

IN3020/4020 – Database Systems

Spring 2021, Week 17.2

Semantic DBMS (Part 1)

Egor V. Kostylev (with M. Naci Akkøk)

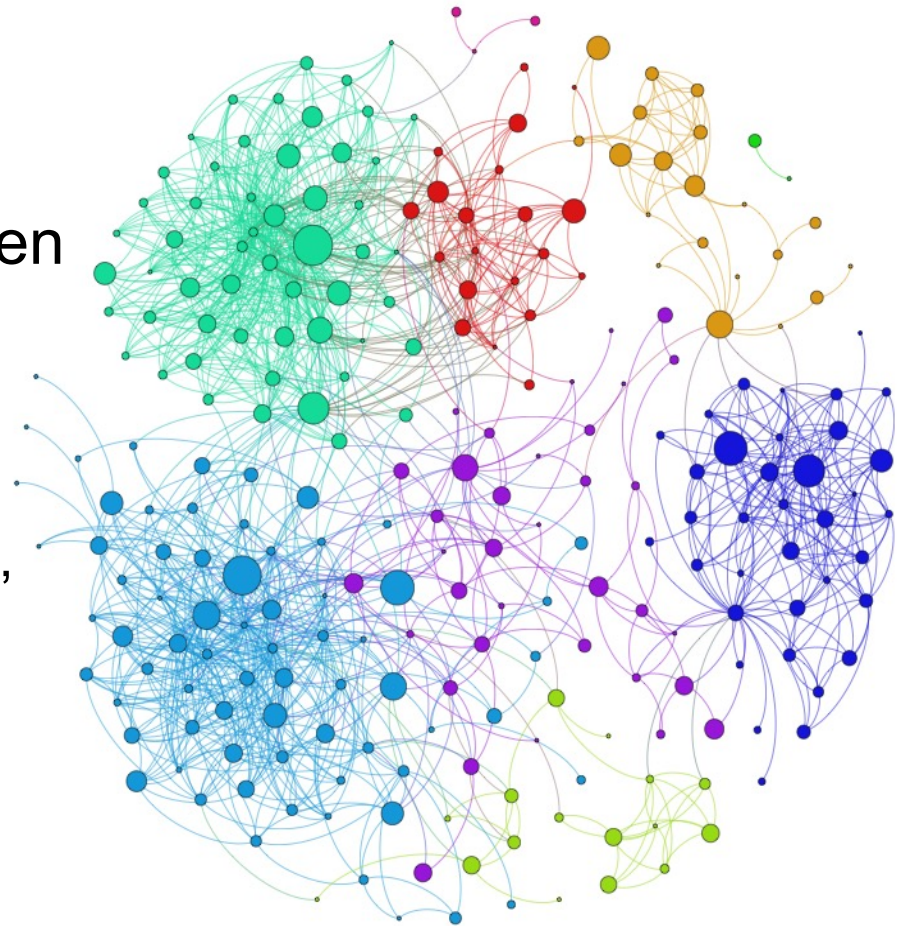
Based upon slides by D. Roman from Spring 2019



1. Overview of (knowledge) graphs

Why graphs?

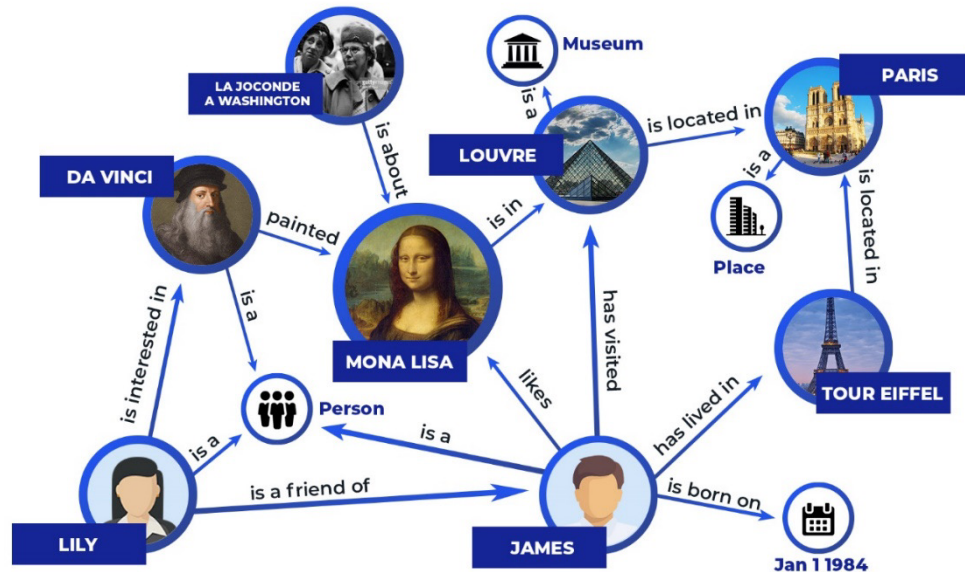
- **Universal mechanism** for describing **complex data**
- **Shared terminology** between disciplines
- **Many data are graphs**
 - Social media, economic networks, the Web, molecules, networks of neurons, etc.



Picture taken from <http://allthingsgraphed.com/public/images/linkedin/linkedin-network.png>.

What are Knowledge Graphs?

- Part of the **Knowledge Representation and Reasoning (KRR)** branch of AI
- Capture **data (facts)** and **semantics (metadata and rules)**
- **Nodes** connected by **relationships**



Why Knowledge Graphs now?

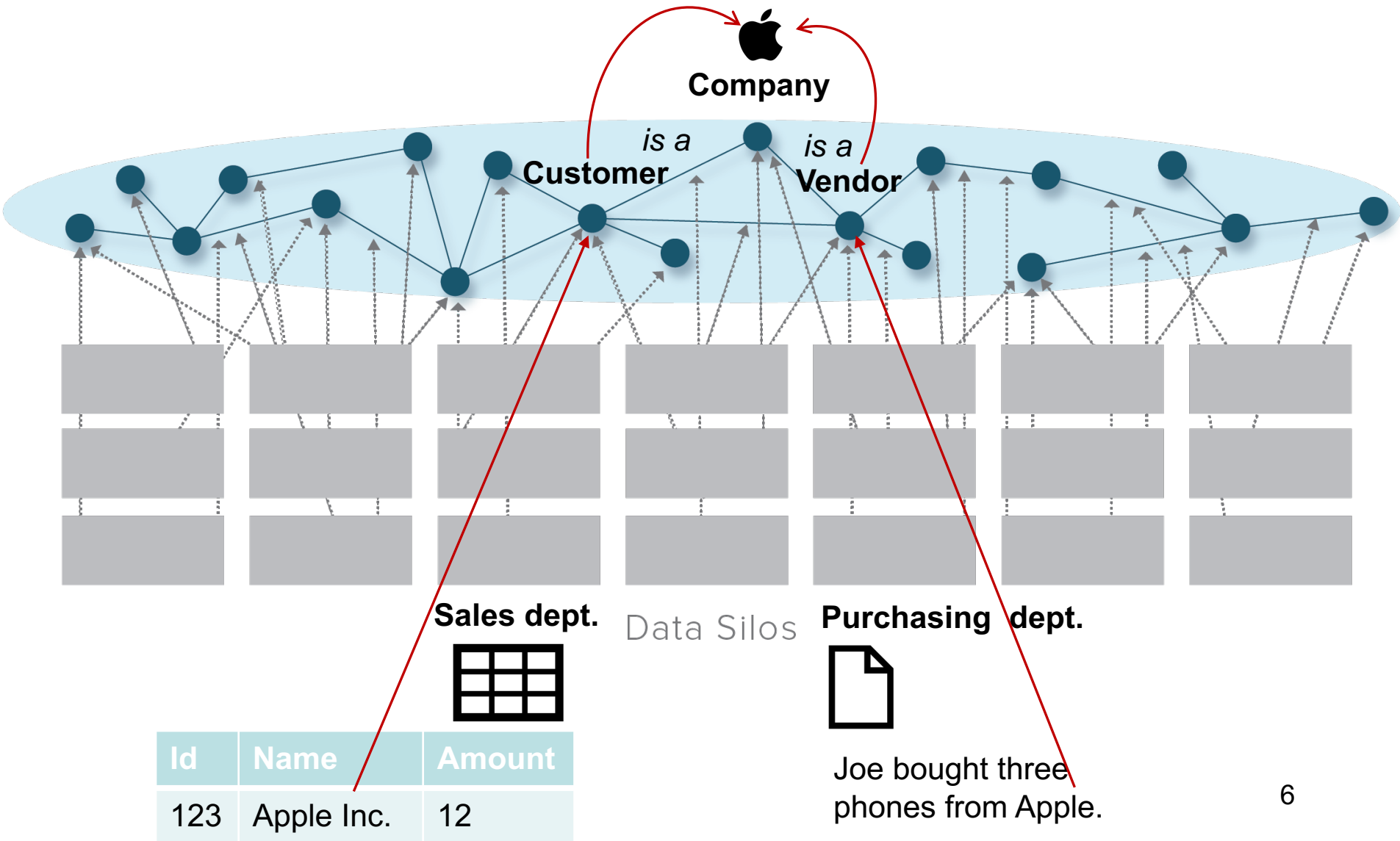
Gartner Top 10 **Data and Analytics** Technology Trends for 2019

