Database Theory and Applications for Biomedical Research and Practice

BMIN 502 / EPID 635
Week 11: Introduction to graph database

John H. Holmes, PhD

Institute for Biomedical Informatics

Agenda

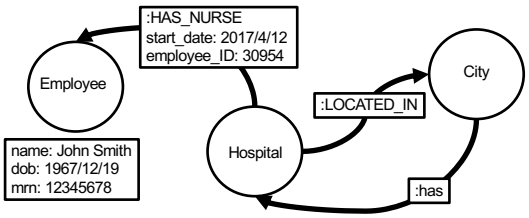
- Introduction to graph models
- Modeling a toy problem as a graph
- Modeling ABIC as a graph
- Graph databases
- Graph modelsGraph query languages
- Introduction to neo4j

Two types of graph models

- Resource Description Framework (RDF)
  - Generic "metadata model" originating in the semantic web
  - Uses a triple construct (subject-predicate-object): "The patient takes Tylenol"
  - Strongly indexed
  - Nodes and edges have no internal structure
- Labeled Property Graph
  - Used in Neo4j
  - Index-free
  - Nodes and edges have internal structure

The Labeled Property Graph Model

- A graphical representation of concepts and the relationships between them, with the properties associated with each



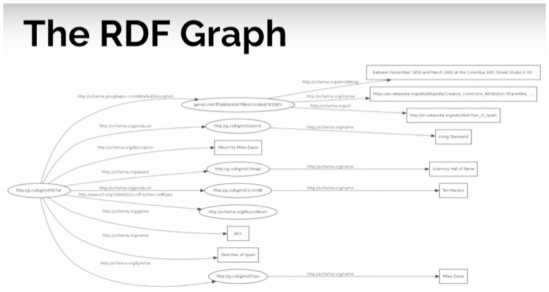
Thus…

- Nodes
  - Equivalent to entities
  - Contain properties
    - Attribute-value pairs
  - Labels indicate identity
- Relationships
  - Express the semantics between nodes
    - Always connected between two nodes, never broken or "orphaned"
  - Directional
  - Properties
    - Indicate possession, quantitative measures, strength, state, etc.

Compare RDF and LPG models

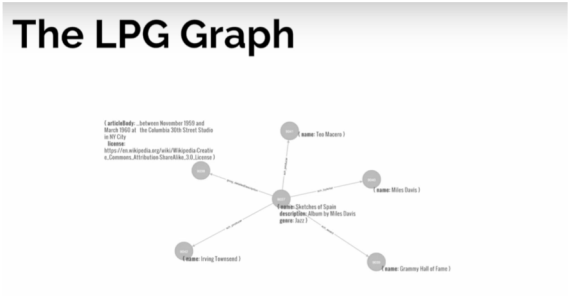
Resource Description Framework	**Labeled Property Graph**
• Not possible to uniquely identify instances of a relationship – Impossible to have connections of the same type between the same pair of nodes, because that would duplicate the triple • You can't provide attributes for relationships	• You can uniquely identify all instances of a relationship – Can lead to huge graphs! • Relationships can and do have attributes – Much more versatility in querying

An RDF graph



Note how every concept is ultimately atomicized!
(E.g., values of attributes are represented as nodes.)

An LPG graph

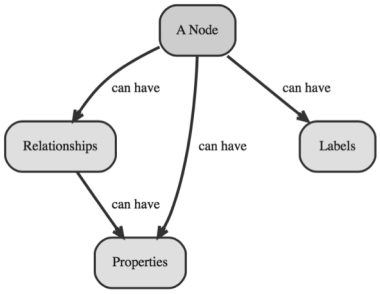


Note how every concept represented in nodes with an internal structure.

How to create a graph model: Step 1: Identify the domain

- This will be the tableau of the graph, characterized as nodes (“nouns”) and relationships (“verbs”)
- Example: John, Mary, Bill, and Ellen are patients at UPHS practices. John and Mary go to 3701 Market Internal Medicine, Ellen goes to 8th and Spruce, and Bill goes to Family Medicine at 39th and Chestnut.

How to create a graph model: Step 2: Identify the nodes

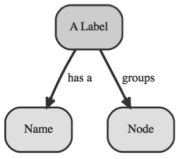


How to create a graph model: Step 2: Identify the nodes

John, Mary, Bill, and Ellen are patients at UPHS practices. John and Mary go to 3701 Market Internal Medicine, Ellen goes to 8th and Spruce, and Bill goes to Family Medicine at 39th and Chestnut.

