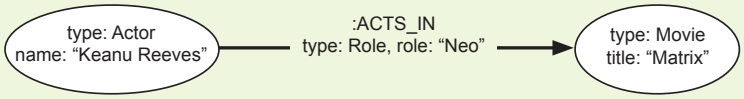
springdataNeo4j Developer Notes

Neo4j - `[IS_A]` -> Property Graph

Nodes with key-value properties are connected by typed, directed **relationships** with properties.



Cypher is a query language:

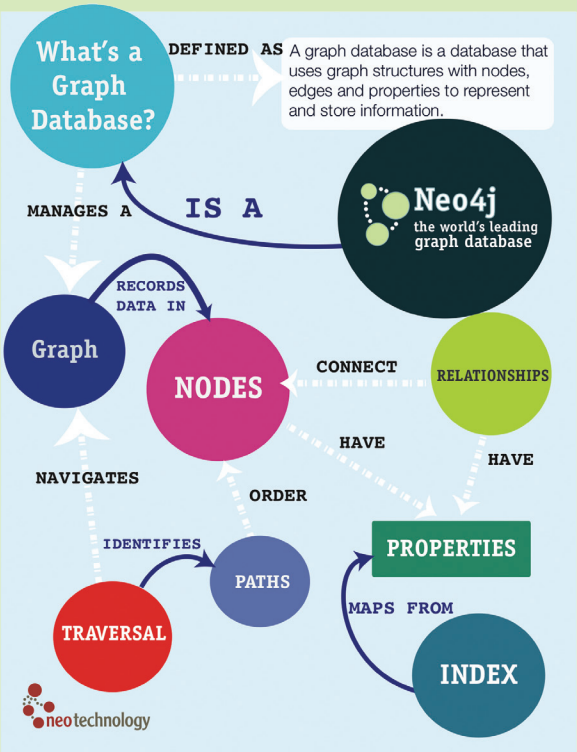
```
START movie=(id)
    MATCH movie<-[:ACTS_IN]-actor
    WHERE actor.age > 20
    RETURN movie, actor.name
    ORDER by movie.title
```

Traversals walk the graph in the prescribed way

```
TraversalDescription actors = Traversal.description()
    .relationships(“ACTS_IN”)
    .filter(Traversal.returnAllButStartNode());
for (Node actors : actors.traverse(movie))
 …
```

Transactional: TX needed for write operations

```
Transaction tx=graphDatabaseService.beginTx();
try {
  … graph operations …
  tx.success();
} catch(Exception e) {
  tx.failure();
} finally {
  tx.finish();
}
```



Example

```
@NodeEntity
public class Movie {
    @GraphId Long id;
    @Indexed(fulltext = true, indexName = “search”)
    String title;
    Person director;
    @RelatedTo(type=”ACTS_IN”, direction = INCOMING)
    Set<Person> actors;
    @RelatedToVia(type = “RATED”)
    Iterable<Rating> ratings;
    @Query(“START movie=node({self})
            MATCH movie-->genre<--similar
            RETURN similar”)
    Iterable<Movie> similarMovies;
}
```

Neo4jTemplate

- convenience methods for graph operations, transaction and exception handling
- index lookups, cypher queries, traversal execution with fluent result converter API
- preconfigured and customizable conversion of results to a variety of types

Spring-Config

```
<beans ...
 xmlns:neo4j=http://www.swf.org/schema/data/neo4j
 xsi:schemaLocation=”http://www.sfw.org/schema/data/neo4j”...>
...
<neo4j:config store-dir=”graph.db”/>
<neo4j:repositories base-package=”com.example.repositories”/>
...
</beans>
```

Testing

```
<neo4j:config graphDatabaseService=”gds”/>
<bean id=”gds” class=”...ImpermanentGraphDatabase/>
```

REST

```
<neo4j:config graphDatabaseService=”rds”/>
<bean id=”rds” class=”...SpringRestGraphDatabase/>
```

Book

Good Relationships – The Spring Data Neo4j Guide Book
by Michael Hungerwith Andreas Kolleger and David Montag