Source: Gartner (February 2019)

- | | |
|----------------------------------|---------------------------------|
| 1. Augmented Analytics | 6. Data Fabric |
| 2. Augmented Data Management | 7. Explainable AI |
| 3. NLP/ Conversational Analytics | 8. Blockchain in Data Analytics |
| 4. Graph analytics | 9. Continuous Intelligence |
| 5. Commercial AI and ML | 10. Persistent Memory Servers |

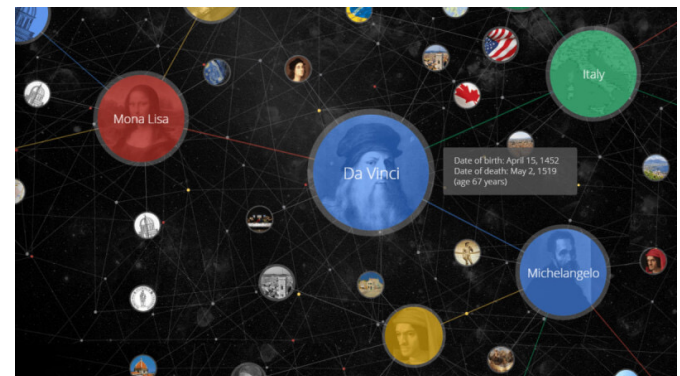
- Still there in 2020 and 2021 (in some form)
- New generation of Web and Enterprise applications
- Advances in NoSQL graph databases
- Enhanced learning

Enterprise Knowledge Graphs: data integration



Google Knowledge Graph

- *“A huge knowledge graph of interconnected entities and their attributes”.*
Amit Singhal, Senior Vice President at Google
- *“A knowledge base used by Google to enhance its search engine’s results with semantic-search information gathered from a wide variety of sources”*
http://en.wikipedia.org/wiki/Knowledge_Graph
- Used to answer direct spoken questions in Google Assistant and Google Home voice queries
- Based on information derived from many sources (e.g., *Wikidata, Wikipedia*)
- Contains over **70 billion facts** and Objects



Google Knowledge Graph (cont')

Entity Search and Summarization



oslo

All Images Maps News Videos More Settings Tools

About 400,000,000 results (0.84 seconds)

Oslo - Wikipedia

<https://en.wikipedia.org/wiki/Oslo>

Oslo is the capital and most populous city of Norway. It constitutes both a county and a municipality. Founded in the year 1040 as Ánslo, and established as a ...

Area code(s): (+47) 00 Elevation: 23 m (75 ft)

Country: Norway Established: 1048

History of Oslo's name · Oslo Metro · Oslo City · University of Oslo

Oslo, Norway - Official travel guide

<https://www.visitoslo.com>

Official travel guide for Oslo with updated info on hotels and accommodation, map, tourist information, congress, attractions, activities, concerts.

Top things to do in Oslo



The Vigeland Park
Park & museum of
Vigeland's sculpture



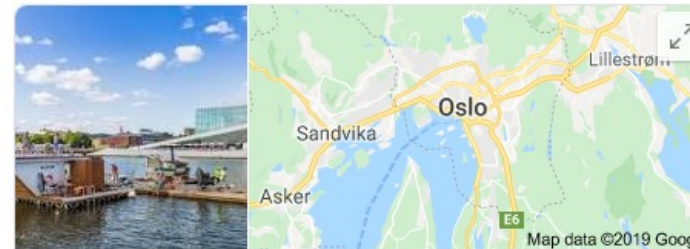
Viking Ship
Museum
Museum with 3 9th-
century Viking ships



Akershus Fortress
Waterside fortress &
former prison



The Royal Palace
Royal residence
open to public in...



Oslo

Capital of Norway

Oslo, the capital of Norway, sits on the country's southern coast at the head of the Oslofjord. It's known for its green spaces and museums. Many of these are on the Bygdøy Peninsula, including the waterside Norwegian Maritime Museum and the Viking Ship Museum, with Viking ships from the 9th century. The Holmenkollbakken is a ski-jumping hill with panoramic views of the fjord. It also has a ski museum.

Weather: 0°C, Wind NE at 4 m/s, 61% Humidity

Local time: Tuesday 11:59

District: Østlandet

Population: 673,469 (2018) Eurostat

Postal code: 0001 – 1299






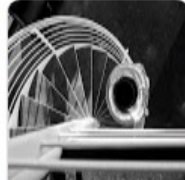



Mayor: Marianne Borgen (SV)

Google Knowledge Graph (cont')

Google Discovering related Entities

All Maps Images News Videos More Settings Tools

Oslo / Colleges and Universities

-  University of Oslo
-  OsloMet – storbyuniver...
-  BI Norwegian Business School - Osl...
-  Oslo University College
-  Oslo School of Architecture and Design
-  Norwegian Academy of Music
-  Oslo National Academy of the Arts
-  Norwegian Police University C...
-  Westerdals Oslo ACT

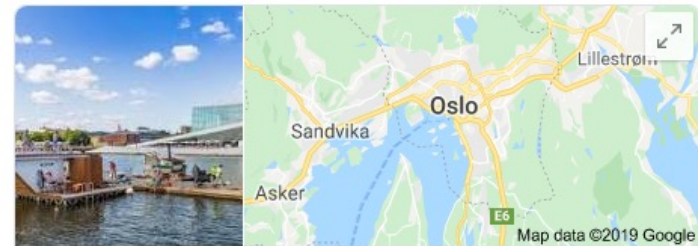
List of universities in Norway - Wikipedia

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

This list of **universities** in Norway presents the country's **universities**, giving their locations, abbreviated titles (in **Norwegian**), and years of establishment. Most **universities** in Norway are public. Most of the **university colleges** were created in 1994, following the **university ... Norwegian Police University College · Oslo, Bodø, Public, University college ...**

People also ask

Do universities in Norway teach in English?



Oslo

Capital of Norway

Google Knowledge Graph (cont')

Factual Answers

Google


when was oslo founded?

All Images Maps News Shopping More Settings Tools

About 12,600,000 results (0.67 seconds)


Oslo / Established

1049



According to the Norse sagas, **Oslo** was **founded** around 1049 by Harald Hardrada. Recent archaeological research however has uncovered Christian burials which can be dated to prior to AD 1000, evidence of a preceding urban settlement.


