Graph Databases

Data is more connected:

- Text
- HyperText
- RSS
- Blogs
- Tagging
- RDF

Data is more Semi-Structured:

- If you tried to collect all the data of every movie ever made, how would you model it?
- Actors, Characters, Locations, Dates, Costs, Ratings, Showings, Ticket Sales, etc.

Document Databases

- MongoDB, CouchDB

- Pros:
  - Simple, powerful data model
  - Scalable
- Cons
  - Poor for interconnected data
  - Query model limited to keys and indexes
  - Map reduce for larger queries

Graph Databases

- Data Model:
  - Nodes and Relationships
- Examples:
  - Neo4j, OrientDB, InfiniteGraph, AllegroGraph
- Cypher
  - Graph database query language

Graph Databases: Pros and Cons

- Pros:
  - Powerful data model, as general as RDBMS
  - Connected data locally indexed
  - Easy to query
- Cons
  - Sharding (lots of people working on this)
    - Scales UP reasonably well
  - Requires rewiring your brain

What are graphs good for?

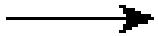
- Real Time Recommendations
- Master Data Management
- Fraud Detection
- Social computing
- Systems management
- Web of things
- Genealogy
- Time series data
- Product catalogue
- Web analytics
- Scientific computing (especially bioinformatics)
- And much more!

What is a Graph?

What is a Graph?

- An abstract representation of a set of objects where some pairs are connected by links.

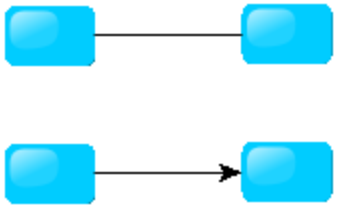
Object (Vertex, Node)



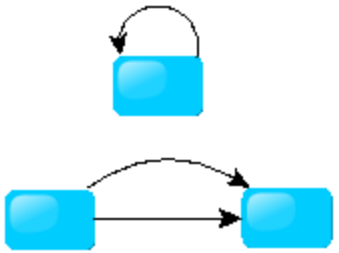
Link (Edge, Arc, Relationship)

