Graph Databases

Studying social interactions with Graph Databases

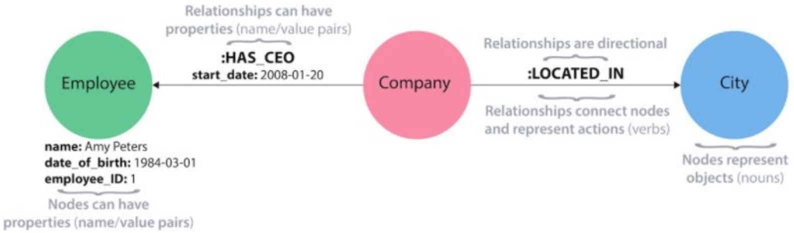
Table of Contents

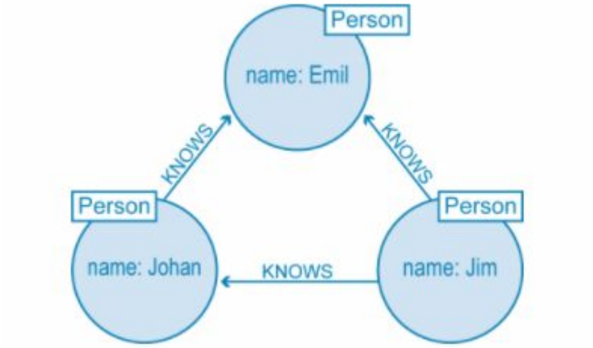
1. Graph Databases technology
2. Why study Graph Databases
3. Use cases for Graph Databases
4. Notable Studies with Graph Databases
5. Examples of graph query & Cypher
6. Graph Databases Tools - GraphQL
7. Graph Databases Tools - Neo4j
8. Graph Databases Tools - MongoDB $graphLookup
9. Wikipedia admin elections - project with Neo4j

Graph Database

Graph Database is a database designed to treat the relationships between data as equally important to the data itself.

Graph Database holds data without constricting it to a predefined model. Instead, the data is stored in a way that it is showing how each individual entity connects with or is related to others.


Relationships can have properties (name/value pairs)
:HAS_CEO
start_date: 2008-01-20
Employee
Company
Relationships are directional
:LOCATED_IN
Relationships connect nodes and represent actions (verbs)
City
name: Amy Peters
date_of_birth: 1984-03-01
employee_ID: 1
Nodes can have properties (name/value pairs)
Nodes represent objects (nouns)

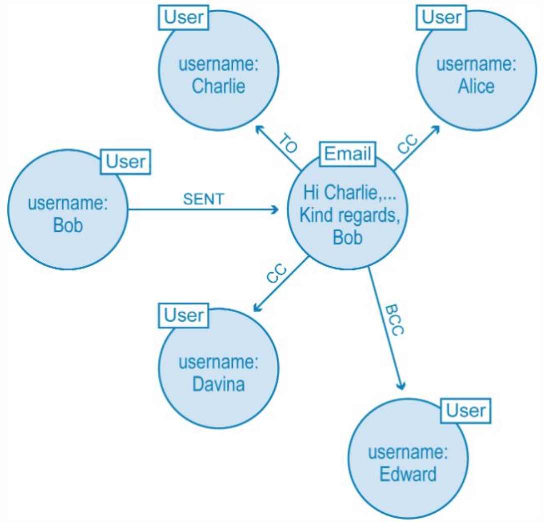


Why study Graph Databases

- Graph Databases give us visual answers/presentations of relationships (human-friendly), which makes them essential in de-coding complex interactions in various fields of study.
- Efficient* for complex connections between entities (analogy: LLL vs Binary Search Tree)
- Graph Databases are concentrating on relationship and quality of relationships between the entities NOT the entities themselves.

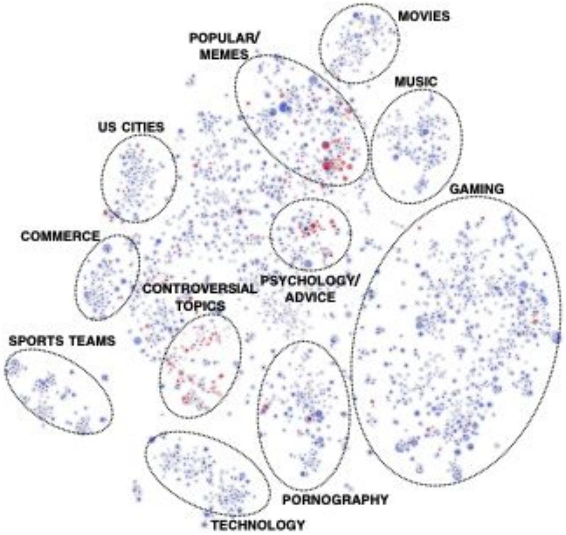
Graph Databases Use Cases:

- Social and Information Networks (e.g., contact tracing, terrorist financing, spam/cyber threats identification/prevention, etc.)
- Biological connections (human interactome, DNA sequence, neural networks, etc.)
- Statistics, NLP, ML algorithms
- Financial Fraud identification/prevention
- Knowledge Graph

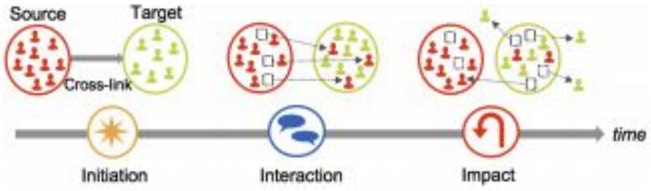


Notable Studies with Graph Databases

1. **Community Interaction and Conflict on the We**
   William L. Hamilton, Jure Leskovec, Dan Jurafsky



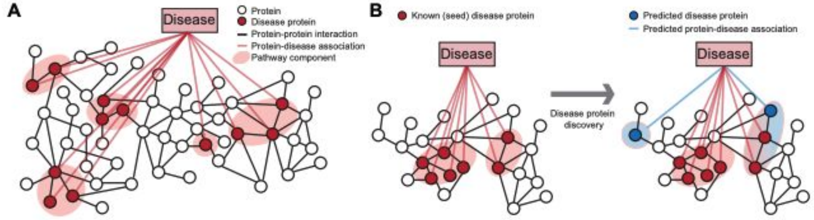
1% of all communities initiate 74% of all conflicts on Reddit. The red nodes (communities) in this map initiate a large amount of conflict, and we can see that these conflict initiating nodes are rare and clustered together in certain social regions.



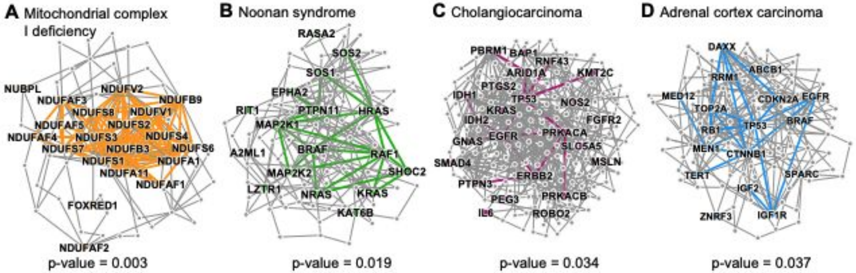
Notable Studies with Graph Databases

2. **Large-scale analysis of disease pathways in the human interactome** by Monica Agrawal, Marinka Zitnik, Jure Leskovec.

Discovering ***disease pathways****, which can be defined as sets of proteins associated with a given* ***disease****, is an important problem that has the potential to provide clinically actionable insights for* ***disease*** *diagnosis, prognosis, and treatment.*

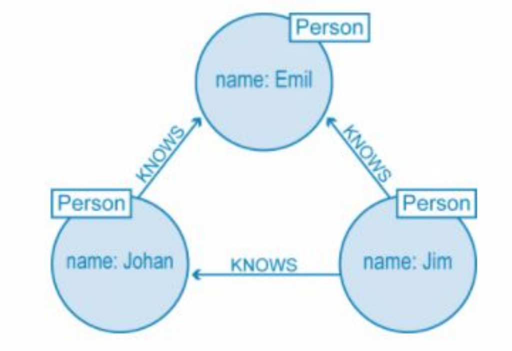

A
Disease
○ Protein
● Disease protein
Protein-protein interaction
Protein-disease association
Pathway component
B
● Known (seed) disease protein
Disease
Disease protein discovery
● Predicted disease protein
Predicted protein-disease association
Disease

Network-based discovery of disease proteins



Disease pathways in the wider PPI network

Cypher & Graph Query Example



1. Data Modeling (80% of efforts)

   Model Flexibility NEGATIVELY correlates with Efficiency

2. Establish patten (mutation)

The pattern of the example graph in Cypher:

(emil)<-[:KNOWS]-(jim)-[:KNOWS]->(johan)-[:KNOWS]->(emil)

3. Binding patterns to specific nodes in DB:

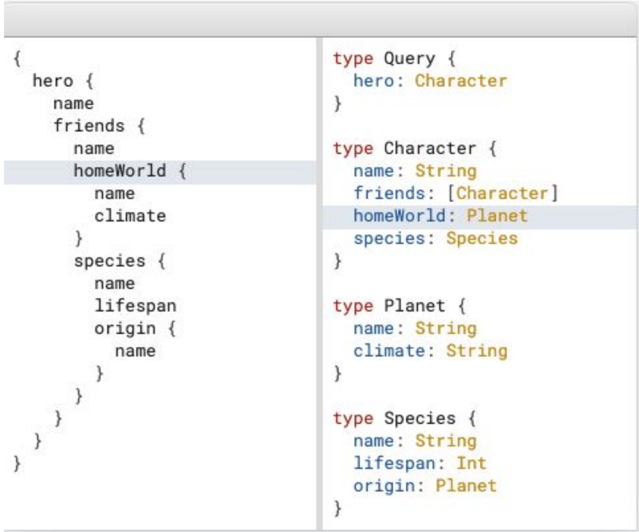
```
(emil:Person {name:'Emil'})
    <-[:KNOWS]-(jim:Person {name:'Jim'})
    -[:KNOWS]->(johan:Person
{name:'Johan'})
    -[:KNOWS]->(emil)
```

4. Running query (e.fg., match)

```
MATCH (a:Person
{name:'Jim'})-[:KNOWS]->(b)-[:KNOWS]->(c),
     (a)-[:KNOWS]->(c)
RETURN b, c
```

Graph Database Tools - GraphQL

- Developed by Facebook in 2015
- GraphQL is an open-source data query and manipulation language for APIs and runtime.
- GraphQL is a query language for APIs and a runtime for fulfilling those queries with your existing data.
- GraphQL APIs are organized in terms of types and fields, not endpoints.
- Supports changes of API WITHOUT versions (avoids introducing breaking changes)

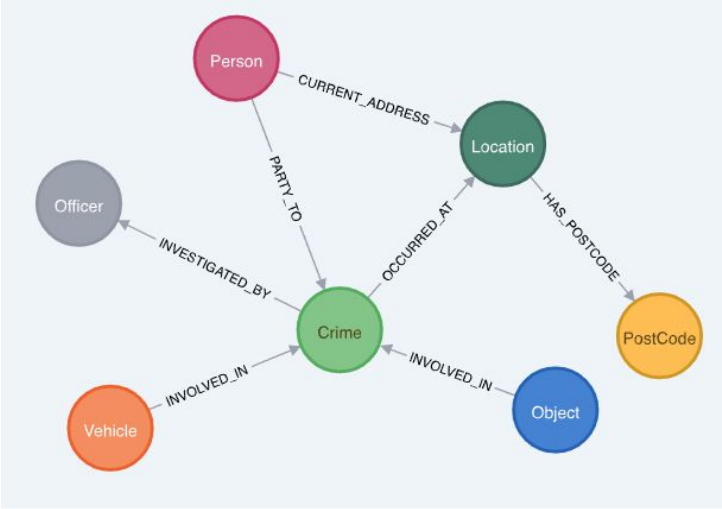


```
{
  hero {
    name
    friends {
      name
      homeWorld {
        name
        climate
      }
      species {
        name
        lifespan
        origin {
          name
        }
      }
    }
  }
}
```

```
type Query {
  hero: Character
}

type Character {
  name: String
  friends: [Character]
  homeWorld: Planet
  species: Species
}

type Planet {
  name: String
  climate: String
}

type Species {
  name: String
  lifespan: Int
  origin: Planet
}
```