[Oslo - Wikipedia](#)
<https://en.wikipedia.org/wiki/Oslo>



Oslo
Capital of Norway

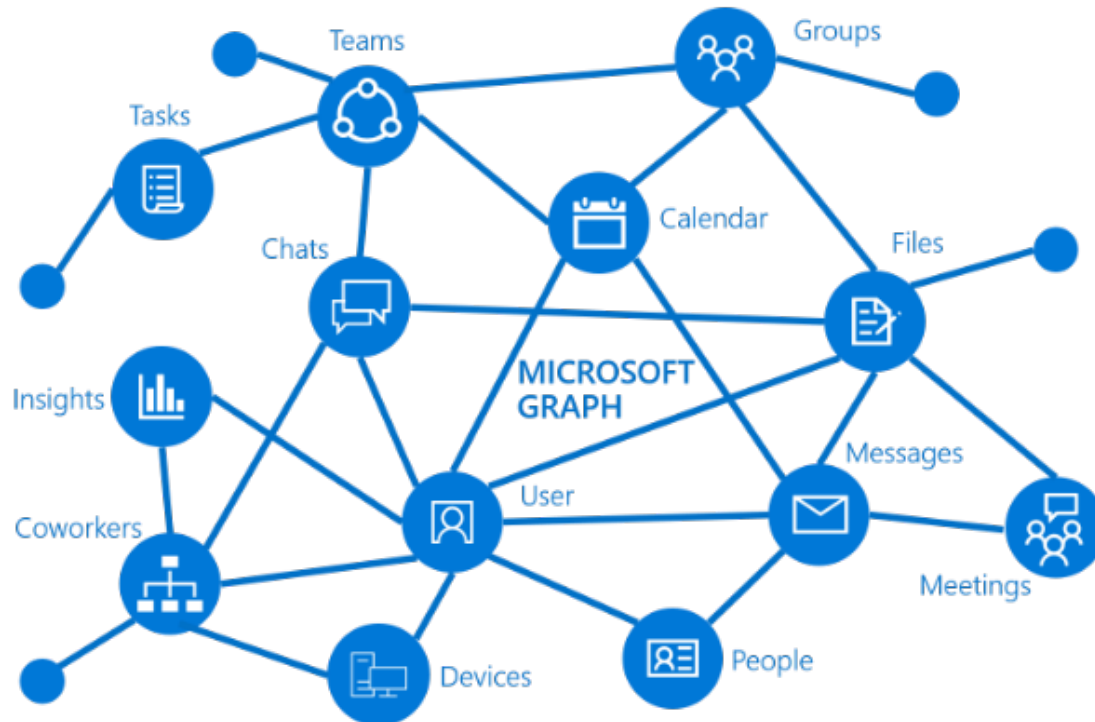
Weather: 0°C, Wind NE at 4 m/s, 60% Humidity
Local time: Tuesday 12:03

Plan a trip

 Oslo travel guide

Who else is building and using Enterprise KGs?

- Microsoft, Siemens, LinkedIn, Airbnb, eBay, Apple, and many others

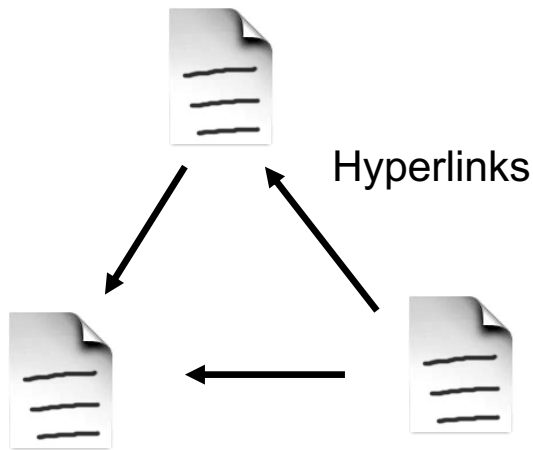


Picture taken from <https://docs.microsoft.com/en-us/graph/overview>

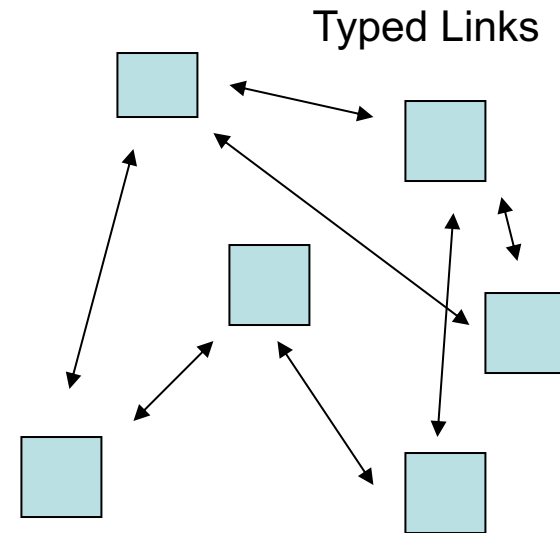
Enterprise KGs are (usually) proprietary

- Enterprise KGs are typically a **closed** implementation of ***Web of Data / Linked Data*** principles

Web of Documents  **Web of Data / Linked Data**



“Documents”



“Things”

Linked Data, Semantic Knowledge Graphs

- Method for **publishing data on the Web**
- **Self-describing** data and relations
- **Interlinking**
- Accessed using **semantic queries**



<http://www.w3.org/standards/semanticweb/data>

- A set of open standards developed by W3C
 - Data format: RDF
 - Knowledge representation: RDFS/OWL
 - Query language: SPARQL



Example RDF Graph

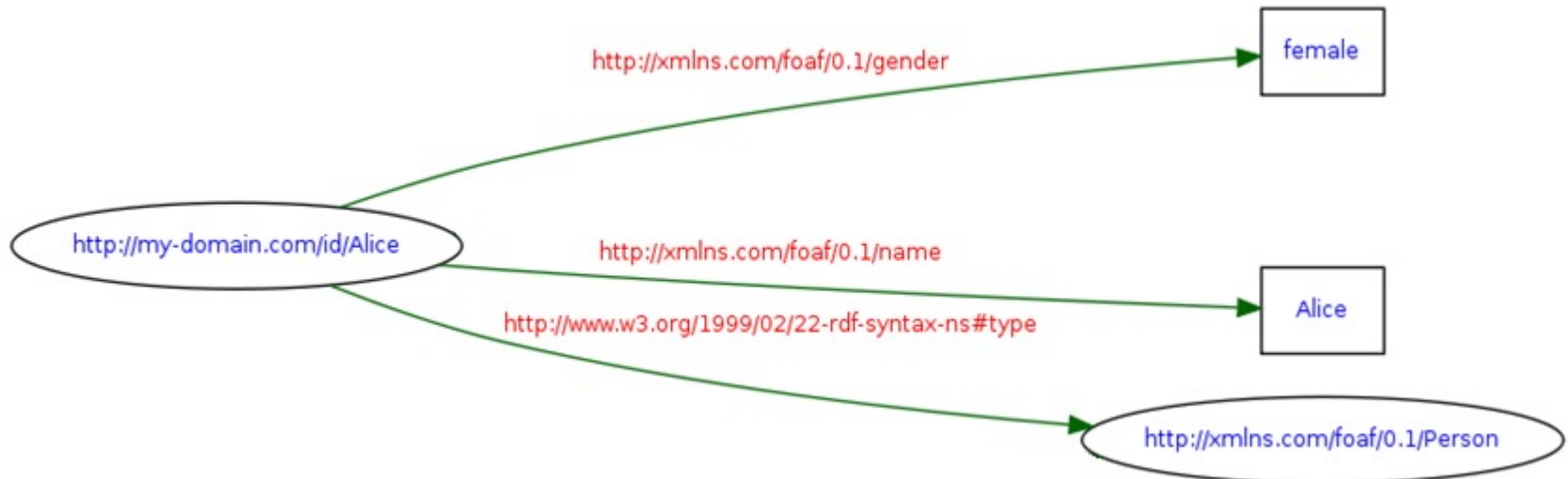
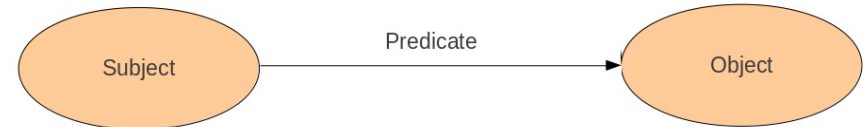
“triple” = (subject, predicate, object)

e.g.

