Building a real-time recommendation engine with Neo4j

OSCON 2017

William Lyon
@lyonwj
William Lyon
Developer Relations Engineer @neo4j
will@neo4j.com
@lyonwj
lyonwj.com
Agenda

- Intro to graph databases and Neo4j
  - Graph data model
  - Cypher query language
- Use cases
  - Recommender systems
- Hands-on!
  - Graph data modeling
  - Cypher
Neo4j

Graph Database

• Property graph data model
• (open)Cypher query language
• Native graph processing
• Language drivers
• Open source

eo4j.com
Graph Data Model
Labeled Property Graph Model
Labeled Property Graph Model
The Graph

https://github.com/johnymontana/neo4j-datasets/tree/master/yelp
The Graph

https://www.youtube.com/watch?v=kSZHFlBDIfM
The Graph

https://github.com/johnymontana/mattermark-graphql-neo4j
The Graph

https://github.com/neo4j-meetups/modeling-worked-example
GRAPHS

GRAPHS EVERYWHERE
openCypher

(queryLanguage) - [:FOR] -> (graphs)

http://www.opencypher.org/
1 MATCH (m:Movie)<-[[:RATED]]-(u:User)
2 WHERE m.title CONTAINS "Matrix"
3 WITH m.title AS movie, COUNT(*) AS reviews
4 RETURN movie, reviews
5 ORDER BY reviews DESC
6 LIMIT 5;
MATCH (m:Movie)<-[][:RATED]-(u:User)
WHERE m.title CONTAINS "Matrix"
WITH m.title AS movie, COUNT(*) AS reviews
RETURN movie, reviews
ORDER BY reviews DESC
LIMIT 5;

<table>
<thead>
<tr>
<th>find</th>
<th>MATCH (m:Movie)&lt;-[][:RATED]-(u:User)</th>
<th>Search for an existing graph pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter</td>
<td>WHERE m.title CONTAINS &quot;Matrix&quot;</td>
<td>Filter matching paths to only those matching a predicate</td>
</tr>
<tr>
<td>aggregate</td>
<td>WITH m.title AS movie, COUNT(*) AS reviews</td>
<td>Count number of paths matched for each movie</td>
</tr>
<tr>
<td>return</td>
<td>RETURN movie, reviews</td>
<td>Specify columns to be returned by the statement</td>
</tr>
<tr>
<td>order</td>
<td>ORDER BY reviews DESC</td>
<td>Order by number of reviews, in descending order</td>
</tr>
<tr>
<td>limit</td>
<td>LIMIT 5;</td>
<td>Only return first four records</td>
</tr>
</tbody>
</table>

https://neo4j.com/sandbox-v2/
Test-Drive Neo4j with Cypher

Impact Analysis
Find all services that depend on Server 1. These would be impacted by an outage of that server.

MATCH (service:Service) WHERE service.name = 'service 1'
RETURN service

Only Webserver VM depends on Server 1. Because we're looking at variable length paths of DependsOn relationships, we're also able to determine that Public Website would be impacted by an outage of Server 1.

Downloading and installing Python
1. Install neo4j-python-driver using pip:
   $ pip install neo4j-driver
2. Copy and paste code at left into py file and run

https://neo4j.com/developer/
Use Case: Personalized Recommendations
Recommendations drive user engagement

Recommendations for you in Kindle Store
“35 percent of what consumers purchase on Amazon and 75 percent of what they watch on Netflix come from product recommendations”

Dreamhouse Series 15% off

People who bought Side Table also bought:

- Coffee Table: $235
- Low Book Shelf: $150
- Bed Side Table: $90

Similar product in from Home Office Series:

- Wood Side Table: $110
- Green Side Table: $135
- Walnut Side Table: $120

Personalized Promotions
Personalized Real-Time Recommendations
Personalized Real-Time Recommendations
Collaborative Filtering
An algorithm that considers users interactions with products, with the assumption that other users will behave in similar ways.

Content Based
An algorithm that considers similarities between products and categories of products.

Data-Model
(Expressed as a graph)
Polyglot Persistence

Products:
- Inventory
- Location
- Category
- Price
- Configurations

Customers/Users:
- Purchase
- Return
- Review
- View
- In-store Purchases
- Location

Data Stores:
- RELATIONAL DB
  - Purchases
- DOCUMENT STORE
  - Product Catalogue
- WIDE COLUMN STORE
  - Views
- DOCUMENT STORE
  - User Review
- RELATIONAL DB
  - In-Store Purchase
- KEY VALUE STORE
  - Shopping Cart
Recommendations require an operational workload — it’s in the moment, real-time!

Good for Analytics, BI, Map Reduce
Non-Operational, Slow Queries

Data Lake

- Purchases
- Product Catalogue
- Views
- User Review
- In-Store Purchase
- Shopping Cart
Graph-based recommendations
Using Data Relationships for Recommendations

**Content-based filtering**
Recommend items based on what users have liked in the past

**Collaborative filtering**
Predict what users like based on the similarity of their behaviors, activities and preferences to others
Collaborative Filtering
Collaborative Filtering
Collaborative Filtering
In Cypher

1. MATCH (will:Person {name: "Will"})-[:PURCHASED]->(b:Book)<=[:PURCHASED]-(o:Person)
2. MATCH (o)-[:PURCHASED]->(rec:Book) WHERE NOT exists((will)-[:PURCHASED]->(rec))
3. RETURN rec
In Cypher

```
MATCH (will:Person {name: "Will"})-[[:PURCHASED]]->(b:Book)<-[:PURCHASED]-(o:Person)
MATCH (o)-[:PURCHASED]->(rec:Book) WHERE NOT exists((will)-[:PURCHASED]->(rec))
RETURN rec
```