With Spring Data, the ever popular Spring Framework has cultivated a new patch of ground, bringing Big Data and NOSQL technology like Neo4j to enterprise developers. This guide introduces you to Spring Data Neo4j, using the fast, powerful and scalable graph database Neo4j to enjoy the benefits of having good relationships in your data.

www.neotechnology.com/2012/01/good-relationships/

More books are available at Neo Technology’s Book Store:
www.neotechnology.com/bookstore/

springdataNeo4j Developer Notes

Repositories

```
interface MovieRepository extends
 GraphRepository<Movie> {
  @Query("START movie={0}
          MATCH m<-[rating:RATED]-user
          RETURN rating")
  Iterable<Rating> getRatings(Movie movie);
  // Co-Actors
  Iterable<Person> findByActorsMoviesActorName(name)
}

<neo4j:repositories base-package="com.example.dao"/>
```

- Provide CRUD, Finder, Query and Traversal abilities by default.
- Automatic Finders can be derived from method names
- @Query annotations.
- Custom method implementations possible

Book

***Spring Data* by Jon Brisbin, Oliver Gierke, Thomas Risberg, Mark Pollack and Michael Hunger**

This book shows you how Spring's data access framework can help you connect to either non-relational or relational databases, or a combination of the two.

www.neotechnology.com/2012/01/spring-data/

AspectJ - STS / eclipse setup

- any Spring Tool Suite version
- import existing maven project or set aspectj nature
- Project Properties -> AspectJ -> Aspect Paths
- AspectJ Configurator installed
- sometimes build after restart required

Entity – Annotations

- `@NodeEntity`
  - o marks a domain class as backed by a node in the graph
- `@RelationshipEntity`
  - o marks a domain class as backed by a relationship in the graph
- `@GraphId`
  - o for id of graph-entity
- `@Indexed([unique=true],[indexType=Type.FULLTEXT],[indexName=name])`
  - o automatic property-index, unique entity creation
- `@RelatedTo([direction=Direction],[type=TYPE])`
  - o define type and direction of relationship, used on reference fields, single or collections
- `@RelatedToVia([direction=Direction],[type=TYPE])`
  - o like `@RelatedTo`, but targets Relationship-Entities
- `@Query(query,[params=key,value,...],[type=QueryType])`
  - o computed field, executes graph query in Cypher
- `@GraphTraversal(traversal=~TraversalBuilder.class, [params=key,value,...])`
  - o computed field, executes traversal starting from entity-node

Roo Add-on

```
roo> addon search graph
roo> addon install id --searchResultId 01
roo> project --topLevelPackage net.cineasts
roo> graph setup --provider NEO4J
--databaseLocation cineasts
roo> graph entity --class ~.model.Movie
roo> field string title
roo> graph entity --class ~.model.Actor
roo> field string name
roo> graph relationship --to Movie --from Actor
--fieldName movies --type ACTS_IN
--cardinality ONE_TO_MANY
roo> controller all --package ~.web
```

Maven Dependency

```
<dependency>
    <groupId>org.springframework.data</groupId>
    <artifactId>spring-data-neo4j[-aspects]</artifactId>
    <version>2.1.0.RELEASE</version>
</dependency>
```

Libraries

spring-data-neo4j
Provides Repository based Object <-> Graph Mapping that copies data from and to the graph database

spring-data-neo4j-aspects
Provides transparent property and relationship mapping inside of transactions with live read- and write-through to the graph database

spring-data-neo4j-cross-store
Allows to partially store properties and relationships of JPA entities in the graph database

spring-data-neo4j-rest
Provides transparent remote access to Neo4j-Server via its REST Api (uses org.neo4j:neo4j-rest-graphdb)

Links

http://neo4j.org
http://springsource.org/spring-data/neo4j
http://cineasts.net
http://spring.neo4j.org
http://spring.neo4j.org/guide
http://spring.neo4j.org/discussions
http://spring.neo4j.org/docs
http://spring.neo4j.org/download
http://spring.neo4j.org/issues
http://spring.neo4j.org/source

springdataNeo4j
The Best NOSQL Database for Spring