Different Kinds of Graphs

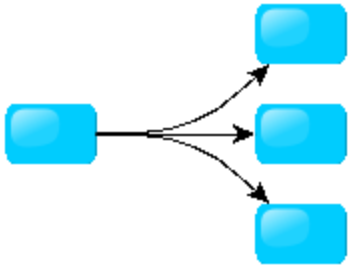
- Undirected Graph
- Directed Graph



- Pseudo Graph
- Multi Graph



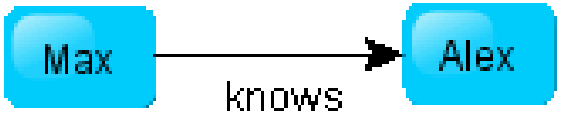
- Hyper Graph



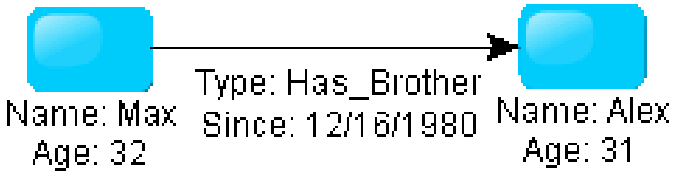
More Kinds of Graphs

- Weighted Graph

- Labeled Graph



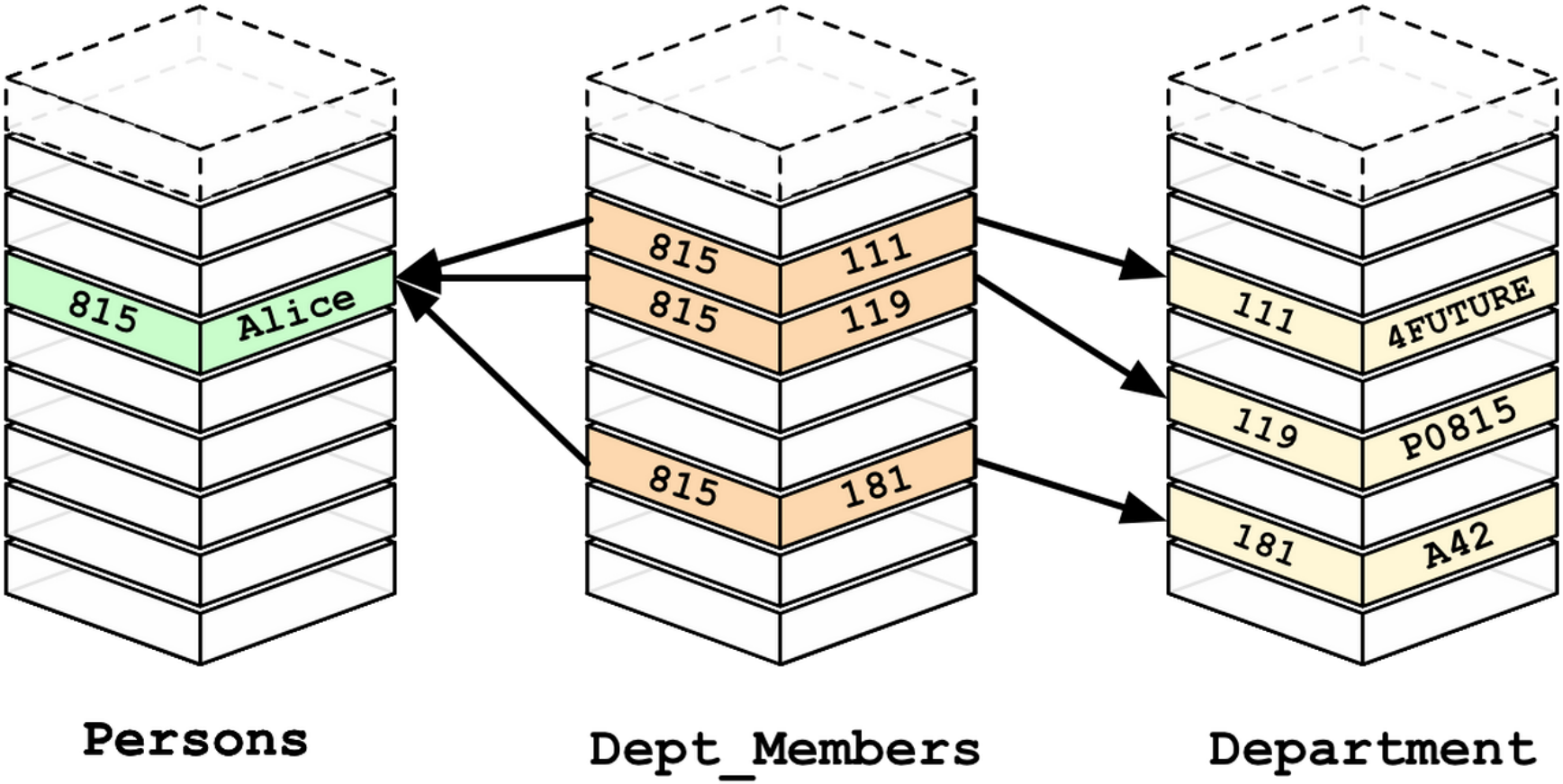
- Property Graph



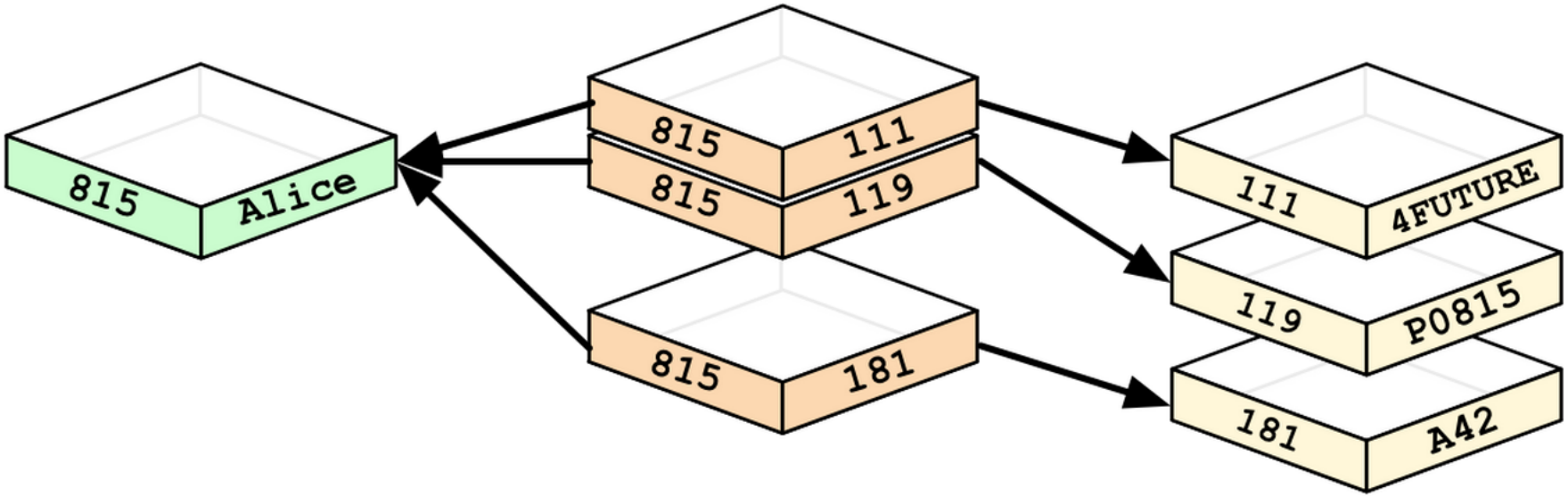
What is a Graph Database?

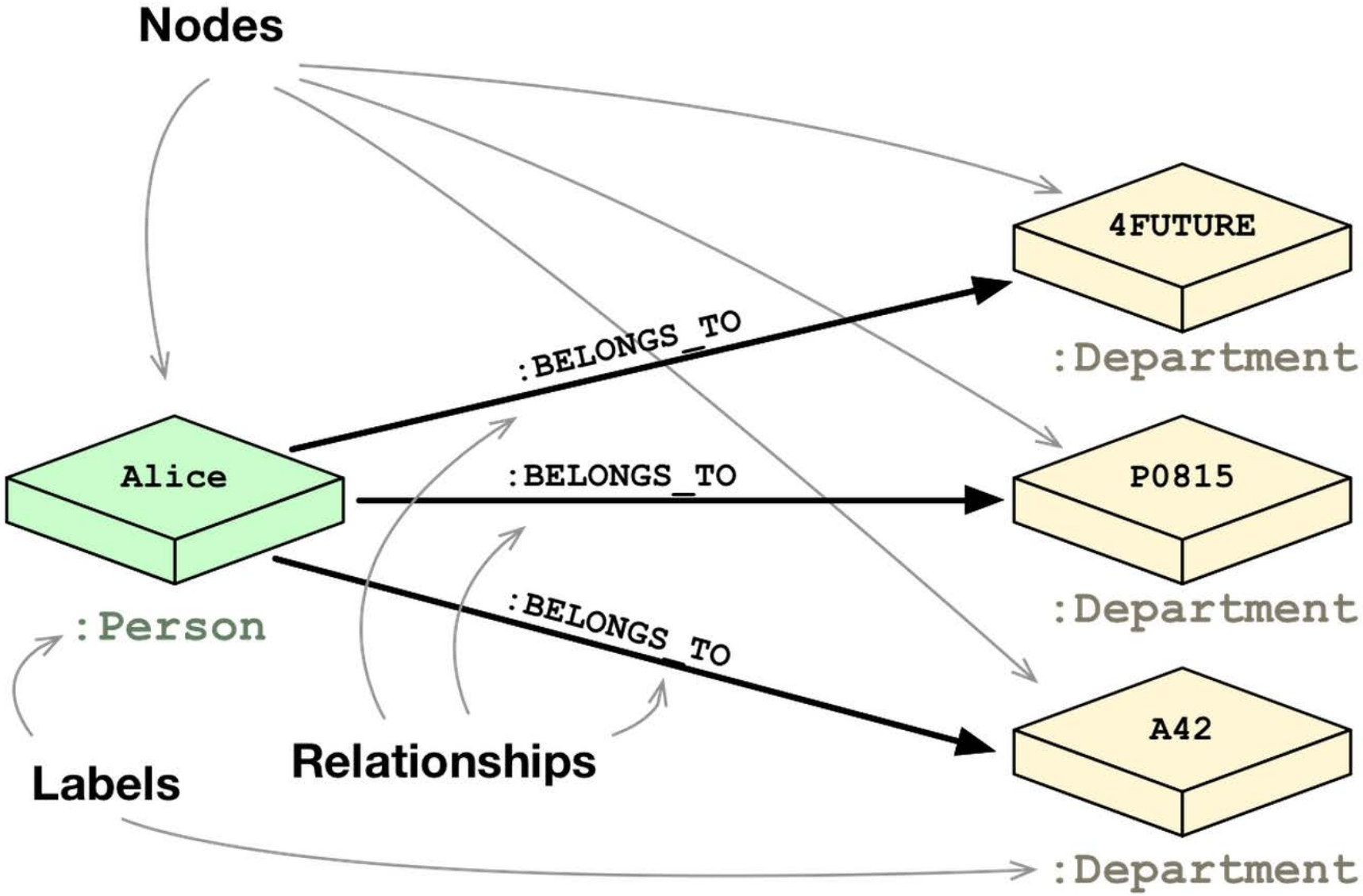
- A database with an explicit graph structure
- Each node knows its adjacent nodes
- As the number of nodes increases, the cost of a local step (or hop) remains the same
- Plus an Index for lookups

