Property Graphs

ANNA QUERALT

FACULTAT D'INFORMÀTICA DE BARCELONA

Data Model

The Property Graph Data Model

Born in the database community

- Meant to be queried and processed
- GQL standard, but only since April 2024

Two main constructs: nodes and edges

- Nodes represent entities
- Edges relate pairs of nodes, and may represent different types of relationships

Both nodes and edges:

- May be labeled
- May have a set of properties represented as attributes (key-value pairs)

Further considerations:

- Edges are directed
- Multi-graphs are allowed

Direction in edges is just about their meaning

Formal Definition

Definition 2.3 (Property graph). A property graph G is a tuple $(V, E, \rho, \lambda, \sigma)$, where:

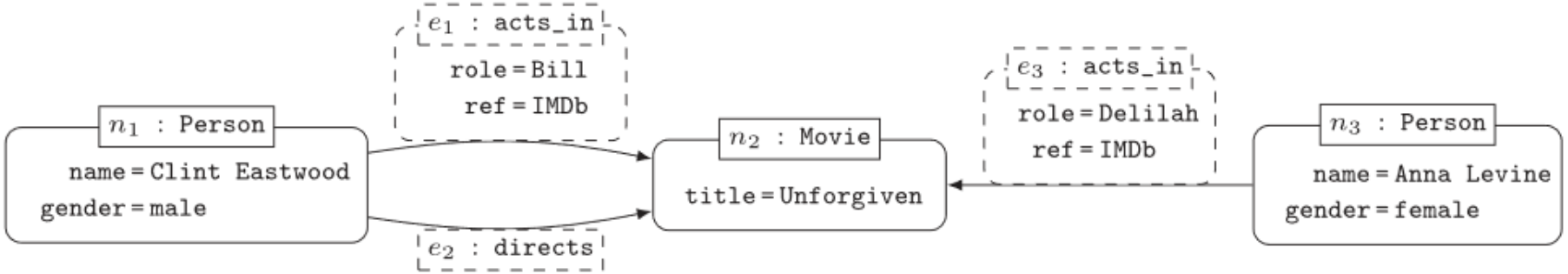
(1) V is a finite set of *vertices* (or *nodes*).
(2) E is a finite set of *edges* such that V and E have no elements in common.
(3) $\rho : E \to (V \times V)$ is a total function. Intuitively, $\rho(e) = (v_1, v_2)$ indicates that e is a directed edge *from* node v_1 *to* node v_2 in G.
(4) $\lambda : (V \cup E) \to Lab$ is a total function with Lab a set of labels. Intuitively, if $v \in V$ (respectively, $e \in E$) and $\lambda(v) = \ell$ (respectively, $\lambda(e) = \ell$) , then ℓ is the label of node v (respectively, edge e) in G.
(5) $\sigma : (V \cup E) \times Prop \to Val$ is a partial function with $Prop$ a finite set of properties and Val a set of values. Intuitively, if $v \in V$ (respectively, $e \in E$), $p \in Prop$ and $\sigma(v, p) = s$ (respectively, $\sigma(e, p) = s$), then s is the value of property p for node v (respectively, edge e) in the property graph G.

Extracted from: R. Angles et al. Foundations of Modern Query Languages for Graph Databases

Does this definition admit...

- Several edges between the same nodes?
- Edges between more than 2 nodes?
- Edges with a single node?
- Properties in edges?
- Nodes without a label? And edges?
- The same property in different nodes or edges?
- Nodes/edges with the same label and different set of properties?

Example of Property Graph



A valid graph according to the previous formal definition. Instantiation:

$V =$ $E =$

ρ σ

λ

Traversal Navigation

We define the graph traversal pattern as:

“the ability to rapidly traverse structures to an arbitrary depth and with an arbitrary path description” [Marko Rodriguez]

