo4j.py>
- ◎ Neo4jrb – bindings for JRuby (incl RESTful API)
 - <http://wiki.neo4j.org/content/Ruby>
- ◎ Clojure
 - <http://wiki.neo4j.org/content/Clojure>
- ◎ Scala (incl RESTful API)
 - <http://wiki.neo4j.org/content/Scala>
- ◎NET? Erlang?



Integration of Neo4j in Grails



☆☆☆☆☆
(0 Ratings)

AUTHOR(S):

Stefan Armbruster

CURRENT RELEASE:

0.1

GRAILS VERSION:

1.1.1 > *

TAGS

neo4j - persistence +

[Fisheye](#) [Docs](#) [Edit Plugin](#)

Installation

Description

Faq

Screenshots

[Edit](#) [View Info](#)

The plugin's goal is to provide an alternative approach for storing Grails domain classes: in the Neo4j database.

Neo4j is a relative new and very interesting approach for persistence in a non-SQLish way. Neo4j is a graph database and uses the concept of

Nodes

A node is the basic building block. It normally represents a "something", a entity.

Relationships

1. Create your sample application:

```
grails create-app neo4jtest; cd neo4jtest
```

2. Remove the hibernate plugin:

```
grails uninstall-plugin hibernate
```

3. Add the Neo4j plugin:

```
grails install-plugin neo4j
```

4. create some sample domain classes:

```
grails create-domain-class Author
grails create-domain-class Book
```

5. create a controller for the domain class

```
grails create-controller Author
grails create-controller Book
```

6. modify the domain classes:

```
class Author {
    String name
    Date dob

    static hasMany = [ books: Book ]
}
```

and

```
class Book {
    String title
    static belongsTo = [author:Author]
}
```

7. modify the controller to use dynamic scaffolding:

```
class AuthorController {
    def scaffold = true
}
```

```
class BookController {
    def scaffold = true
}
```

8. start up the application:

```
grails run-app
```

9. use it, love it: go to <http://localhost:8080/neo4jtest>, add some authors and books.

10. to explore the Neo4j node space created with your grails app, check out [Neoclipse](#).

- [Gnome and CTI \(computer-telephone-integration\)](#)

Tag Cloud

cti gnome **grails** linux **neo4j** telephone
ubuntu

Blogroll

- [GroovyBlogs](#)

Categories

- [Uncategorized](#)