Relational Databases

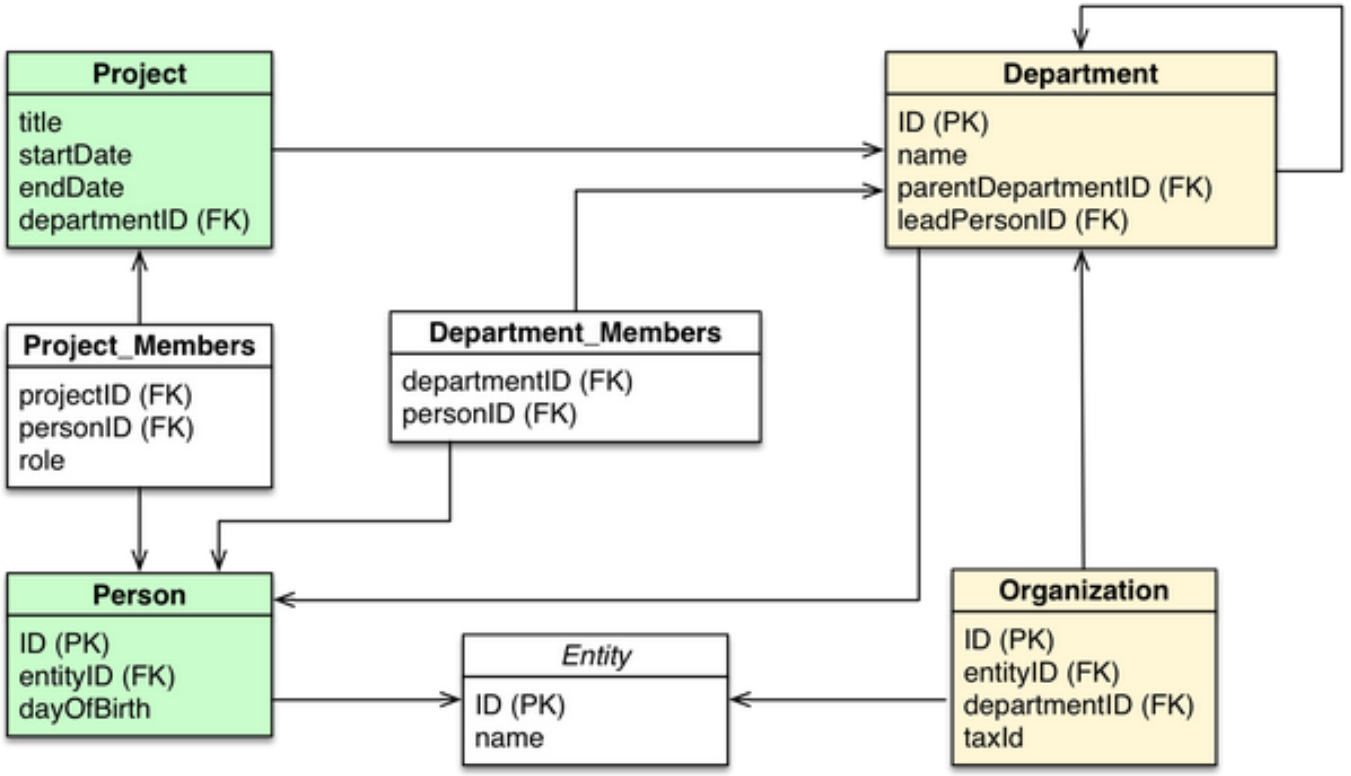


Graph Databases

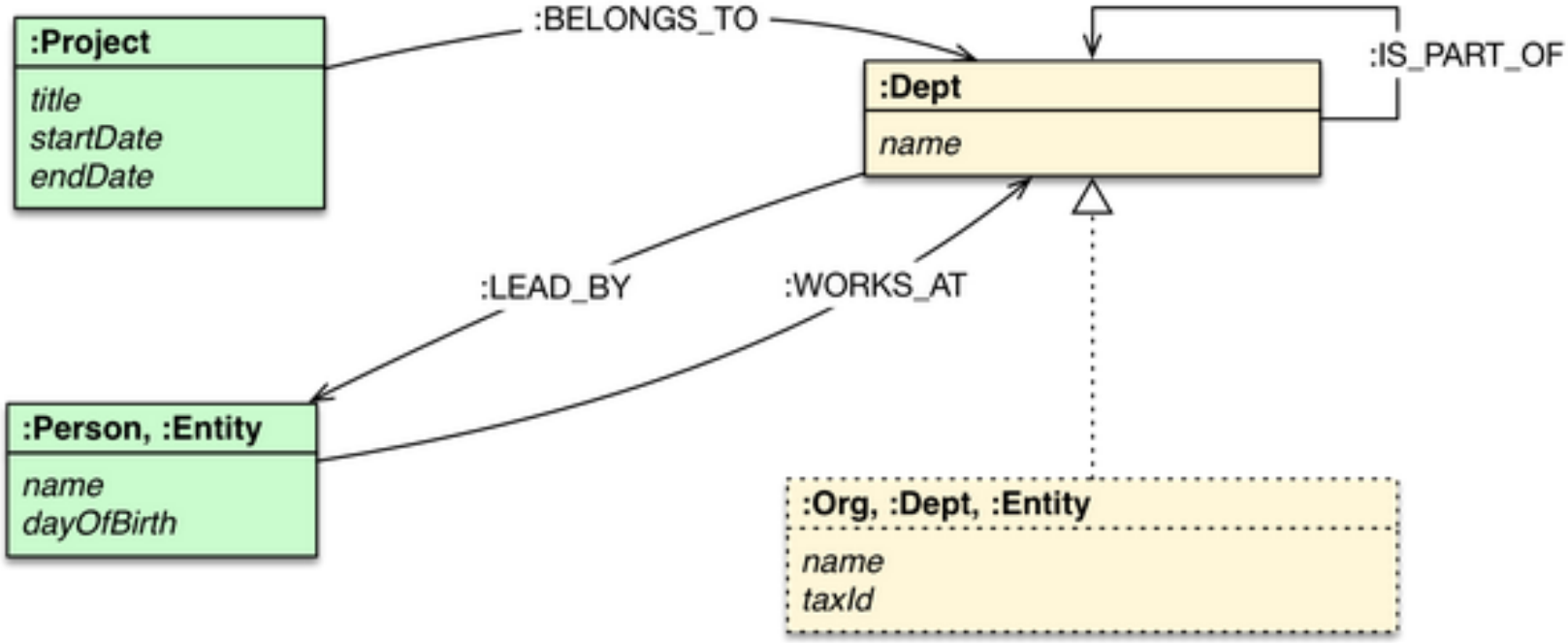



Nodes
4FUTURE
:Department
:BELONGS_TO
Alice
:BELONGS_TO
P0815
:Department
:Person
:BELONGS_TO
A42
Relationships
Labels
:Department