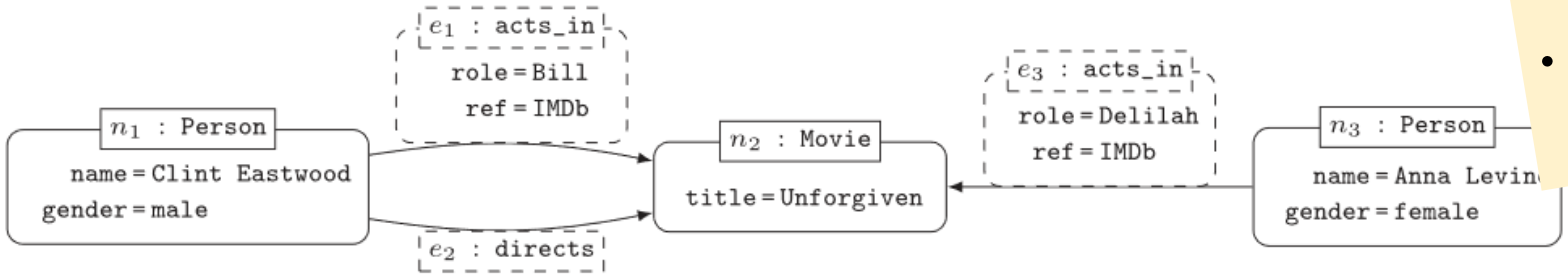
Traversal Navigation

We define the graph traversal pattern as:

*"the ability to **rapidly** traverse structures to an **arbitrary depth** and with an **arbitrary path description**"* [Marko Rodriguez]

Totally opposite to set theory (on which relational databases are based on)

- Sets of elements are operated by means of the relational algebra



Examples:

- Who acted with Clint Eastwood in the same movie?
- Movies in which the director is also an actor

Traversing Data in a RDBMS

Obtaining all the items that Alice has ordered

Users

User	Address	Phone	Email
Alice			
Bob	456 Bar Ave.		bob@example.org
…	…	…	…
Zach	99 South St.		zach@example.org

Orders

Order	Date	Status	UserId
1234	20120808	delivered	Alice
5678	20120816	dispatched	Alice
…	…	…	…
5588	20120613	delivered	Zach

Items

Id	Description	Handling
abcd	Strawberry ice-cream	freezer
efab	Brussels sprouts	
cdef	Espresso beans	
…	…	…

OrderItems

OrderId	ItemId
1234	abcd
1234	efab
1234	cdef
5678	cdef
…	
5588	hijk

```
SELECT i.id, i.description
FROM users u, orders o,
  order_items oi, items i
WHERE u.user = 'Alice' AND
  u.user = o.userId AND
  o.order = oi.orderId AND
  oi.itemId = i.id
```

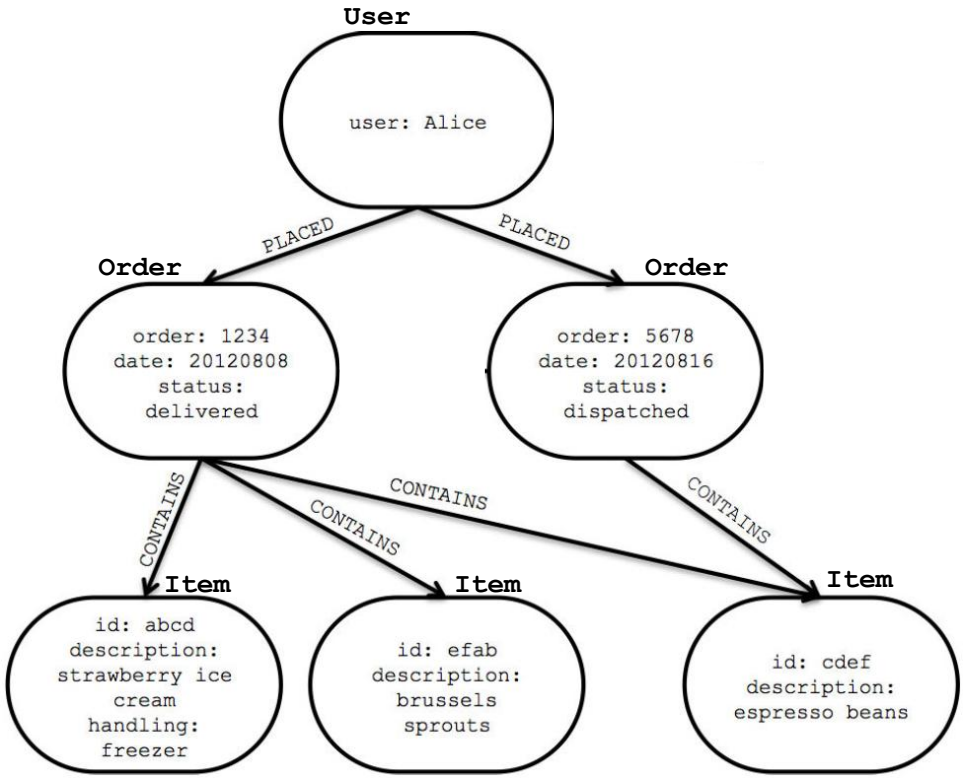
Navigation in RDBMS is equivalent to join (schema level) + selection (instance level)