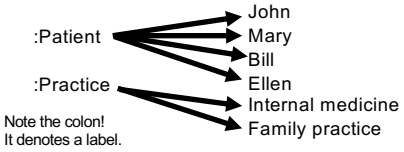
How to create a graph model: Step 3: Specify the node labels

- Labels define sets
- Equivalent to the name of an entity, which is an abstraction (“a patient”, not “the patient”)
- Important to the functioning of the graph database

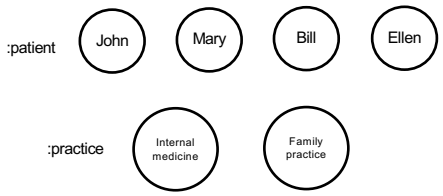


How to create a graph model: Step 3: Specify the node labels

John, Mary, Bill, and Ellen are patients at UPHS practices. John and Mary go to 3701 Market Internal Medicine, Ellen goes to 8th and Spruce, and Bill goes to Family Medicine at 39th and Chestnut.



How to create a graph model: Step 4: Apply the labels as roles



Colors aren't absolutely necessary, but they help to indicate the distinction between labels and roles

How to create a graph model: Step 5: List the relationships

John, Mary, Bill, and Ellen are patients at UPHS practices. John and Mary go to 3701 Market Internal Medicine, Ellen goes to 8th and Spruce, and Bill goes to Family Medicine at 39th and Chestnut.

How to create a graph model: Step 6: Draw the graph and generate the code

We will return to this!

Now, how about ABIC as a graph?

Let's model it

Graph Databases

What is a graph database? Wikipedia says it well…

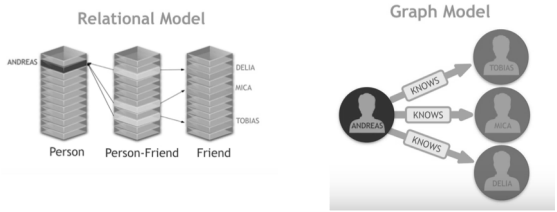
- Uses graph structures (rather than tables)
- Allows semantic queries
  - Rretrieval of both explicitly and implicitly derived information based on syntactic, semantic and structural information contained in data
- The graph explicitly relates data items
- Nodes instantiate items
- Relationships link data in nodes directly, without need for joins

https://en.wikipedia.org/wiki/Graph_database

Comparison of relational and graph databases

Relational	Graph
• Storing and manipulating relationships are difficult • Performance degrades with increasing number and complexity of relationships • Queries get complex real fast, when you rely on a lot of joins • Adding new relationships, tables, and types of data requires schema redesign	• Can handle large volumes of heterogeneous data • Graph models are intuitive • Model and data stored as a graph • Data relationships can be queried quickly (real-time) • Can be updated without requiring schema redesign

Another view: Look at the relationship flexibility…



And think about the complexity of the joins required for the relational model!

Graph query languages

Query languages in graph databases: SPARQL

- SQL-like language for querying RDF graphs

```
PREFIX foaf:
SELECT ?craft ?homepage
{
  ?craft foaf:name "Apollo 7" .
  ?craft foaf:homepage ?homepage
}
```

Query languages in graph databases: Gremlin

- Graph traversal language for querying in a variety of platforms and within other languages (Python, Java, JavaScript, etc.)

```
def gt=graph.traversal();
gt.V()
.hasLabel("band")
.has("genre",within("indie","HipHop","Techno/EDM"))
.outE("performing_at")
.inV().path();
```

Query languages in graph databases: Cypher

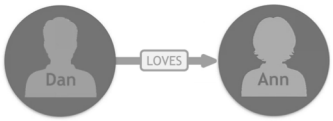
- Declarative query language for Neo4J

```
MATCH (n1:Label1)-[rel:TYPE]->(n2:Label2)
WHERE rel.property > {value}
RETURN rel.property, type(rel)
```

```
1  MATCH (n:Person)-[:FRIEND]-(f)
2  WITH count(f) as c, n
3  MATCH (n)-[:FRIEND]-()-[:FRIEND]-(fof)
4  RETURN n, c, fof
```

A simple Cypher query

https://www.youtube.com/watch?v=83P81ebgCxA




```
CREATE (:Person { name:"Dan"} ) -[:LOVES]-> (:Person { name:"Ann"} )
```

Predicate Label Property Relationship Label Property

This query inserts two nodes into the database, each with an instantiated property (name) and an instantiated relationship (:LOVES)

Running neo4j: A first run

- Open the application
- Click on New
- Click on create New Graph and then Create a Local Graph
- Enter a name and password for the database and click on Create
- Click Start
- Click on the name of the DB in the lower left and click on Open Browser
- Click on Write Code button (in center)
- Click on Create a graph (Movie Graph)
- Let's walk through the tutorial together

How to create a graph model: Step 6: Draw the graph and generate the code

- Start the Arrow Tool: www.apcjones.com/arrows/#
- Instantiate your graph from Step 5 using the Arrow Tool
- Export the Cypher code and open in the Console
- Copy and paste the code into the Browser
- Visualize your graph there