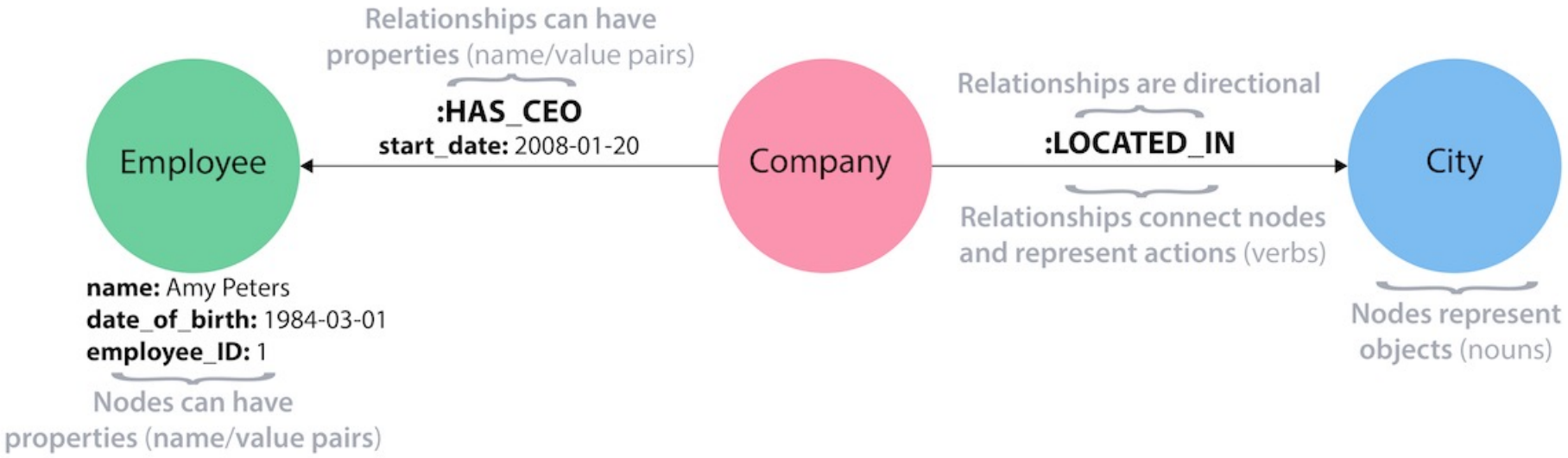
Relational Databases



Graph Databases



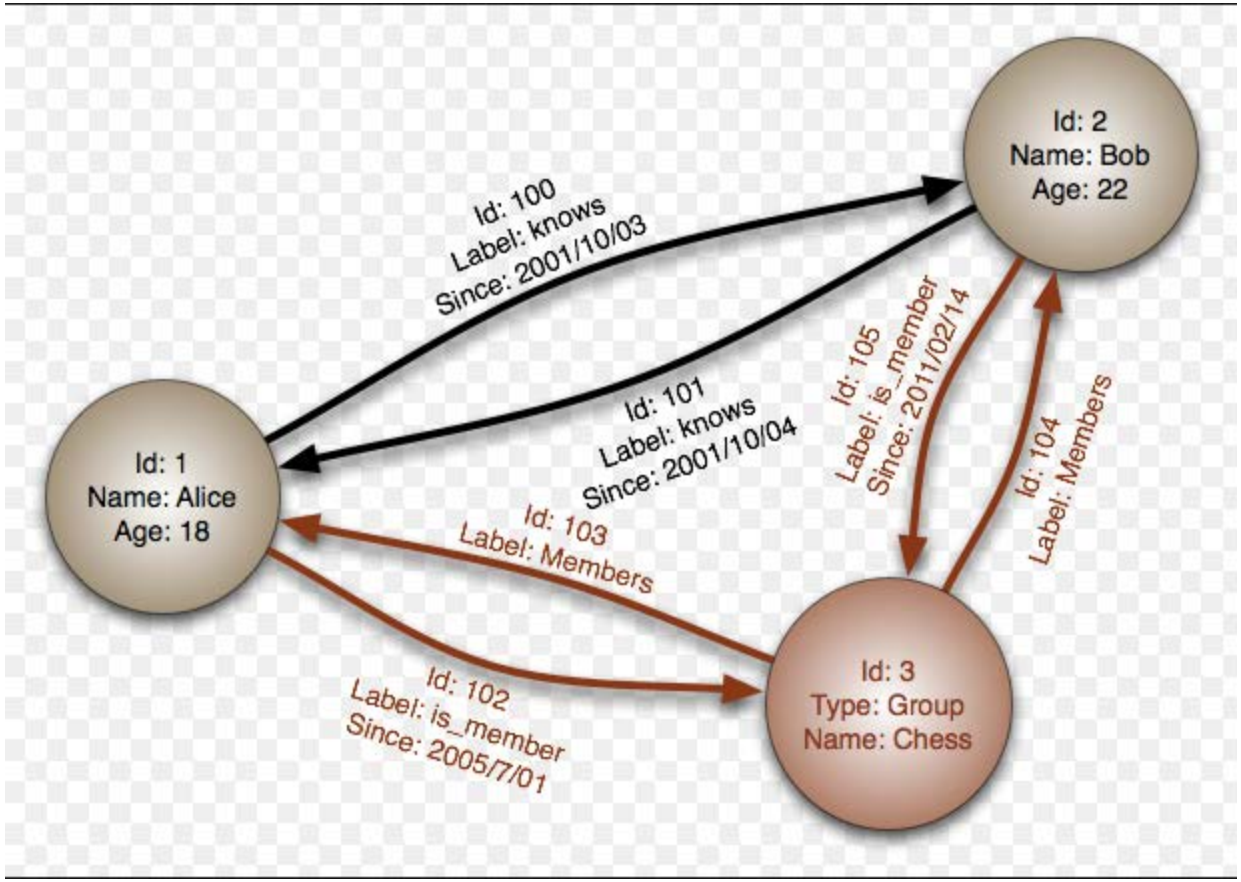
Property graph model



Graph Databases

- Database that uses graph structures with nodes, edges and properties to store data
- Provides index-free adjacency
  - Every node is a pointer to its adjacent element
- Edges hold most of the important information and connect
  - nodes to other nodes
  - nodes to properties

Graph Databases



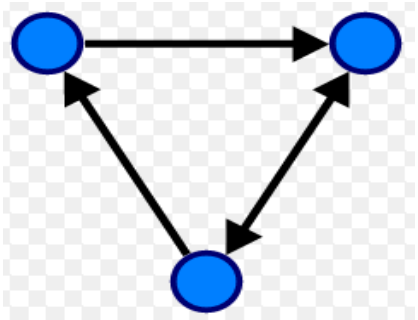
(Wikipedia)

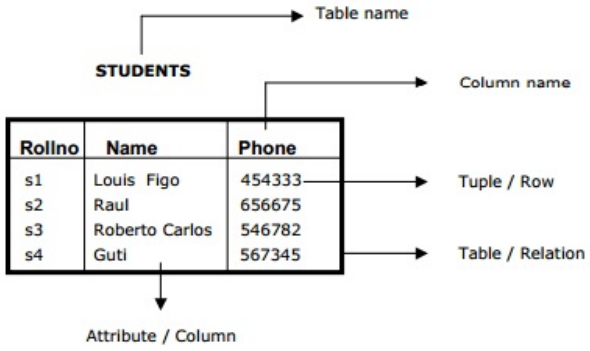
Advantage of Graph Databases

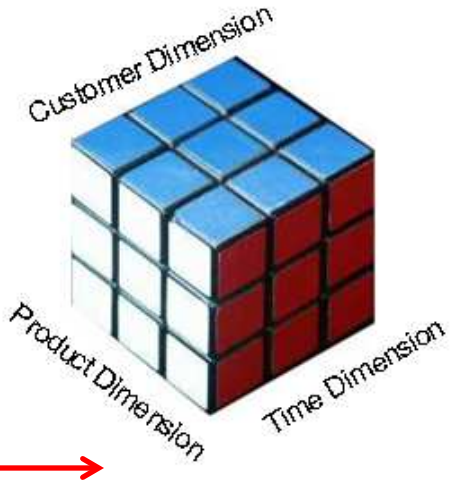
- When there are relationships that you want to analyze Graph databases become a very nice fit because of the data structure
- Graph databases are very fast for associative data sets
  - Like social networks
- Map more directly to object oriented applications
  - Object classification and Parent->Child relationships

Disadvantages

- If data is just tabular with not much relationship between the data, graph databases do not fare well
- OLAP support for graph databases is not well developed
  - Lots of research happening in this area



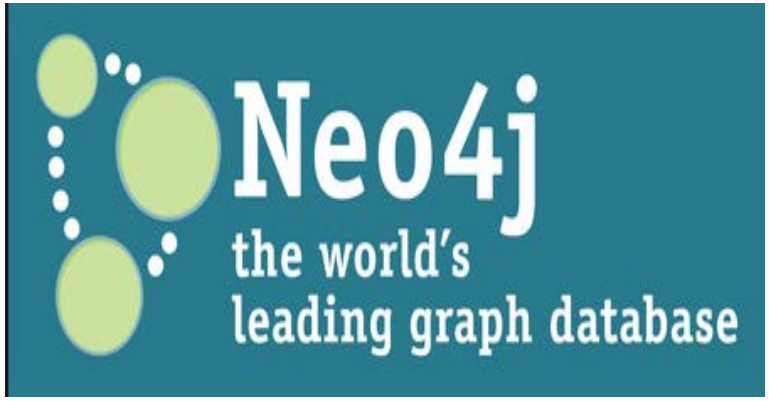




Ease of aggregation

What is Neo4j

- Developed by Neo Technologies
- Most Popular Graph Database
- Implemented in Java
- Open Source