(Alice has gender “female”)

(Alice has name “Alice”)

(Alice is a Person)

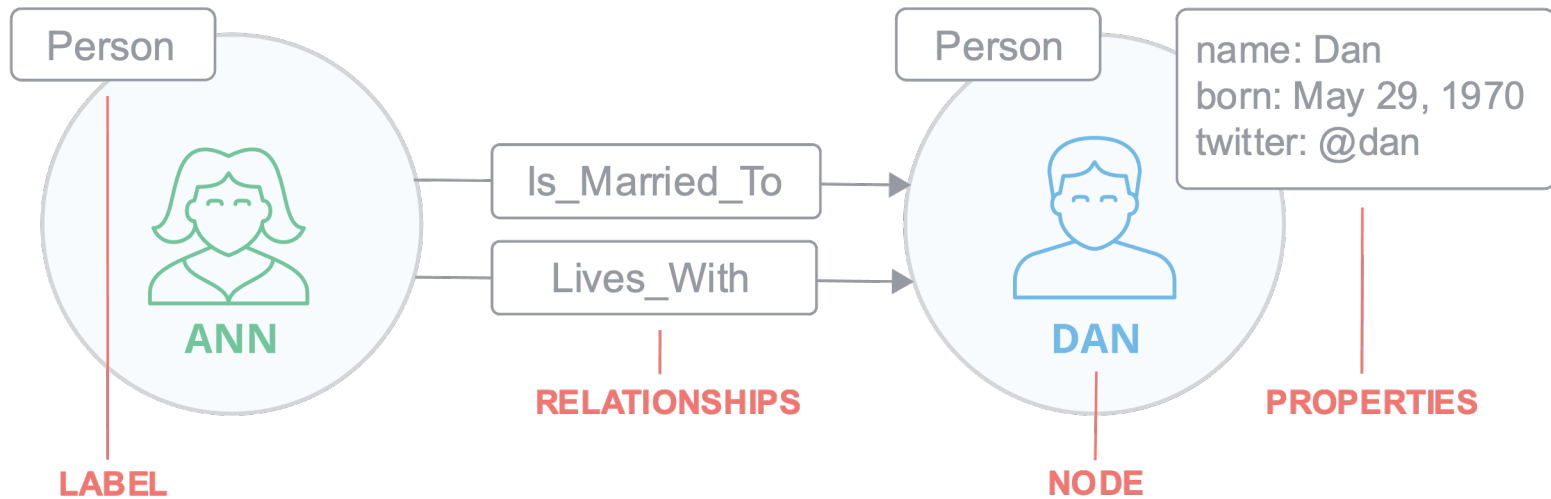


Logical entailment with Linked Data (RDFS/OWL)

(John is a student) entails (John is a person)
(Student kind of person)

(has name applies to students) entails (Mary is a student)
(Mary has name “Mary”)

Labeled Property Graphs (e.g., Neo4j)



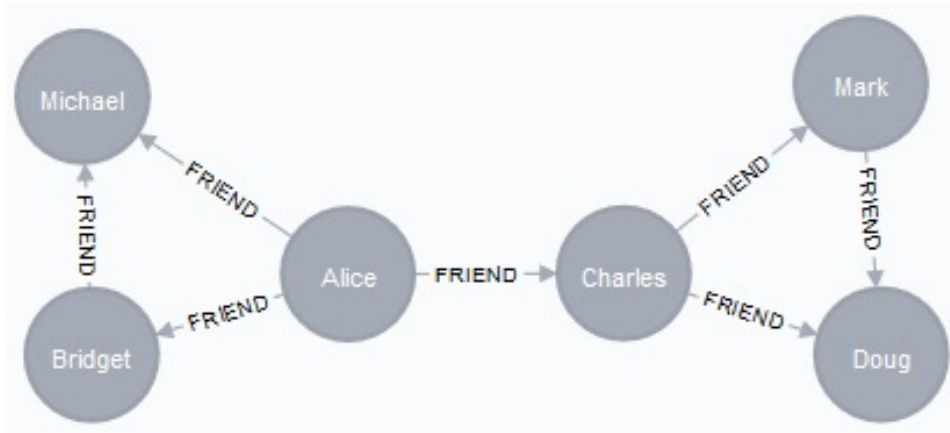
- **Node, Relationship, Property**
- **Label, Relationship type**
- Some constraints, e.g., a node can have none or several labels, each relationship has exactly one type, etc.

Graph Algorithms

- **Centralities:** determine the importance of distinct nodes in a network
(PageRank, Betweenness Centrality, Closeness Centrality)
- **Community detection:** evaluate how a group is clustered or partitioned, as well as its tendency to strengthen or break apart
(Louvain, Label Propagation, Connected Components, Triangle Count / Clustering Coefficient)
- **Path finding:** find the shortest path or evaluate the availability and quality of routes
(Minimum Weight Spanning Tree, All Pairs- and Single Source - Shortest Path, A* Algorithm, Yen's K-Shortest Paths, Random Walk)

Example: Louvain

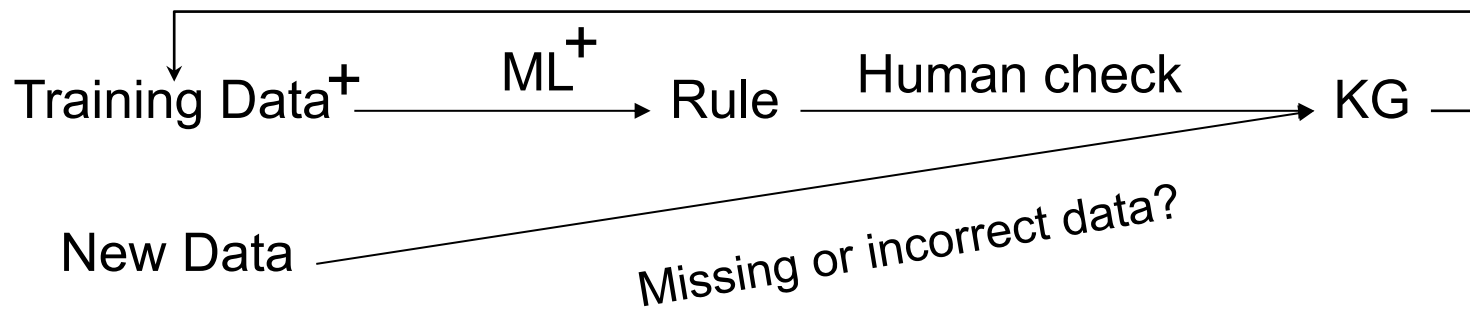
- Used for detecting communities in networks
- Evaluates how much more densely connected the nodes within a community are, compared to how connected they would be in a random network



Name	Community
Alice	0
Bridget	0
Michael	0
Charles	1
Doug	1
Mark	1

Knowledge Graphs and ML

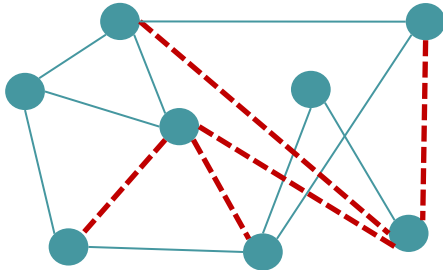
- Examples of classical ML tasks on graphs
 - Node classification: Predict a type of a given node
 - Link prediction: Predict whether two nodes are linked
 - Community detection: Identify densely linked clusters of nodes
 - Network similarity: How similar are two (sub)networks
- ML algorithms can use KGs — both, as input and as output



Opportunities for combining KGs and ML

Link prediction

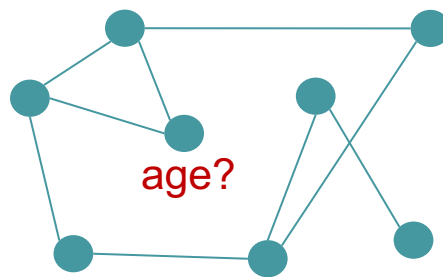
(1-1, 1-n, n-m)