Cardinalities:

|Users|: 5.000.000
|Orders|: 1.000.000.000
|OrderItems|: 3.000.000.000
|Items|: 35.000

Is this really a traversal navigation?

Traversing Data in a Graph Database



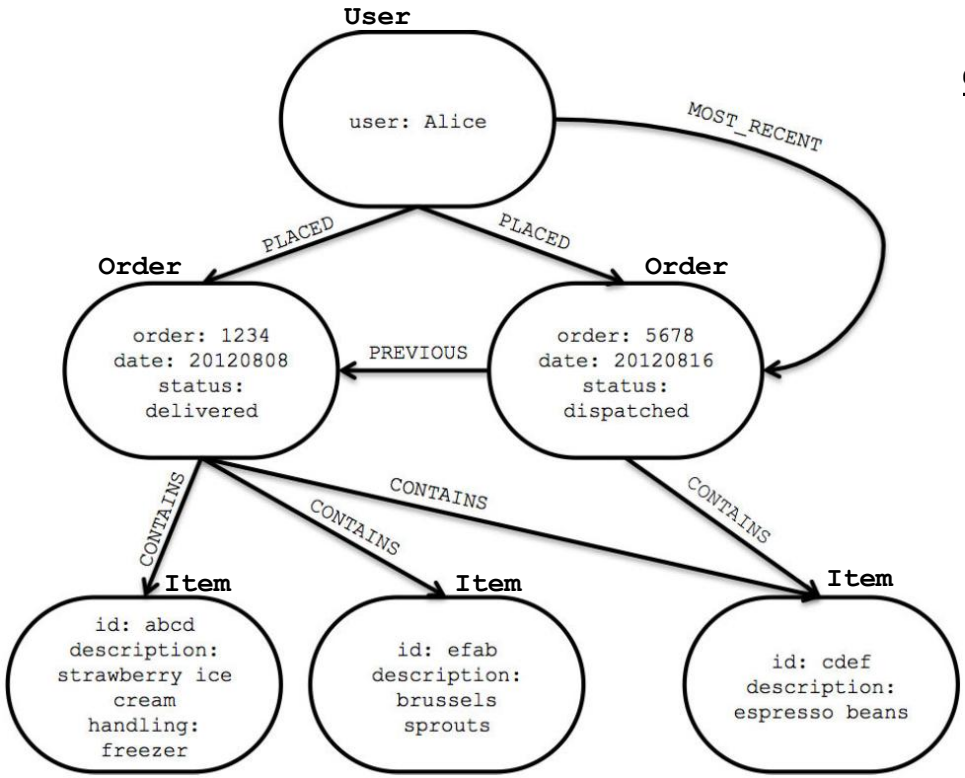
What if now an order can be placed by more than one person?

- Relationships are explicit
  - No need for Foreign Keys
  - No need to add nodes for “artificial” concepts
  - No need for JOIN operation, JOINs are “hard-wired”
- Traversing from one node to another is a **constant time** operation

Activity

What would be the cost of the same query (obtaining all the items that Alice has bought) in a graph database?