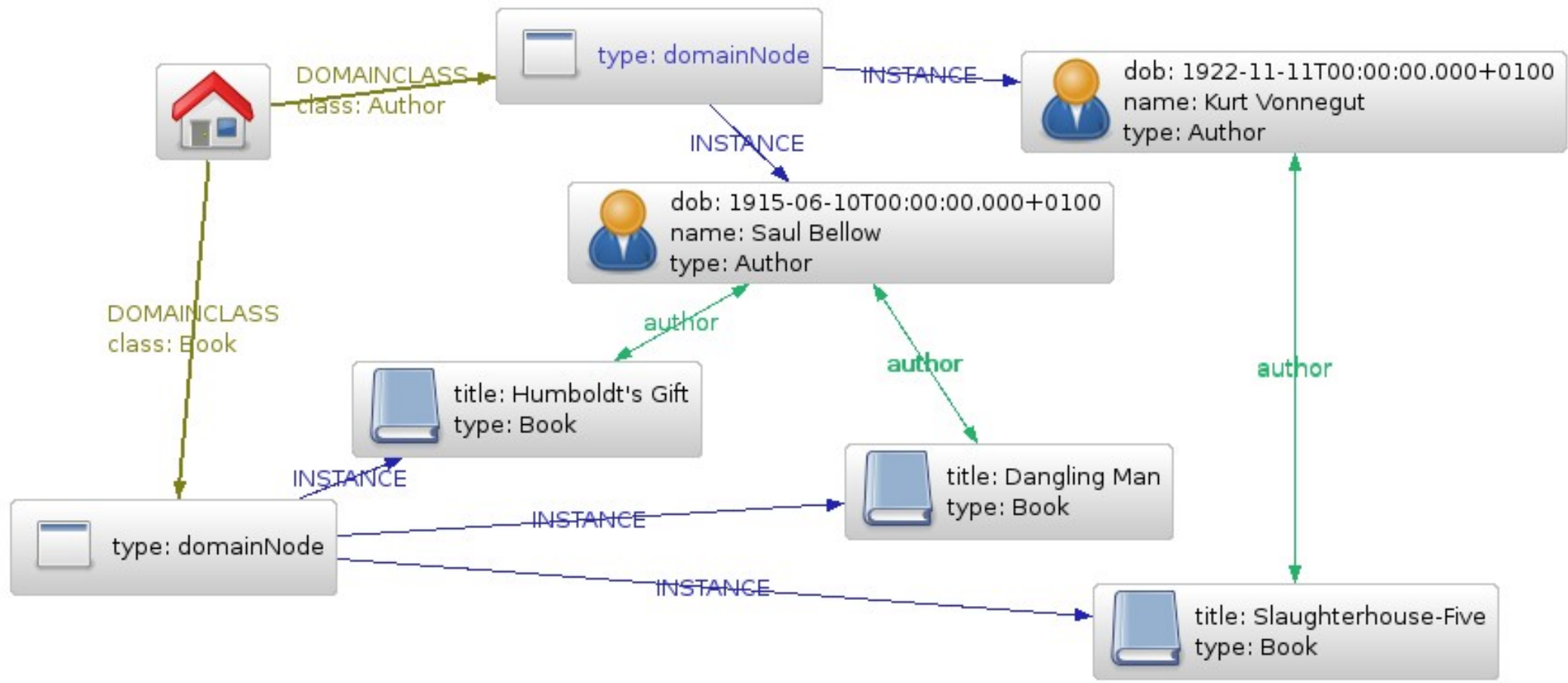
Archives

- [October 2009](#)
- [September 2009](#)

Meta

- [Log in](#)

Grails Neoclipse screendump




Scale out – replication

- ◎ Rolling out Neo4j HA before end-of-year
 - Side note: ppl roll it today w/ REST frontends & onlinebackup
- ◎ Master-slave replication, 1st configuration
 - MySQL style... ish
 - Except all instances can write, synchronously between writing slave & master (strict consistency)
 - Updates are asynchronously propagated to the other slaves (eventual consistency)
- ◎ This can handle billions of entities...
- ◎ ... but not 100B

Scale out – partitioning

◎ Sharding possible today

- ... but you have to do a lot of manual work
- ... just as with MySQL
- Great option: shard on top of resilient, scalable OSS app server  Newton, see: www.codecauldron.org

◎ Transparent partitioning? Neo4j 2.0

- 100B? Easy to say. Sliiiiightly harder to do.
- Fundamentals: BASE & eventual consistency
- Generic clustering algorithm as base case, but give lots of knobs for developers

How ego are you? (aka other impls?)

- ◎ Franz' **AllegroGraph** (<http://agraph.franz.com>)
 - Proprietary, Lisp, RDF-oriented but real graphdb
- ◎ FreeBase **graphd** (<http://bit.ly/13VITB>)
 - In-house at Metaweb
- ◎ **Kloudshare** (<http://kloudshare.com>)
 - Graph database in the cloud, still stealth mode
- ◎ Google **Pregel** (<http://bit.ly/dP9IP>)
 - We are oh-so-secret
- ◎ Some academic papers from ~10 years ago
 - $G = \{V, E\}$ #FAIL

Conclusion

- ◎ Graphs & Neo4j => teh awesome!
- ◎ Available NOW under AGPLv3 / commercial license
 - AGPLv3: “if you’re open source, we’re open source”
 - If you have proprietary software? Must buy a commercial license
 - But up to 1M primitives it’s free for all uses!
- ◎ Download
 - <http://neo4j.org>
- ◎ Feedback
 - <http://lists.neo4j.org>

: julianbrowne **twitter**

Home Profile Find People Settings Help Sign out

Going to play with Neo4j this w/e.
Seems to me that even after arguments
about ACID/scale/CAP it's just more
human & agile to be graph-based



5:42 PM Jul 3rd from web



julianbrowne



Questions?



Image credit: lost again! Sorry :(



<http://neotechnology.com>