Introduction to Graph Databases

Thomas C Bourgeois

28 October 2014
Introduction

- Presents and overview of graphs and graph databases
- Focus on ‘why?’ and ‘how?’ (rather than on any specific ‘what?’)
- Examples
Graphs
Graph.
Graph?
Graph!
Graphs… Have Nodes
And Graphs...

Have Edges
In a Directed Graph, Edges Have Sources and Targets
Graphs Can Convey Relationships

A \rightarrow \text{KNOWS}\rightarrow C \rightarrow \text{KNOWS}\rightarrow D \rightarrow \text{KNOWS}\rightarrow E \rightarrow \text{KNOWS}\rightarrow F

A \rightarrow \text{KNOWS}\rightarrow B

A \rightarrow \text{KNOWS}\rightarrow E

Or Process Flows,
Or Paths Through Mazes,
Or Mazes Themselves.
Graph Databases
Relational Databases...

Use Tables to Hold Data

<table>
<thead>
<tr>
<th>HOUSEHOLD</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NAME</td>
<td>EYE_COLOR</td>
<td>HEIGHT</td>
<td>AGE</td>
</tr>
<tr>
<td>1</td>
<td>Tom</td>
<td>Hazel</td>
<td>6.0</td>
<td>49</td>
</tr>
<tr>
<td>2</td>
<td>Shawn</td>
<td>Brown</td>
<td>5.2</td>
<td>null</td>
</tr>
<tr>
<td>3</td>
<td>Ling</td>
<td>Yellow</td>
<td>1</td>
<td>13</td>
</tr>
<tr>
<td>4</td>
<td>Muffin</td>
<td>Green</td>
<td>1</td>
<td>13</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SITS_ON</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>ID</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
</tr>
</tbody>
</table>
Relational Databases Operate…

On Data *Sets*

- Intersection
- Union
- Overlap
- Exclusion
SQL: Structured Query Language...

```
SELECT H1.NAME AS source, H2.NAME AS target
FROM HOUSEHOLD AS H2
INNER JOIN
  (HOUSEHOLD AS H1 INNER JOIN
   SITS_ON AS S ON H1.ID = S.SOURCE)
ON H2.ID = S.TARGET
```
Graph Databases...

Use Graphs to Hold Data

Use Graphs to Hold Data
Graph Databases Operate...

On Data *Relationships*
MATCH (n:Person), (m: Animal)
WHERE (m)-[:SITS_ON]->(n)
RETURN m, n
The Neo4J Graph Database

- Available from neo4J.com
- Open Source Community Edition for Windows, MacOS & Linux
- Java Based (Oracle JDK7 / OpenJDK7)
- REST and html interfaces
The Neo4J Data Model

Nodes

Relationships
The Neo4J Data Model

Labels and Properties

Animal:
{name: ‘Ling’,
color: ‘Black’,
type: ‘cat’}

Person:
{name: ‘Tom’,
age: 49,
eyes: ‘hazel’}

SITS_ON
{frequency: ‘always’}
Neo4J Interface and Examples
Classroom Data Model

Person: {name: 'A', student:true}

Person: {name: '00', teacher:true}

Person: {name: 'Tom', age: 49, eyes: 'hazel'}

KNOWS

LIKES
Cypher: Create a Node

- CREATE (n:Person {name:'A', student:true})
  RETURN (n)

- This will create a node object and return the object
Classroom (REST)

```json
{"statements":[
{"statement":"CREATE (n:Person {nodes}) RETURN n",
"parameters":
{"nodes":[
{"name":"00","teacher":true,"x":0,"y":40},
{"name":"A","student":true,"x":120,"y":40},
{"name":"B","student":true,"x":240,"y":40},
]}
]}
```
Cypher: Create a Relationship

- MATCH (m:Person {name: '00'}), (n:Person {name: 'A'}) CREATE (m)-[r:KNOWS]->(n) RETURN m.name, n.name, type(r)

- Finds two specific nodes (m), (n) and create a KNOWS relationship (r) between them and returns some details
Relationships (REST)

```json
{"statements": [
    {
      "statement": "MATCH (m:Person {name: '00'}) , (n:Person {name: 'A'})  CREATE (m)-[r:KNOWS]->(n) RETURN m.name, n.name, type(r)"
    },
    {
      "statement": "MATCH (m:Person {name: '00'}) , (n:Person {name: 'B'})  CREATE (m)-[r:KNOWS]->(n) RETURN m.name, n.name, type(r)"
    },
    ...
]}
```
Adjacency Matrix

- KNOWS is red
- LIKES is blue
- Source: row
- Target: column
Some Interesting Queries

- ‘a’ knows ‘b’ who knows ‘c’ (that ‘a’ does not know)

MATCH (a  { name: 'A' })-[[:KNOWS*2..2]-
(friend_of_friend)
WHERE NOT (a)-[:KNOWS]-(friend_of_friend)
RETURN friend_of_friend.name, COUNT(*)
ORDER BY COUNT(*) DESC , friend_of_friend.name
Paths Between a and b

- MATCH (a:Person { name:"M" }), (o:Person { name:"O" }),
  p = shortestPath((a)-[*..15]-(o))
  RETURN p

- MATCH (r { name:“R” }), (z { name:“Z” }),
  p = allShortestPaths((r)-[*]-(z))
  RETURN p
So There’s That.

- Neo4J models connected data
- Answers ‘Path-Oriented’ Questions
- Supports Modern I/O
- Open Source Community Edition
Connections To…

- Visualization
- Big Data
- ‘Differently Structured’ Data
- ‘Differently Structured’ Questions about Data
Questions?