- *Assume you can find the node "Alice" in constant time*



Cardinalities:

|Users|: 5.000.000
|Orders|: 1.000.000.000
|Items|: 35.000

Graph DBs vs Relational DBs

The following table reports the result of an experiment aimed at finding friends-of-friends in a social network, to a maximum depth of 5

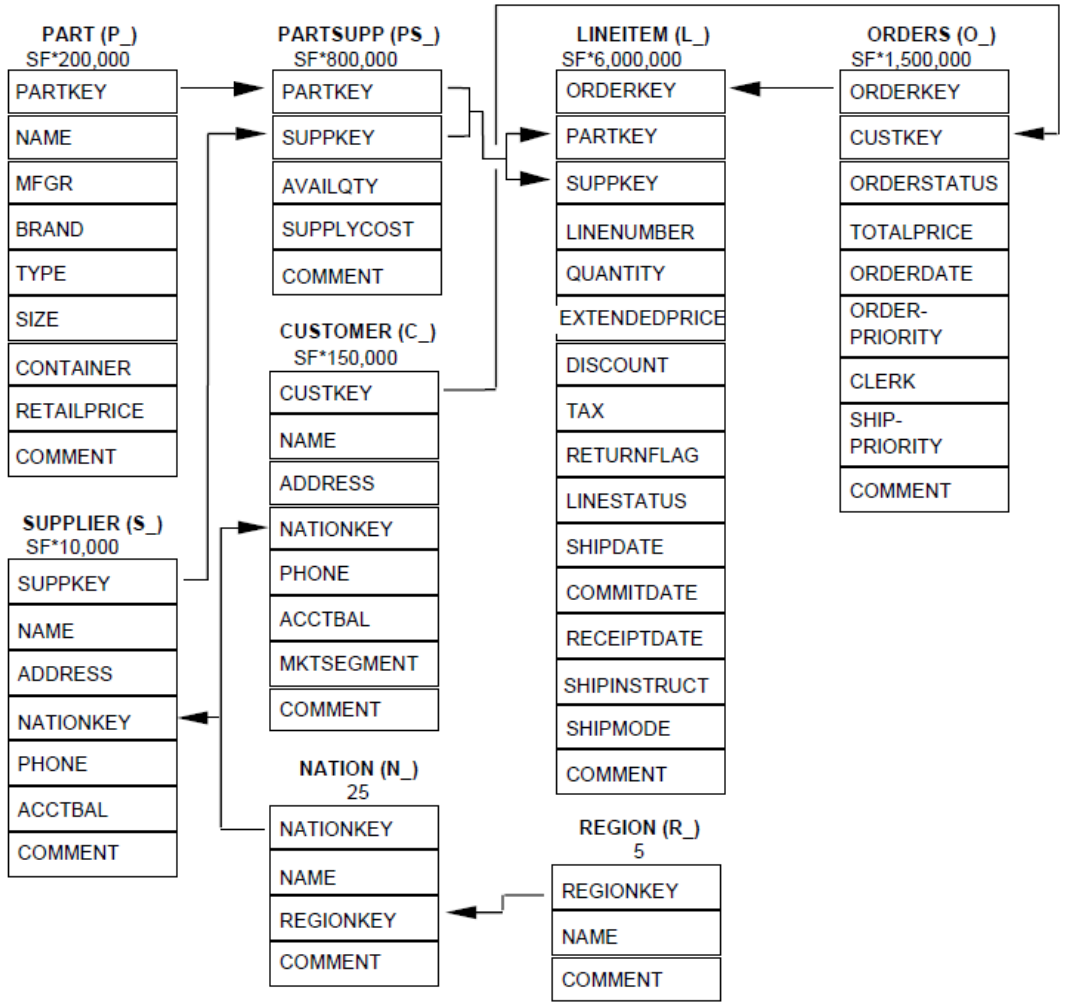
- For a social network containing 1,000,000 people, each of them with approximately 50 friends.

Depth	RDBMS execution time (s)	Neo4j execution time (s)	Records returned
2	0.016	0.01	~2500
3	30.267	0.168	~110,000
4	1543.505	1.359	~600,000
5	Unfinished	2.132	~800,000

From *Neo4j in Action. Jonas Partner, Aleksa Vukotic, and Nicki Watt. MEAP. 2012*

Activity: Modeling in Graph DBs

- *Objective: Learn how to model property graphs*
- *Tasks:*
  - *(15') Model the TPC-H database as a property graph*
  - *(15') Discussion*



Summary

Property graphs are:

- Directed
- Multigraph
- Nodes/edges may have labels (similar to the concept of typing) and properties (equivalent to the concept of attributes)

Completely different mindset, compared to RDBMS:

- Schema-less
- No FKs, no JOINS, no relationship tables
- Navigating an edge is a constant time operation

Thanks! Any Question?