Attribute prediction

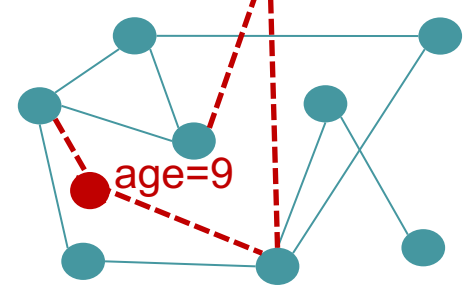
age=63

age=65

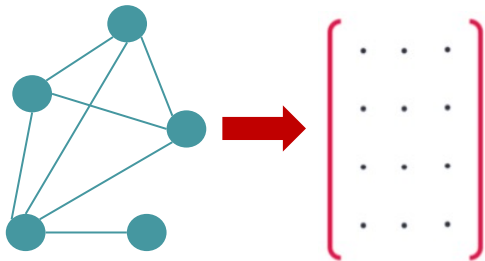


Subgraph prediction

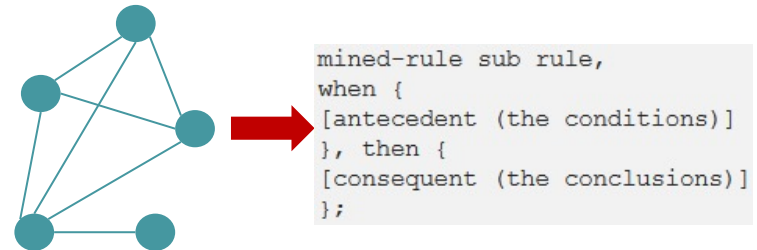
age=64



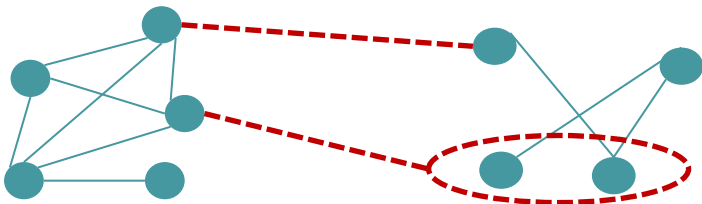
KG embeddings



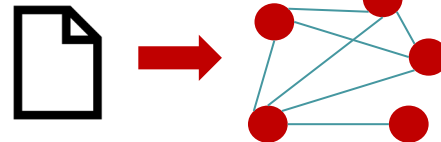
Rule mining



Graph matching



Automated KG creation from text



Optimal path finding on KGs

Query answering on KGs

....

Common tasks when working with KGs

Designing KGs

- How do we represent the structure of graph?
- Manually designed or (semi-) automated/learned?

Populating KGs with data

- Data pre-processing/transformation/reconciliation
- Data validation

Storing/Accessing KGs data

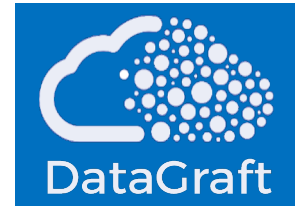
- Big data infrastructure
- Query languages

Analyzing KGs data

- Deriving new knowledge (inference, learning)
- Insights/Analytics

Building KGs applications

- Search, data integration, etc.



<https://datagraft.io>

Summary

- KGs is an emerging technology
 - Various types of KGs, e.g. semantic and property graphs
 - KGs are becoming popular in Enterprise applications
- Large and small companies are developing and using KGs
 - Research projects
 - Graph databases are maturing
- KGs and ML complement each other

2. RDF Graphs

Topics:

- Resource Description Framework (RDF)
- RDF Schema (RDFS) + little bit of OWL
- SPARQL: RDF query language

- All are W3C standards

Intro to Resource Description Framework (RDF)

(Most of the examples in the upcoming slides are taken from: <http://www.w3.org/TR/rdf-primer/>)

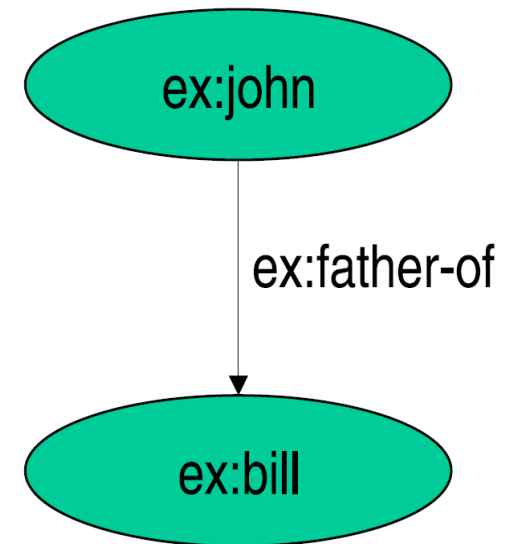
- RDF is a language that enable to describe making statements on resources
 - John is father of Ann
- RDF graph is a set of triples (here in Turtle syntax)

subject1 predicate1 object1 . subject2 predicate2 object2

- **Subject:** Resource or blank node
- **Predicate:** Resource
- **Object:** Resource (or collection of resources), literal or blank node
- Often written as **<subject, predicate, object>** in texts
- Example: **<ex:john, ex:father-of, ex:bill>**
- Can be seen as a FO logical **ex:father-of(ex:john,ex:bill)** with binary predicate **ex:father-of** relating object **ex:john** to object **ex:bill**