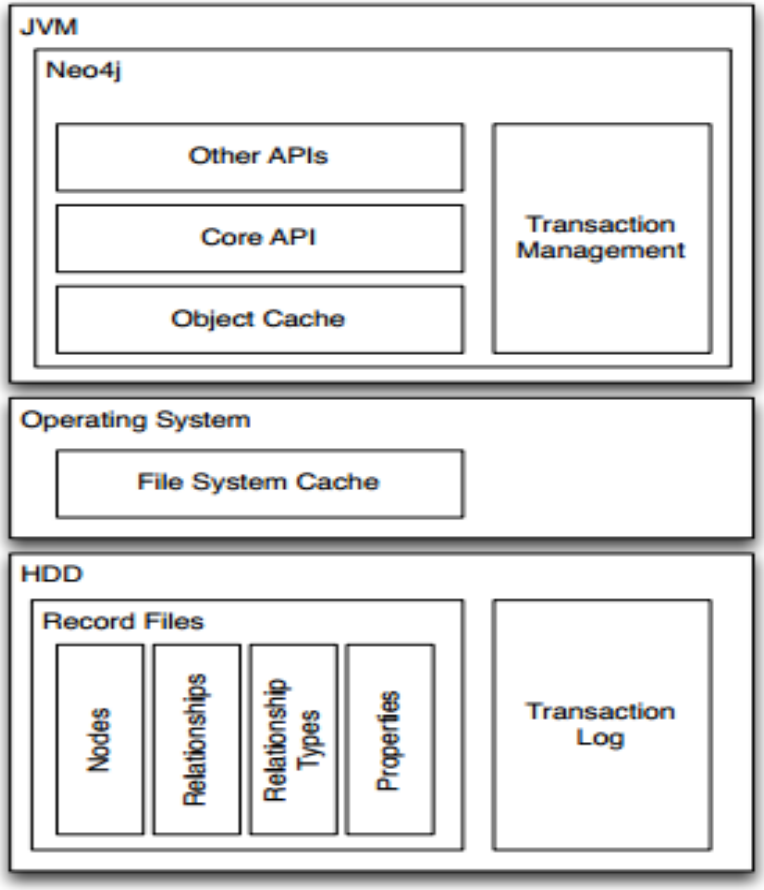
(www.neo4j.org)

Salient features of Neo4j

- Neo4j is schema free – Data does not have to adhere to any convention
- ACID – atomic, consistent, isolated and durable for logical units of work
- Easy to get started and use
- Well documented and large developer community
- Support for wide variety of languages
  - Java, Python, Perl, Scala, Cypher, etc

(Mistry, Deep, 2013)

Neo4j Software Architecture

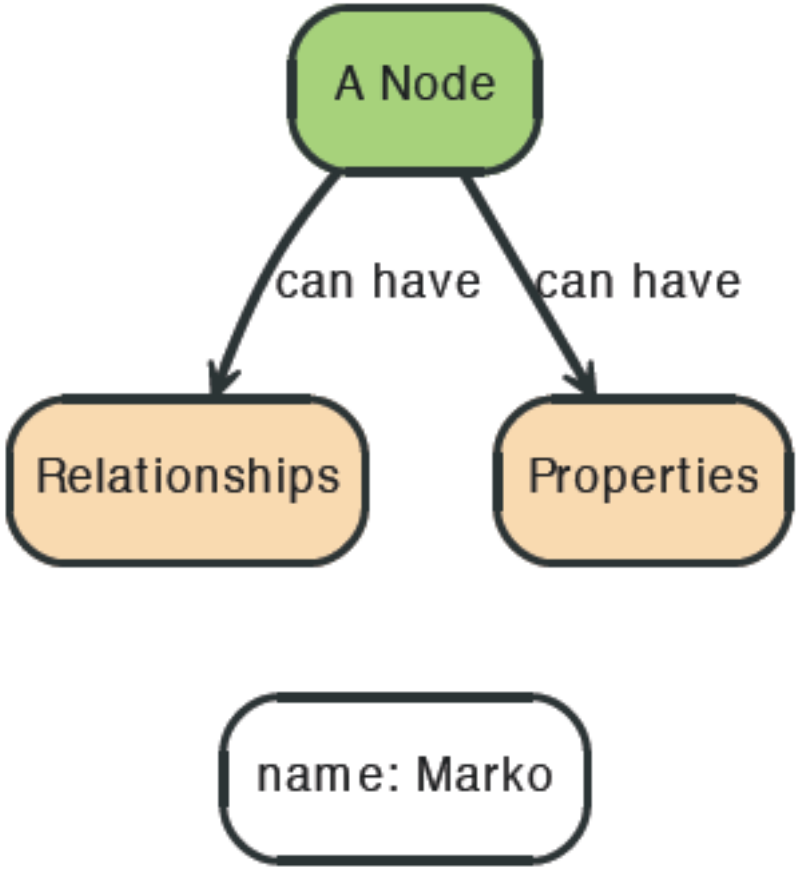


(Bachman, 2013, p.11)

Neo4j Tips

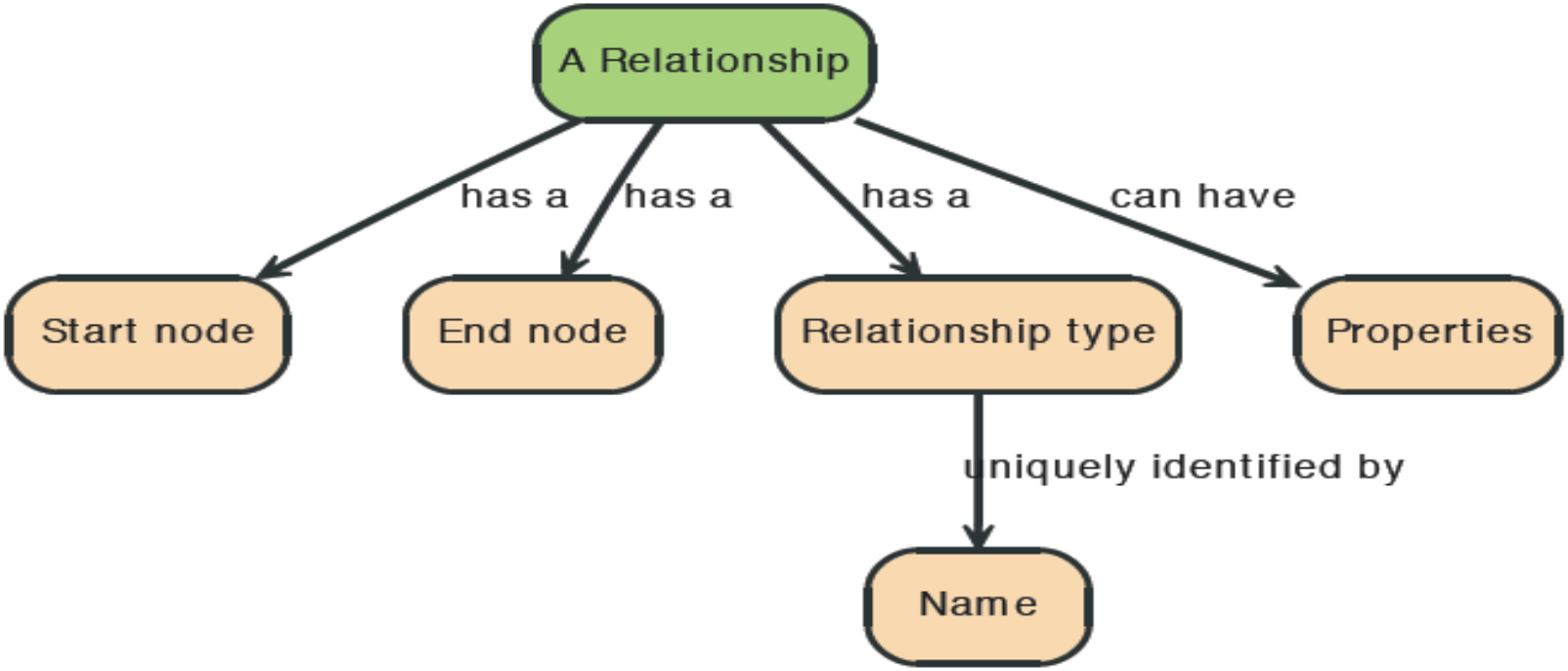
- Each entity table is represented by a label on nodes
- Each row in a entity table is a node
- Columns on those tables become node properties.
- Join tables are transformed into relationships, columns on those tables become relationship properties