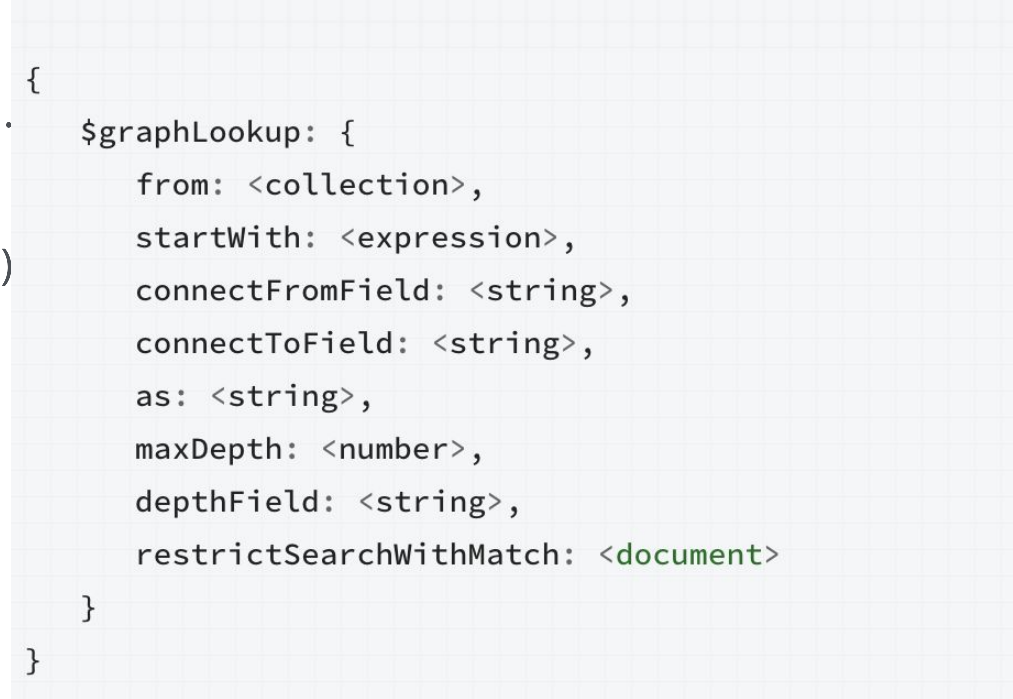
Graph Database Tools - Neo4j

- Neo4j is a graph database management system with ACID-compliant transactional database.
- The most popular graph database
- Allows developers to have complete control over data modeling and efficiency
- Requires path identification and binding
- Offers variety of libraries and products for data analysis and visualization.
- Operates on Cypher and data is stored in native graphs.



Graph Database Tools - MongoDB ($graphLookup)

- MongoDB is a NoSQL document-oriented database program.
- MongoDB uses JSON-like documents with optional schemas (flexible model!)
- $graphLookup is an aggregation function
- MongoDB is NOT a graph database



```
{
   $graphLookup: {
      from: <collection>,
      startWith: <expression>,
      connectFromField: <string>,
      connectToField: <string>,
      as: <string>,
      maxDepth: <number>,
      depthField: <string>,
      restrictSearchWithMatch: <document>
   }
}
```

Project with Graph Database Neo4j

Wikipedia is a free encyclopedia written collaboratively by volunteers around the world.

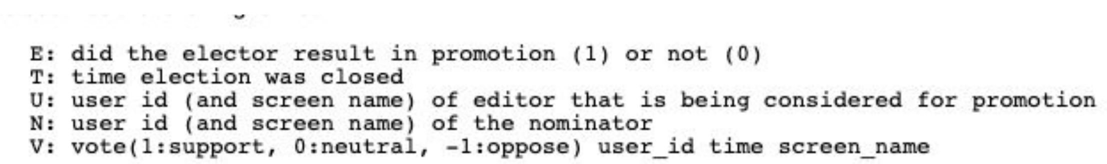
A small part of Wikipedia contributors are administrators (users with access to additional technical features that aid in maintenance).

In order for a user to become an administrator a Request for adminship (RfA) is issued and the Wikipedia community via a public discussion or a vote decides who to promote to adminship.

Project with Graph Database Neo4j

This project will be using the extracted administrator elections and vote history data, which has 2,800 elections with around 100,000 total votes and about 7,000 users participating in the elections.

Dataset for the project:



```
E: did the elector result in promotion (1) or not (0)
T: time election was closed
U: user id (and screen name) of editor that is being considered for promotion
N: user id (and screen name) of the nominator
V: vote(1:support, 0:neutral, -1:oppose) user_id time screen_name
```

Objective of the project:

Construct graph model and based on the model to Identify and understand voting preferences.

Project Strategy and Deliverables

1. Data Cleaning
2. Build Data Model (mutation to graph data)
3. Build Relationship matrix
4. Bind matrix to DB
5. Build Graph

Deliverables:

1.Visual presentations of election results (graph data)

2. Graph presentation of user votes

3. Analysis (who votes for who) purely delivered on graph presentation.

Questions?

THANK YOU!