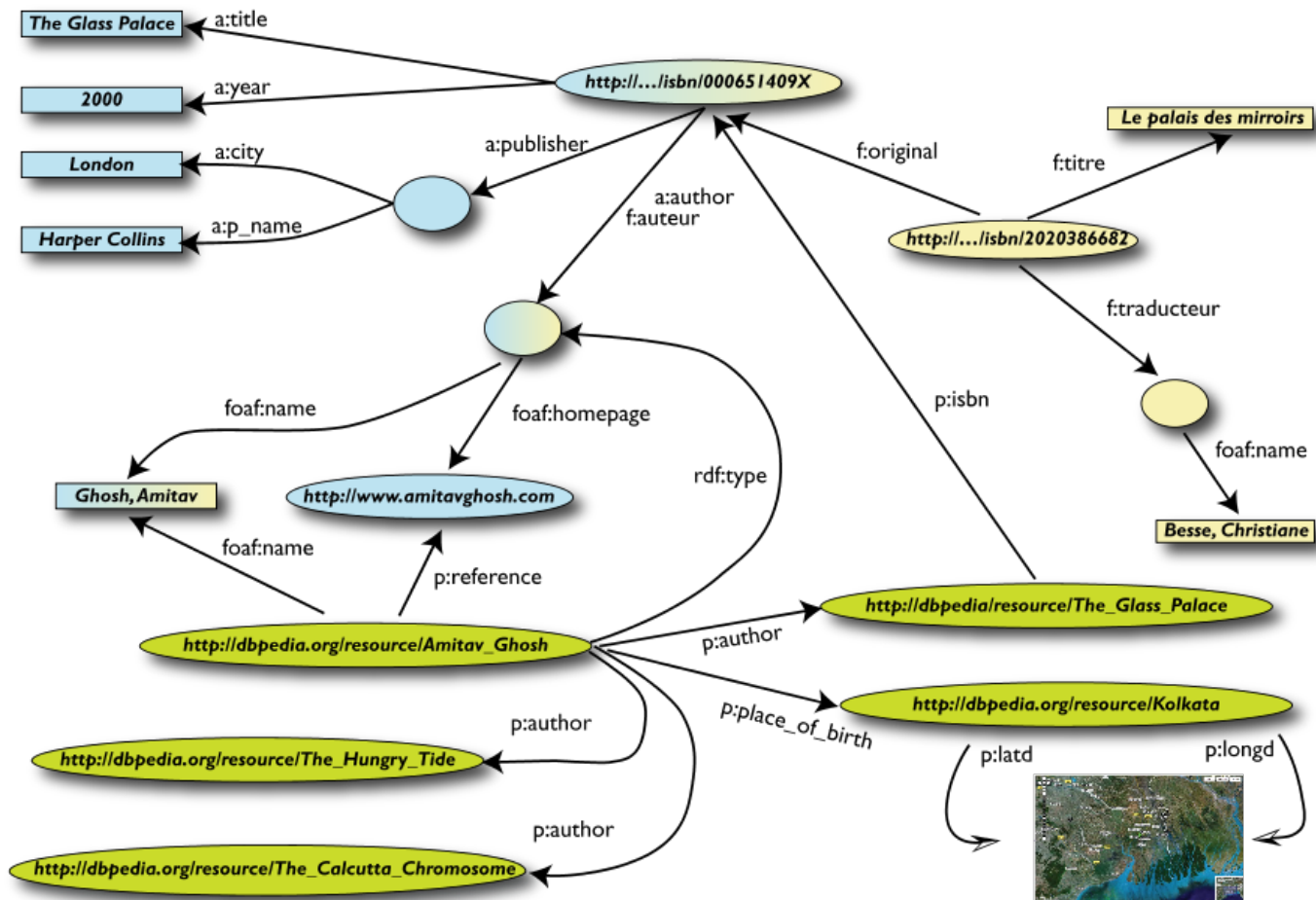
RDF Triple Graph Representation

- The triple graph is often represented as a labeled directed graph
 - **Nodes**: subjects and objects
 - **Edges**: predicates
- Such graph is called in the Artificial Intelligence community a **semantic net**
- **<ex:john, ex:father-of, ex:bill>**
- Remember the mismatch: edges can also be nodes (rare in practice)



RDF: a Direct Connected Graph-Based Model

- Different interconnected triples lead to a more complex graphic model
- Basically a RDF document is a direct connect graph

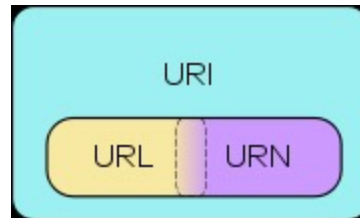


Resources

- A resource may be:
 - Web page (e.g. `http://www.w3.org`)
 - A person (e.g. `http://www.w3.org/People/Berners-Lee/`)
 - A book (e.g. `urn:isbn:4-534-34674-4`)
 - Anything denoted with a URI! (or IRI, Unicode version of URI)
- A URI is an *identifier* and **not** a location on the Web
- RDF allows making statements about resources:
 - `<http://www.w3.org/People/Berners-Lee/, http://www.w3.org/HasName, "Tim">`
 - `<urn:isbn:0-345-33971-1, http://www.w3.org/Has-Author, "John">`

URI, URN, URL

- A Uniform Resource Identifier (URI) is a string of characters used to identify a name or a resource on the Internet

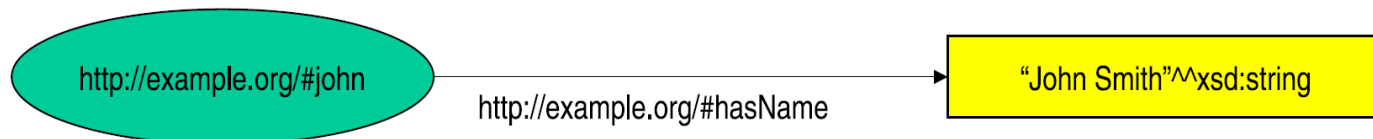


- A URI can be a URL or a URN
- A Uniform Resource Name (URN) defines an item's identity
 - the URN *urn:isbn:urn:isbn:4-534-34674-4* is a URI that specifies the identifier system, i.e. International Standard Book Number (ISBN), as well as the unique reference within that system and allows one to talk about a book, but doesn't suggest where and how to obtain an actual copy of it
- A Uniform Resource Locator (URL) provides a method for finding it
 - the URL *https://www.uio.no/studier/emner/matnat/ifi/INF1300* identifies a resource (INF1300's home page) and implies that a representation of that resource (such as the home page's current HTML code, as encoded characters) is obtainable via HTTP from a network host named *https://www.uio.no*

Literals

- Plain literals
 - E.g. "any text"
 - Optional language tag, e.g. "Hello, how are you?"@en-GB
- Typed literals
 - E.g. "hello"^^xsd:string, "1"^^xsd:integer
 - Recommended datatypes:
 - XML Schema datatypes
- Only as *object* of a triple, e.g.:

```
<http://example.org/#john,  
    http://example.org/#hasName,  
    "John Smith"^^xsd:string>
```



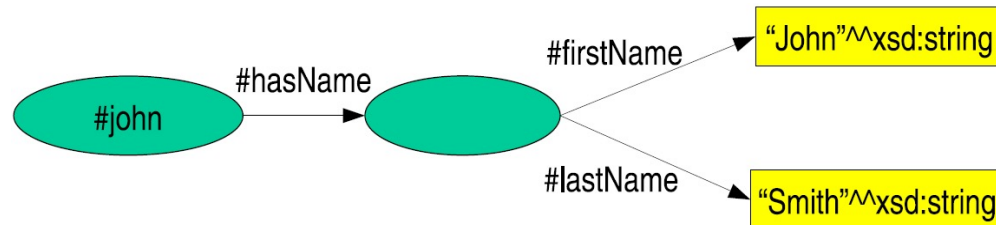
Datatypes

- One pre-defined datatype: `rdf:XMLLiteral`
 - Used for embedding XML in RDF
- Recommended datatypes are XML Schema datatypes, e.g.:
 - `xsd:string`
 - `xsd:integer`
 - `xsd:float`
 - `xsd:anyURI`
 - `xsd:boolean`

Blank Nodes

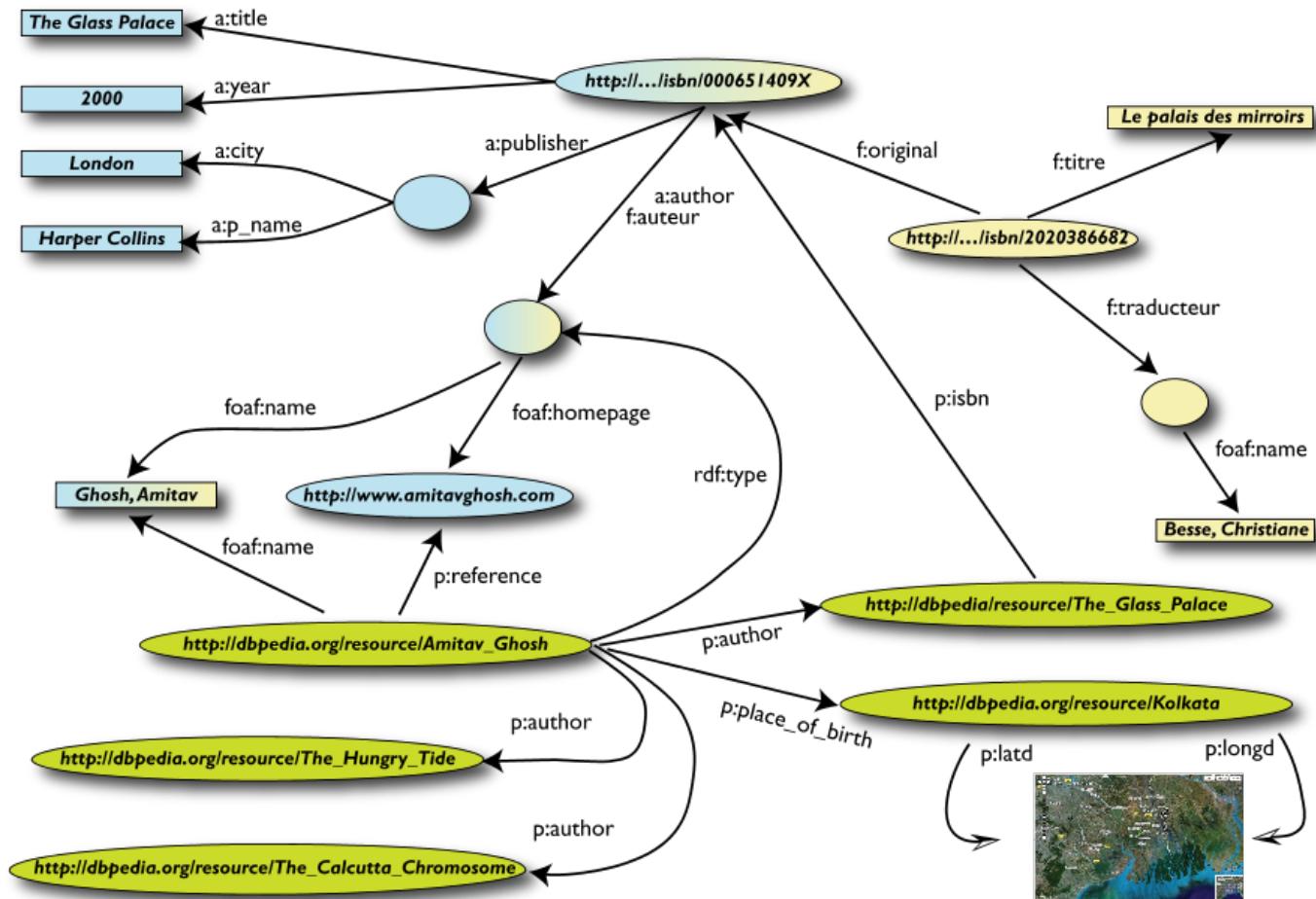
- Blank nodes are nodes without a URI
- Used for
 - Unnamed resources
 - More complex constructs (see below)
- Usually written with `_:` prefix
- Differently from SQL NULLs, there may be many
- For example:

```
<#john, #hasName, _:johnsname>  
<_:johnsname, #firstName, "John"^^xsd:string>  
<_:johnsname, #lastName, "Smith"^^xsd:string>
```