Node in Neo4j

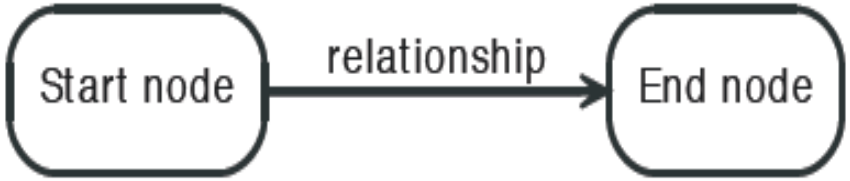


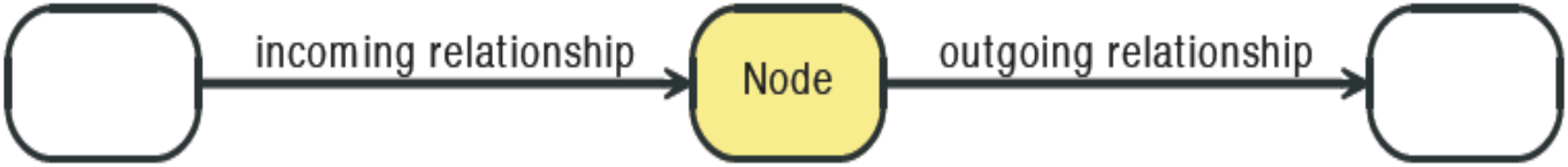
Relationships in Neo4j

- Relationships between nodes are a key part of Neo4j.

