Graph Database, think different!

- Nodes
- Edges (directed or not)
- Properties
“We found Neo4j to be literally thousands of times faster than our prior MySQL solution, with queries that require 10-100 times less code. Today, Neo4j provides eBay with functionality that was previously impossible.”

Volker Pacher, Senior Developer, eBay
Neo4j

• Graph database
  (Like SQL server e.g. PostgreSQL, MySQL)
• Implemented in Java

and

Cypher

• Graph query language for Neo4J
  (Like SQL)
• Declarative
First query

Get all nodes of type *Program* that have the name *Hello World!*

MATCH (a : Program)
WHERE a.name = ‘Hello World!’
RETURN a
Query relationships

Get all relationships of type *Author* connecting *Programmers* and *Programs*:

MATCH (a : Programmer) -[r : Author]-(b : Program)
RETURN r
Matching nodes and relationships

Nodes:

(a), (), (:Ntype), (a:Ntype), (a { prop: ‘value’ } ),
(a:Ntype { prop: ‘value’ } )

- Relationships:

(a)--> (b), (a)--> (b), (a)<-- (b), (a)--> (), (a)-[r]--> (b),
(a)-[:Rtype]--> (b), (a)-[[:R1] | :R2]--> (b),
(a)-[r:Rtype]--> (b)

- May have more then 2 nodes:

(a)--> (b)<-- (c), (a)--> (b)--> (c)

- Path:

p = (a)-->(b)
More options:

• Relationship distance:
  (a)-[:Rtype*2]->(b) – 2 hops of type Rtype.
  (a)-[:Rtype*]-(b) – any number of hops of type Rtype.
  (a)-[:Rtype*2..10]-> (b) – 2-10 hops of Rtype.
  (a)-[:Rtype*..10]-> (b) – 1-10 hops of Rtype.
  (a)-[:Rtype*2..]-> (b) – at least 2 hops of Rtype.

Could be used also as:
(a)-[r*2]->(b) – r gets a sequence of relationships
(a)-[*{prop:val}]->(b)
Operators

• Mathematical
  +, -, *, /, %, ^ (power, not XOR)

• Comparison
  =, <>, <=, >=, =~ (Regex), IS NULL, IS NOT NULL

• Boolean
  AND, OR, XOR, NOT

• String
  Concatenation through +

• Collection
  Concatenation through +
  IN to check if an element exists in a collection.
More WHERE options

- WHERE others.name IN ['Andres', 'Peter']
- WHERE user.age IN range (18,30)
- WHERE n.name =~ 'Tob.*'
- WHERE n.name =~ '(?i)ANDR.*' - (case insensitive)
- WHERE (tobias)-->()
- WHERE NOT (tobias)-->()
- WHERE has(b.name)
- WHERE b.name? = 'Bob'
  (Returns all nodes where name = 'Bob' plus all nodes without a name property)
Functions:

• On paths:
  • MATCH shortestPath( (a)-[*]-(b) )
  • MATCH allShorestPath( (a)-[*]-(b) )
  • Length(path) – The path length or 0 if not exists.
  • RETURN relationships(p) - Returns all relationships in a path.

• On collections:
  • RETURN a.array, filter(x IN a.array WHERE length(x)= 3)
    FILTER - returns the elements in a collection that comply to a predicate.
  • WHERE ANY (x IN a.array WHERE x = "one“ ) – at least one
  • WHERE ALL (x IN nodes(p) WHERE x.age > 30) – all elements
  • WHERE SINGLE (x IN nodes(p) WHERE var.eyes = "blue") – Only one
    * nodes(p) – nodes of the path p
A general query structure

MATCH [Nodes and relationships]
WHERE [Boolean filter statement]
RETURN [DISTINCT] [statements [AS alias]]
ORDER BY [Properties] [ASC\DESC]
SKIP [Number] LIMIT [Number]
With

• Manipulate the result sequence before it is passed on to the following query parts.