RDF: a Direct Connected Graph-Based Model

- Different interconnected triples lead to a more complex graphic model
- Basically a RDF document is a direct connect graph



Blank Nodes for Complex Constructs

- **Representation of complex data**

A blank node can be used to indirectly attach to a resource a consistent set of properties which together represent a complex data

- **Anonymous classes in OWL**

The ontology language OWL uses blank nodes to represent anonymous classes such as unions or intersections of classes, or classes called restrictions, defined by a constraint on a property

RDF Containers

- Grouping property values:

“The lecture is attended by John, Mary and Chris” **Bag**

*“[RDF-Concepts] is edited by Graham and Jeremy
(in that order)”* **Seq**

*“The source code for the application may be found at
ftp1.example.org,
ftp2.example.org,
ftp3.example.org”* **Alt**

RDF Containers 2

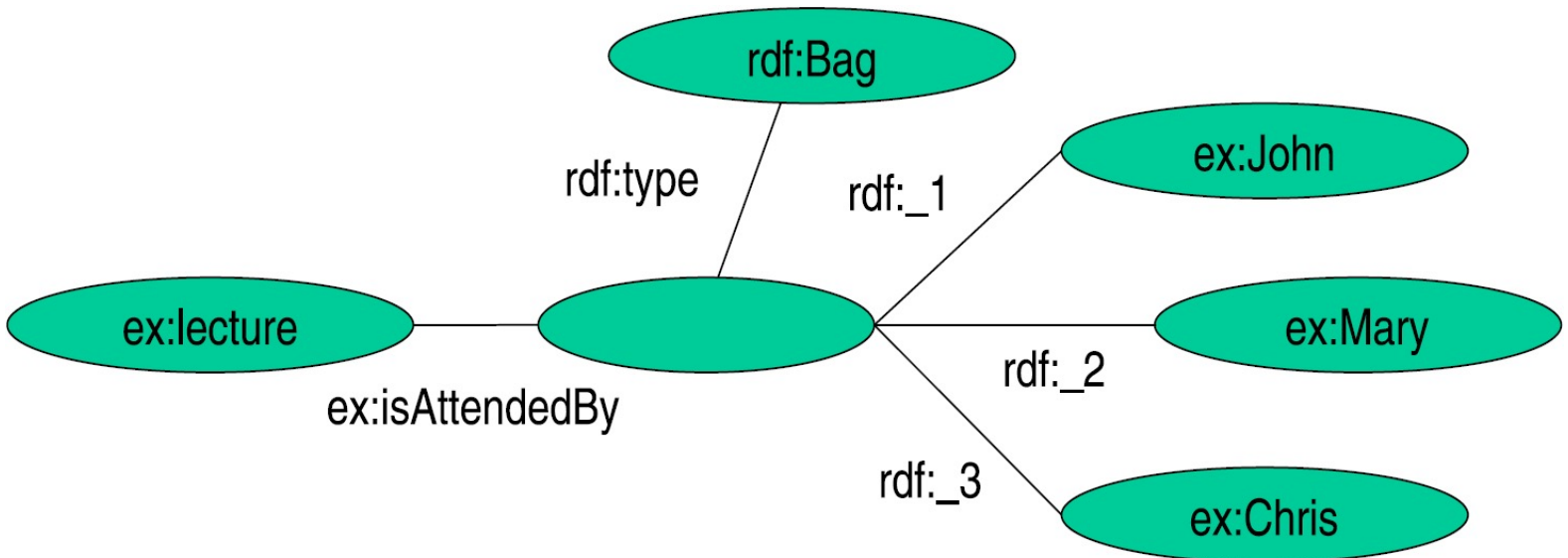
- Three types of containers:
 - `rdf:Bag` - unordered set of items
 - `rdf:Seq` - ordered set of items
 - `rdf:Alt` - set of alternatives
- Every container has a triple declaring the `rdf:type`
- Items in the container are denoted with
 - `rdf:_1, rdf:_2, . . . , rdf:_n`

RDF Containers 2

- Three types of containers:
 - `rdf:Bag` - unordered set of items
 - `rdf:Seq` - ordered set of items
 - `rdf:Alt` - set of alternatives
- Every container has a triple declaring the `rdf:type`
- Items in the container are denoted with
 - `rdf:_1`, `rdf:_2`, ... , `rdf:_n`
- Limitations:
 - Semantics of the container is up to the application
 - What about closed sets?
 - How do we know whether Graham and Jeremy are the only editors of [RDF-Concepts]?

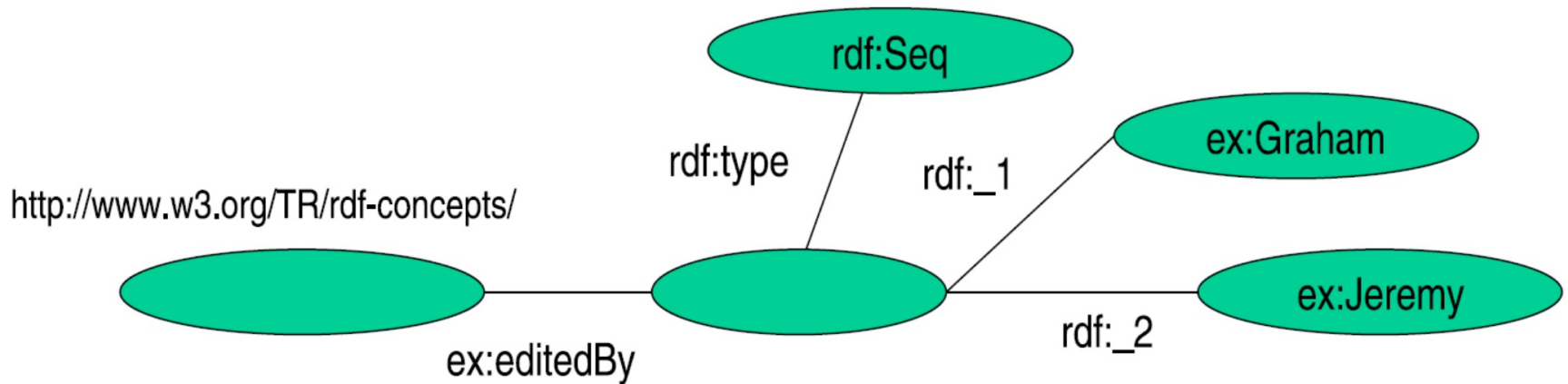
RDF Containers Graph Representation: Bag