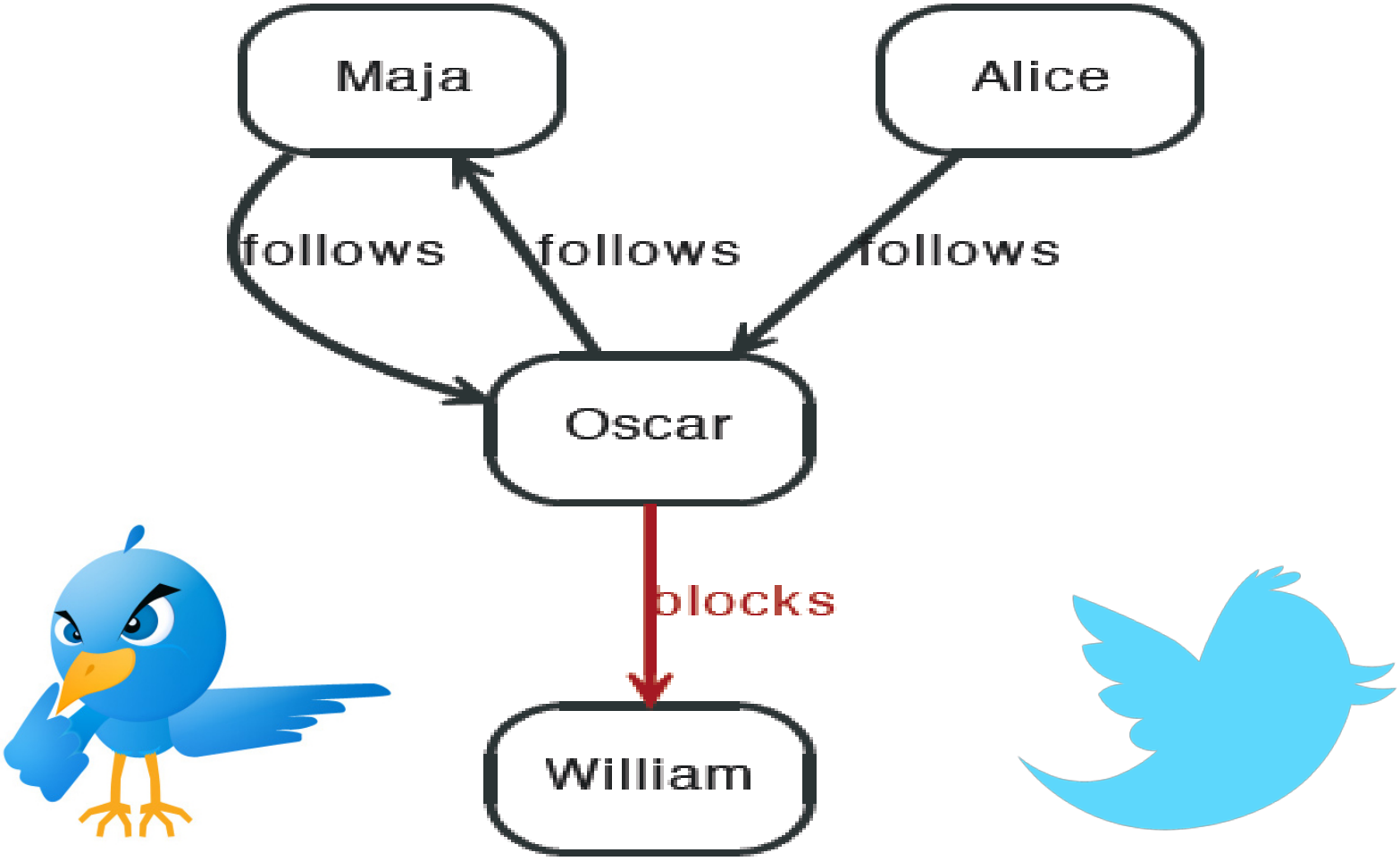


Relationships in Neo4j





Twitter and relationships

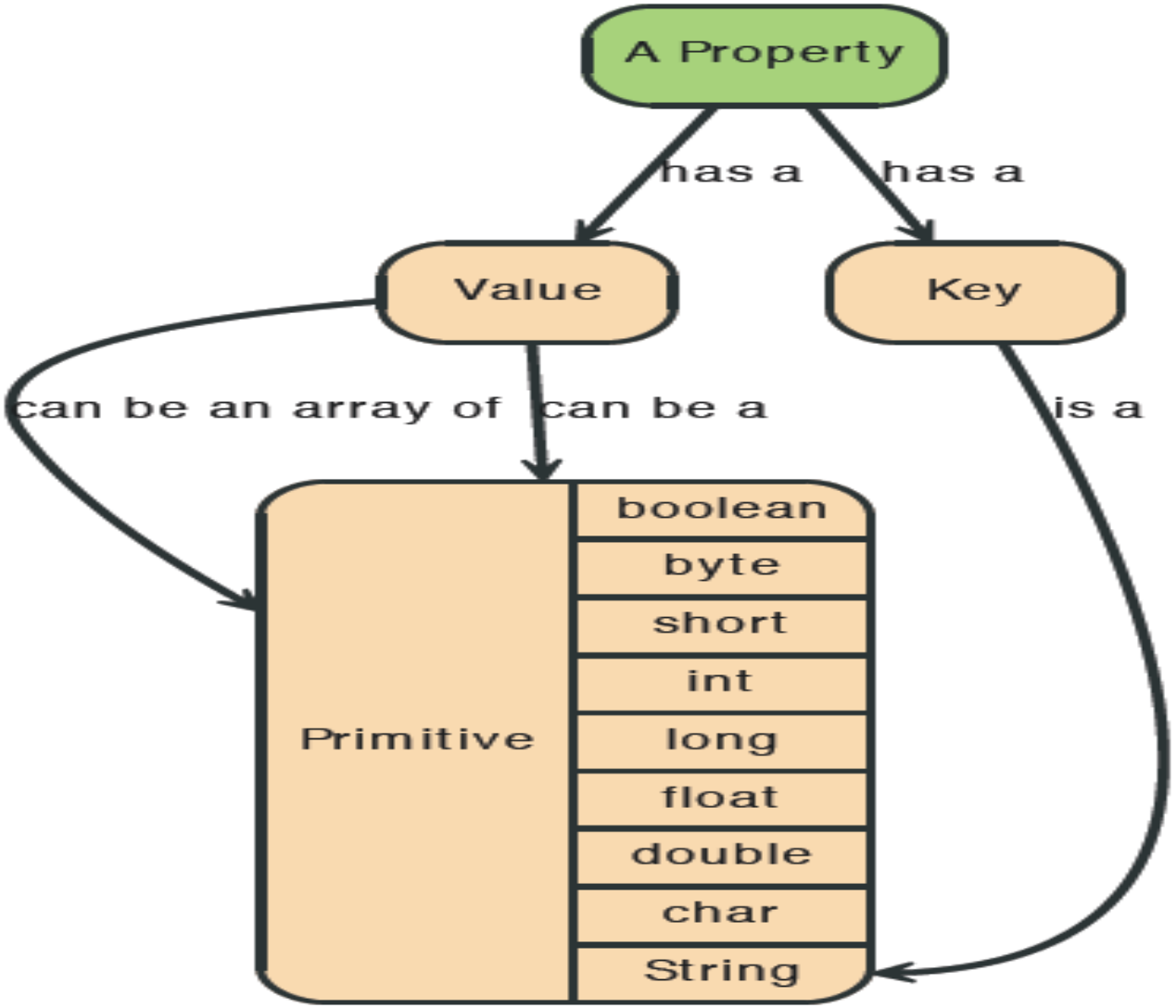


Properties

- Both nodes and relationships can have properties.
- Properties are key-value pairs where the key is a string.
- Property values can be either a primitive or an array of one primitive type.

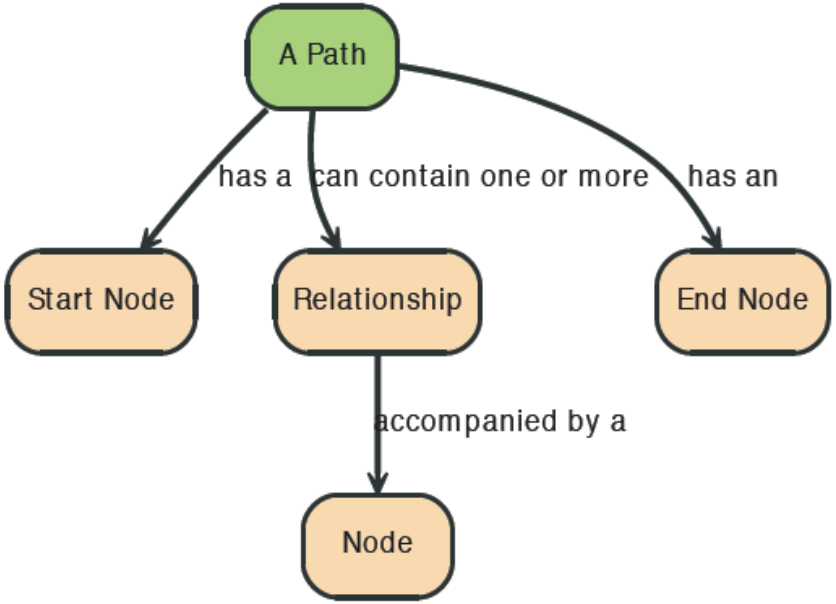
  For example String, int and int[] values are valid for properties.

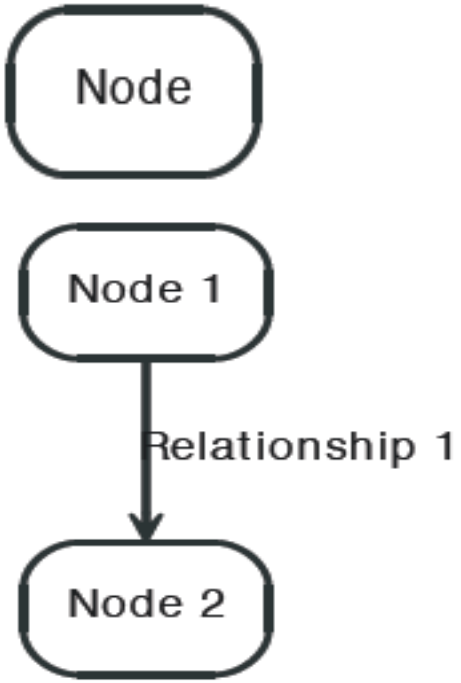
Properties


A Property
has a
has a
Value
Key
can be an array of
can be a
is a
Primitive
boolean
byte
short
int
long
float
double
char
String

Paths in Neo4j

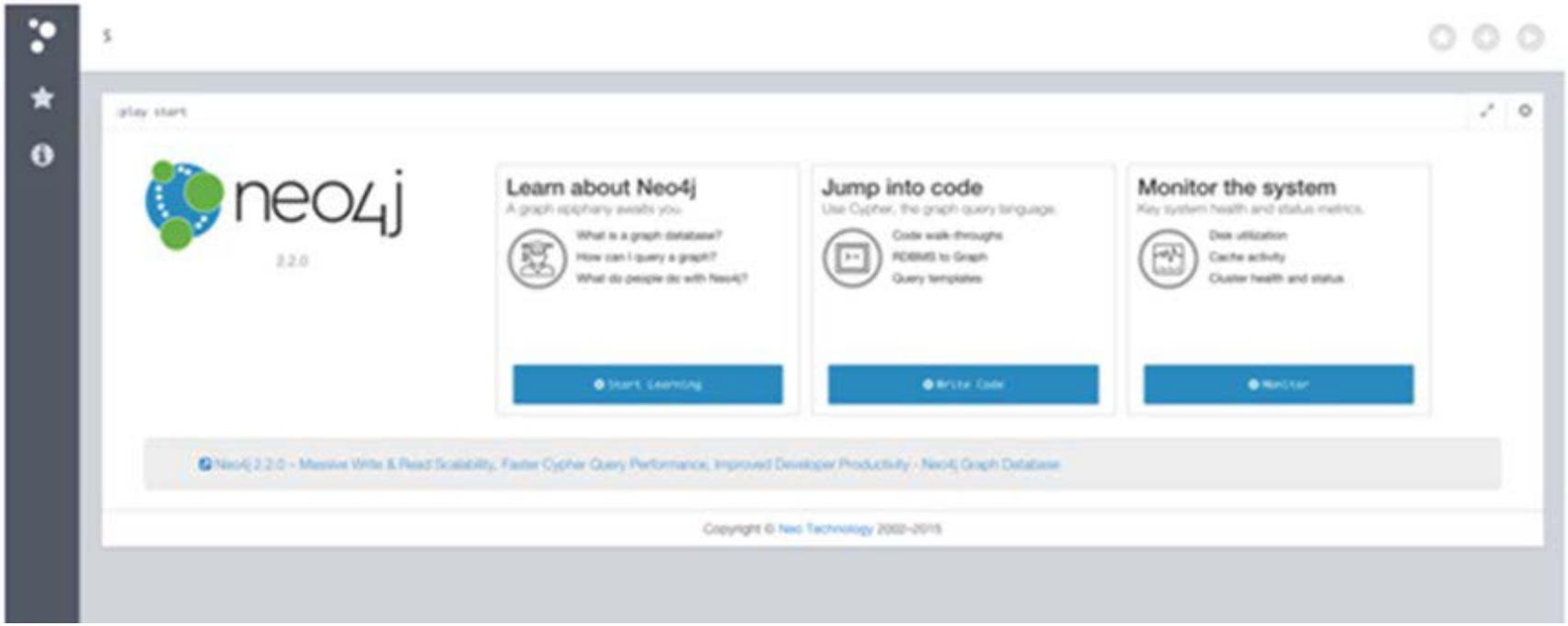
- A path is one or more nodes with connecting relationships, typically retrieved as a query or traversal result.