Basic initial approach. Improvements:
- aggregate across all purchases
- scoring / normalize
- compute similarity metrics
Content Filtering

Will \text{PURCHASED} \rightarrow \text{Data Structures} \text{HAS_TAG} \rightarrow \text{Big Data}

\text{Advanced NoSQL} \text{HAS_TAG}
Content Filtering

Will PURCHASED Data Structures HAS_TAG Big Data

Purchased Advanced NoSQL HAS_TAG
Content Filtering w/ Cypher

1 MATCH (will:Person {name: "Will"})-[[:PURCHASED]]->(b:Book)<-[:HAS_TAG]-(t:Tag)
2 MATCH (t)<-[:HAS_TAG]-(other:Book) WHERE NOT exists((will)-[:PURCHASED]->(other))
3 RETURN other
Content Filtering - Concept Hierarchy

Big Data

IS_SUB_TAG_OF

Data

IS_SUB_TAG_OF

Databases
Content Filtering - Concept Hierarchy

Will PURCHASED → Data Structures HAS_TAG → Big Data

Big Data IS_SUB_TAG_OF Data

Advanced NoSQL HAS_TAG Databases

Databases IS_SUB_TAG_OF Data
Content Filtering - Concept Hierarchy

- Will
  - PURCHASED
  - PURCHASED
  - Data Structures
  - HAS_TAG
  - Advanced NoSQL
  - HAS_TAG
  - Databases
  - Big Data
  - IS_SUB_TAG_OF
  - Data
  - IS_SUB_TAG_OF
Content Filtering - Concept Hierarchy w/ Cypher

1 MATCH (will:Person {name: "Will"})-[[:PURCHASED]]-(b:Book)<=[:HAS_TAG]-(t:Tag)
2 MATCH (t)-[:IS_SUB_TAG_OF]-(Tag)<=[:IS_SUB_TAG_OF]-(ot:Tag)
3 MATCH (ot)<=[:HAS_TAG]-(other:Book) WHERE NOT exists((will)-[:PURCHASED]-(other))
4 RETURN other
Content Filtering - Concept Hierarchy w/ Cypher

1. MATCH (will:Person {name: "Will"})-[:PURCHASED]->(b:Book)<-[:HAS_TAG]-(t:Tag)
2. MATCH (t)-[:IS_SUB_TAG_OF]->(Tag)<-[:IS_SUB_TAG_OF]-(ot:Tag)
3. MATCH (ot)<-[:HAS_TAG]-(other:Book) WHERE NOT exists((will)-[:PURCHASED]->(other))
4. RETURN other

Basic initial approach. Improvements:
- aggregate across all purchases
- cold start
- variable length concept hierarchy
- tag similarity / clusters
Neo4j Sandbox

Welcome to the Neo4j Sandbox. If you have any questions or problems, feel free to reach out to us at devrel@neo4j.com.

Your Current Sandboxes

- Get Started with your Neo4j Sandbox
  1. Visit the Neo4j Browser. Log in with the credentials found under the "Details" tab above. A tutorial will guide you through the database and example data, while teaching you how properties work in real-world use cases.
  2. Start building your application targeted by Neo4j. Write your own code in PHP, Java, JavaScript, Python, or one of any number of other languages, using templates provided.
  3. Download Neo4j to your own computer, or start a long-running Neo4j instance in the cloud on AWS or other hosting platforms.

Launch a New Sandbox

Each sandbox includes data, interactive guides with example queries, and sample code.

- Fundamentals Training
  Neo4j Fundamentals classroom training with instructor-led guides.
  Launch Sandbox

- Turnpike
  Design connections in and around the Turnpike administration using the database and Neo4j.
  Launch Sandbox

- Legal Graph
  U.S. Congress modeled as a Graph: Bills, votes, members, and more.
  Launch Sandbox

- Bank Sandbox
  Use your own data with Neo4j. Add or delete data from scratch.
  Launch Sandbox

neo4jsandbox.com
Want to learn more?

graphdatabases.com

neo4j.com/sandbox
HANDS ON TIME!
Logistics

  - Neo4j Sandbox. Private hosted Neo4j instances
  - **Start with this!**

  - Topic clustering, and NLP
  - **Will be used later**
1) GOTO: bit.ly/neo4joscon
1) GOTO: bit.ly/neo4joscon

2) Sign Up for Neo4j Sandbox
Launch a New Sandbox

Each sandbox includes data, interactive guides with example queries, and sample code.

Recommendations
Generate personalized real-time recommendations using a dataset of movie reviews.
Launch Sandbox

OSCON 2017 Neo4j Workshop
OSCON 2017 Neo4j workshop. Learn how to build a real-time recommendation system with Neo4j.
Launch Sandbox

Trumpworld
Explore connections in and around the Trump Administration using this dataset from Buzzfeed.
Launch Sandbox

Twitter
If signed into Neo4j Sandbox using Twitter, this sandbox will allow you to Graph your Twitter network.
Launch Sandbox

Legis-Graph
US Congress modeled as a Graph - bills, votes, members, and more.
Launch Sandbox

Network and IT Management
Dependency and root cause analysis + more for network and IT management.
Launch Sandbox

Launch me!

bit.ly/neo4joscon
Click “Neo4j Browser”

bit.ly/neo4joscon
Neo4j Browser
- Query workbench / visualization for Neo4j
- Interactive “guides” for our tutorial today
- Embed content, queries

bit.ly/neo4joscon
(you) - [:HAVE] -> (?)
(?) <- [:ANSWERS] - (will)