“The lecture is attended by John, Mary and Chris”



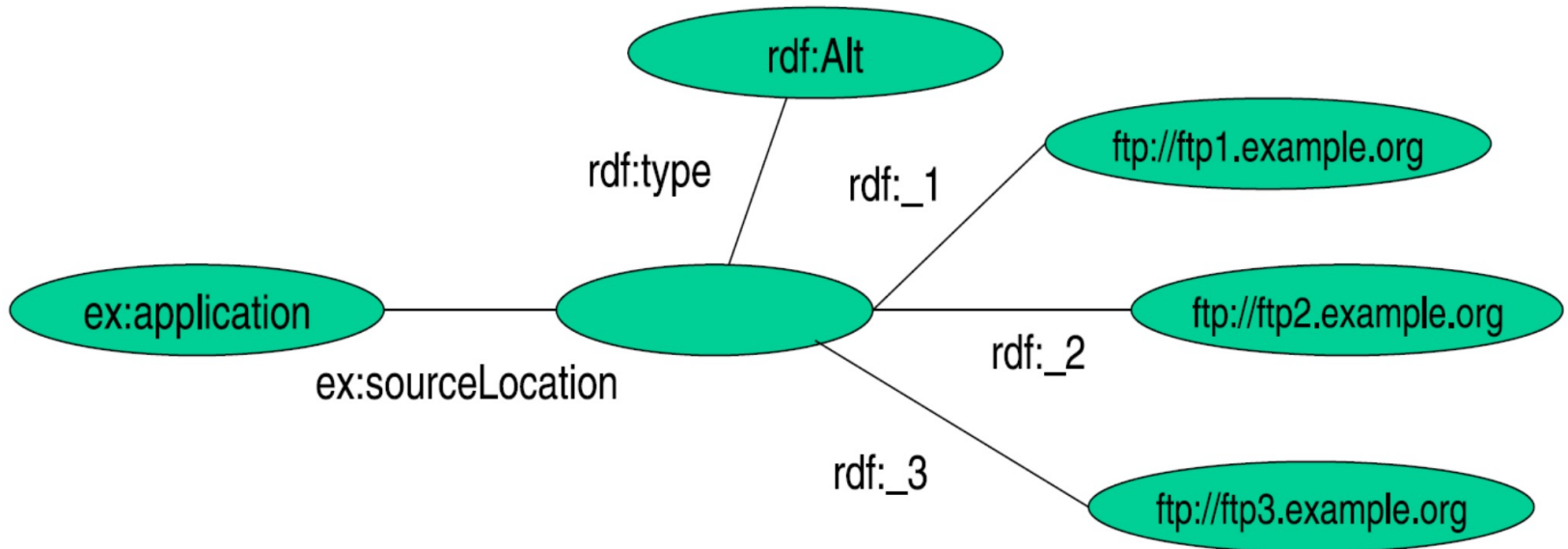
RDF Containers Graph Representation: Seq

*“[RDF-Concepts] is edited by Graham and Jeremy
(in that order)”*



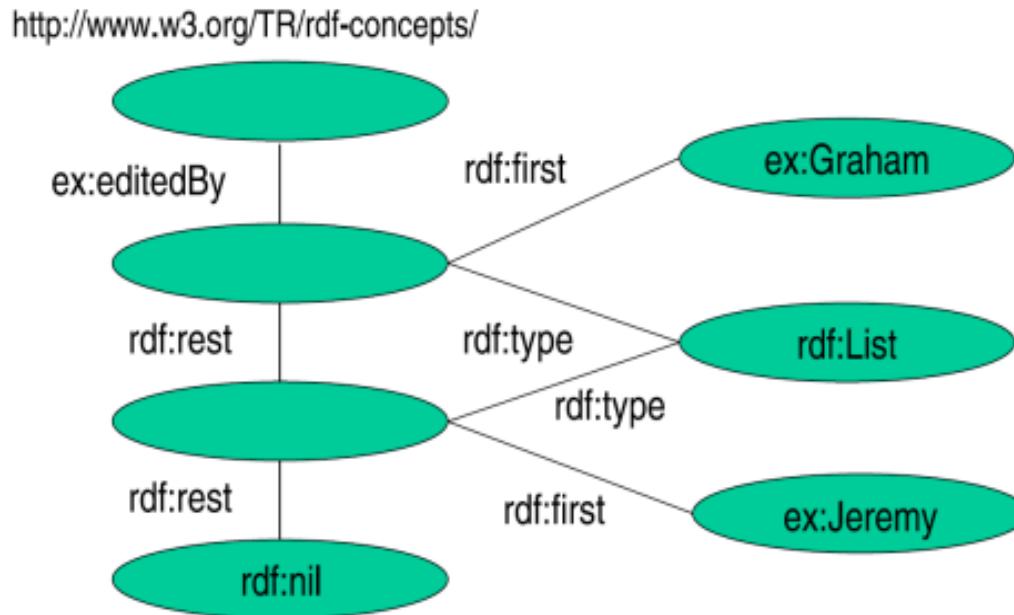
RDF Containers Graph Representation: Alt

“The source code for the application may be found at



RDF Collections

*“[RDF-Concepts] is edited by Graham and Jeremy
(in that order) and nobody else”*”



RDF provides support for describing groups containing only the specified members, in the form of RDF collections.

Reification I

- Reification: statements about statements

Mary claims that John's name is "John Smith".

```
<_:myStatement, rdf:type, rdf:Statement>  
<_:myStatement, rdf:subject, #john>  
<_:myStatement, rdf:predicate, #hasName>  
<_:myStatement, rdf:object, "John Smith">
```

This kind of statement can be used to describe belief or trust in other statements, which is important in some kinds of applications

Necessary because there are only triples in RDF: we cannot add an identifier directly to a triple (then it would be a quadruple)

Reification II

- Reification: statements about statements

Mary claims that John's name is "John Smith".

```
<_:myStatement, rdf:type, rdf:Statement>  
<_:myStatement, rdf:subject, #john>  
<_:myStatement, rdf:predicate, #hasName>  
<_:myStatement, rdf:object, "John Smith">
```



```
<#john, #hasName, "John Smith">
```

In such a way we attached a label to the statement.

Reification III

- Reification: statements about statements

Mary claims that John's name is "John Smith".

```
<_:myStatement, rdf:type, rdf:Statement>  
<_:myStatement, rdf:subject, #john>  
<_:myStatement, rdf:predicate, #hasName>  
<_:myStatement, rdf:object, "John Smith">  
  
<#mary, #claims, #myStatement>
```

RDF uses only binary properties. This restriction seems quite serious because often we use predicates with more than two arguments. Luckily, such predicates can be simulated by a number of binary predicates.

RDF Vocabulary

- RDF defines a number of resources
- We have already seen: `rdf:XMLLiteral`, `rdf:type`, ...
- RDF vocabulary is defined in the namespace:
<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
- Classes:
 - `rdf:Property`, `rdf:Statement`, `rdf:XMLLiteral`
 - `rdf:Seq`, `rdf:Bag`, `rdf:Alt`, `rdf>List`
- Properties:
 - `rdf:type`, `rdf:subject`, `rdf:predicate`, `rdf:object`,
 - `rdf:first`, `rdf:rest`, `rdf:_n`
 - `rdf:value`
- Other:
 - `rdf:nil`

RDF Vocabulary

- Typing using `rdf:type`:
`<A, rdf:type, B>`
“A belongs to class B”
- All properties belong to class `rdf:Property`:
`<P, rdf:type, rdf:Property>`
“P is a property”

`<rdf:type, rdf:type, rdf:Property>`
“rdf:type is a property”

RDF Named Graphs

- Sometimes we need to work with several RDF graphs
- To distinguish them (e.g., in SPARQL), we can give them names
- Often it is convenient to represent triples in named graphs as quadruples