• Usage of WITH:
  • Limit the number of entries that are then passed on to other MATCH clauses.
  • Introduce aggregates which can then be used in predicates in WHERE.
  • Separate reading from updating of the graph. Every part of a query must be either read-only or write-only.
With

MATCH (david { name: "David" })--(otherPerson)--() 
WITH otherPerson, count(*) AS foaf 
WHERE foaf > 1 
RETURN otherPerson

What will be returned?

The person connected to David with the at least more than one outgoing relationship will be returned by the query.

(2 {name:"Anders"}) * foaf = 2
More collections options

- MATCH (user)
  RETURN count(user)
- MATCH (user)
  RETURN count(DISTINCT user.name)
- MATCH (user)
  RETURN collect(user.name)
  Collection from the values, ignores NULL.
- MATCH (user)
  RETURN avg(user.age)
  Average numerical values. Similar functions are sum, min, max.
Adding and manipulating data

In the lecture
Demo time
Vintage tutorials questions

dname dcity

frequents

drinker pub

likes serves

bname btype

pname pcity
Vintage tutorials questions

• באולו ערים מגישים בירה שיווסי אוהב?

MATCH (:drinker { dname: "Yossi" })-[:likes]->(:beer)<-[:serves]-(p : pub)
RETURN p.pcity

• מי הם השתיינים שבכל פאב (המגיש בירה) שיש להם בירה אוהב?

MATCH (p : pub)
WITH collect(p) as Pubs
MATCH (d : drinker)
WHERE ALL (p in Pubs WHERE
 (p)-[:servers]->(:beer)<-[:likes]-(d) )
RETURN d
Exam questions - Win13-14 A

<table>
<thead>
<tr>
<th>Student</th>
<th>Lecturer</th>
<th>Course</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name</td>
<td>Name</td>
</tr>
<tr>
<td>ID</td>
<td>ID</td>
<td>Catalogue_Number</td>
</tr>
<tr>
<td>Address</td>
<td></td>
<td>Syllabus</td>
</tr>
</tbody>
</table>

A. Write a Cypher query stating:

Shave all students who studied all the courses.

MATCH (c:Course)
WITH collect(c) AS courses
MATCH (s:Student)
WHERE ALL (x in courses WHERE (s)-[:Studies]->(x))
RETURN s.name
`MATCH (s:Student)-[:Studies*2..4]->(:Student{Name:"Roy"})
RETURN DISTINCT s.name`
Exam questions - Win13-14 B

<table>
<thead>
<tr>
<th>Student</th>
<th>Lecturer</th>
<th>Course</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name</td>
<td>Name</td>
</tr>
<tr>
<td>ID</td>
<td>ID</td>
<td>Catalogue_Number</td>
</tr>
<tr>
<td>Address</td>
<td></td>
<td>Syllabus</td>
</tr>
</tbody>
</table>

grade-1 semester ומכיל את התכונות Course בין.Student מחבר בין Lecturer ומכיל את התכונות Course.

classroom-1 semester ומכיל את התכונות Course בין.Lecturer מחבר בין Course ומכיל את התכונות Course.

מנמר פונקציית מרחק בין שני סטודנטים שניים לכלكم:

1. המסורטנים A ו-B הם במרחק 1 אם הם למדו קורס משותף.
2. המסורטנים A ו-B הם במרחק 1=א אם ה- הו המספר הקטן ביותר ביוורכר כר שיקים סוטודנטים C
3. אם לא קיימים מredi sistema המרחק لإזיוות 0.

כתבו שאילתה Cypher שמתוחזר:

```
MATCH p=shortestPath((s1:Student {ID:'12345'})-[Studie*:]- (s2:Student {ID:'67890'}))
RETURN length(p)/2
```
MATCH (l:Lecturer)-[:Teaches]->(c:Course)
WITH l, count(c) as numcourses
WHERE numcourses >= 3
RETURN l.name
Learn more...

Check Neo4j online version:

http://console.neo4j.org/
Learn more...

Download Neo4j for free:

http://neo4j.com/download/
Learn more...

Read the Neo4j manual:

http://neo4j.com/docs/stable/

Cypher tutorials:

http://neo4j.com/developer/cypher-query-language/

More Neo4j developers tutorials:

http://neo4j.com/developer/get-started/