Neo4j browser

After you've downloaded and unzipped the Neo4j package, cd into the Neo4j directory and start up the server like this:
$ bin/neo4j start
To make sure you're up and running, try curling this URL:
$ curl http://localhost:7474/db/data/



Cypher

- Query Language for Neo4j
- Easy to formulate queries based on relationships
- Many features stem from improving on pain points with SQL such as join tables

(Hunger, Michael 2013)

Cypher basic syntax

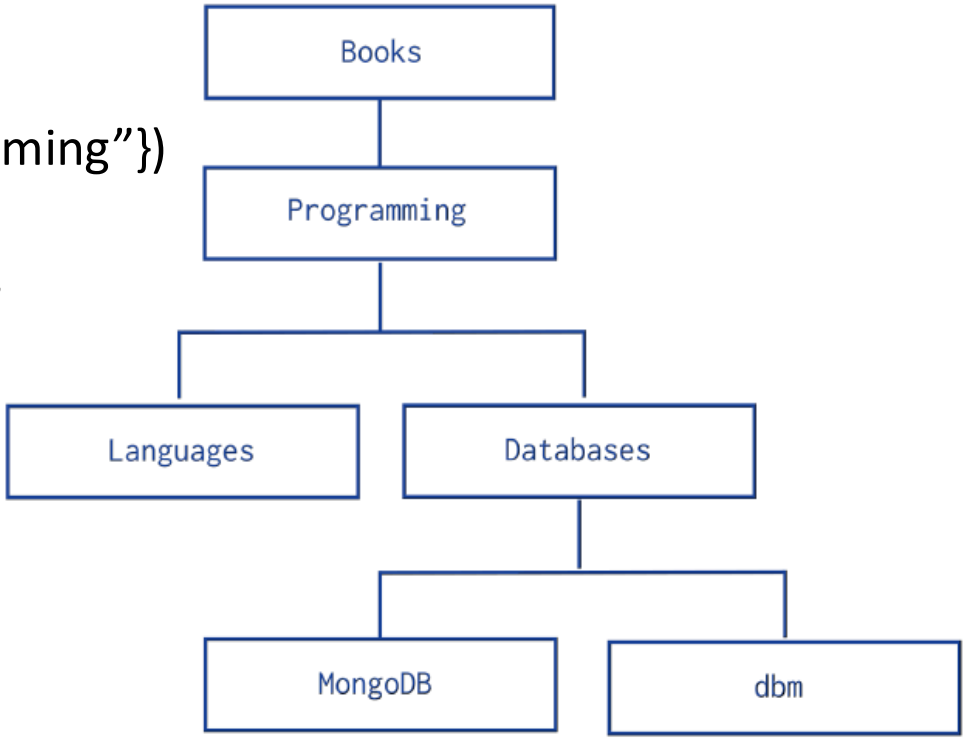
- () Node
- {} Properties
- [] Relationships

- MATCH (n) RETURN n;
- Returns every node in the database

- MATCH (a)-[:CONNECTED]->(b) RETURN a, b;
- Returns all nodes related with the CONNECTED relationship

Collections with Tree-Like Relationships

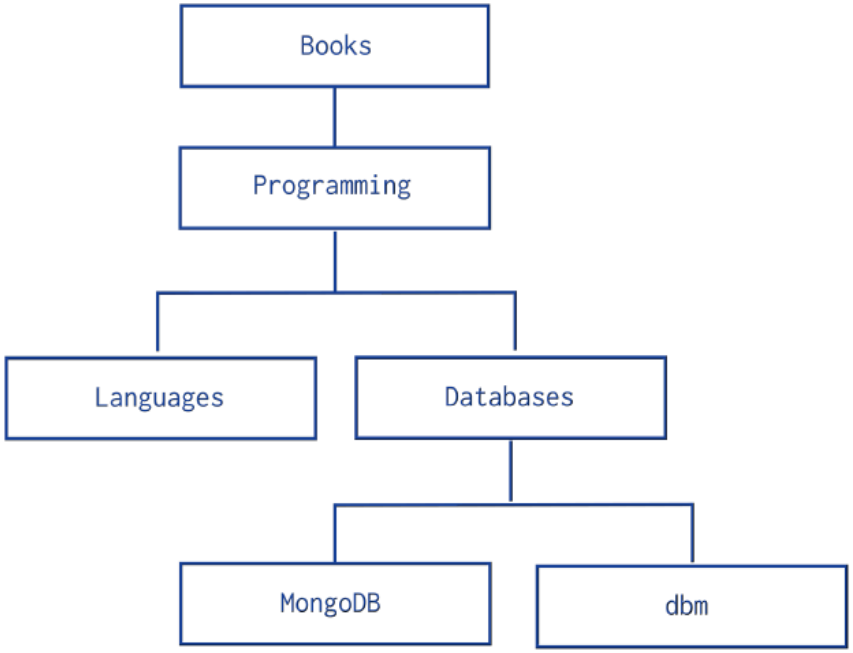
```
Create (c1:Category {name: “Books”})
Create (c2:Category {name: “Programming”})

MATCH (c:Category {name: “Books"}),
(d:Category {name: “Programming"})
CREATE (c)-[p:PARENT_OF]->(d)
```



Collections with Tree-Like Relationships

Given one node, answer queries:

- Report the parent node
  Parent of "Programming"
- Report the children nodes
  Children of "Programming"
- Report the ancestors
  Ancestors of "MongoDB"
- Report the descendants
  Descendants of "Programming"
- Report the siblings
  Siblings of "Databases"



Categories example

Children of “Programming”

```
MATCH (p:Category {name: “Programming”)-[:PARENT_OF]->(c)
RETURN p,c
```

Parent of “Programming”

```
MATCH (p:Category {name: “Programming”)<-[:PARENT_OF]- (c)
RETURN p,c
```

Ancestors of “MongoDB”

```
MATCH (p:Category)-[:PARENT_OF*]->(c:Category{name: “MongoDB”})
RETURN p,c
```

Descendants of “Programming”

```
MATCH (p:Category {name: “Programming”)-[:PARENT_OF*]->(c)
RETURN p,c
```

Siblings of “Databases”

```
MATCH (p:Category {name: "Databases"})<-[r:PARENT_OF]-(c),
(c:Category)-[q:PARENT_OF]->(d)
RETURN d
```

For today…

- Create three nodes:
  - One for you and two other friends
  - Connect them with a FRIEND relationship
  - Query the three nodes, and submit a png of the graph view to ICON