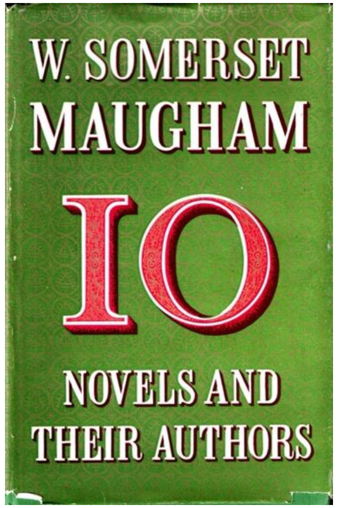
What is RDF?
Part 1: RDF as a Data Model

Amazon Product Catalogue

- Goal: Central catalogue of all products to support search & Q&A systems
- Example Questions:
  - Sizes of Levi’s 511 jeans?
  - Screen sizes of Apple Watch?
  - How many “Loose Jeans” products are sold?
  - How many “Jeans” products are sold?
  - Which products are subject to Textile Labeling Act of Canada?

RDF motivation



Wearables Watches

Smart Watches

Loose Jeans Pants

Clothing

Biographies Novels

Books

screen sizes

38mm 40mm

42 mm

material

Wool Canvas

Cotton

publishers

Penguin Wiley

Scholastic

- Many other properties

RDF motivation

Wearables

Watches

Smart Watches

Electronics

Wearables

Watches

Smart Watches		
…	…	…

Watches		
…	…	…

Wearables		
…	…	…

Electronics		
…	…	…

RDF motivation

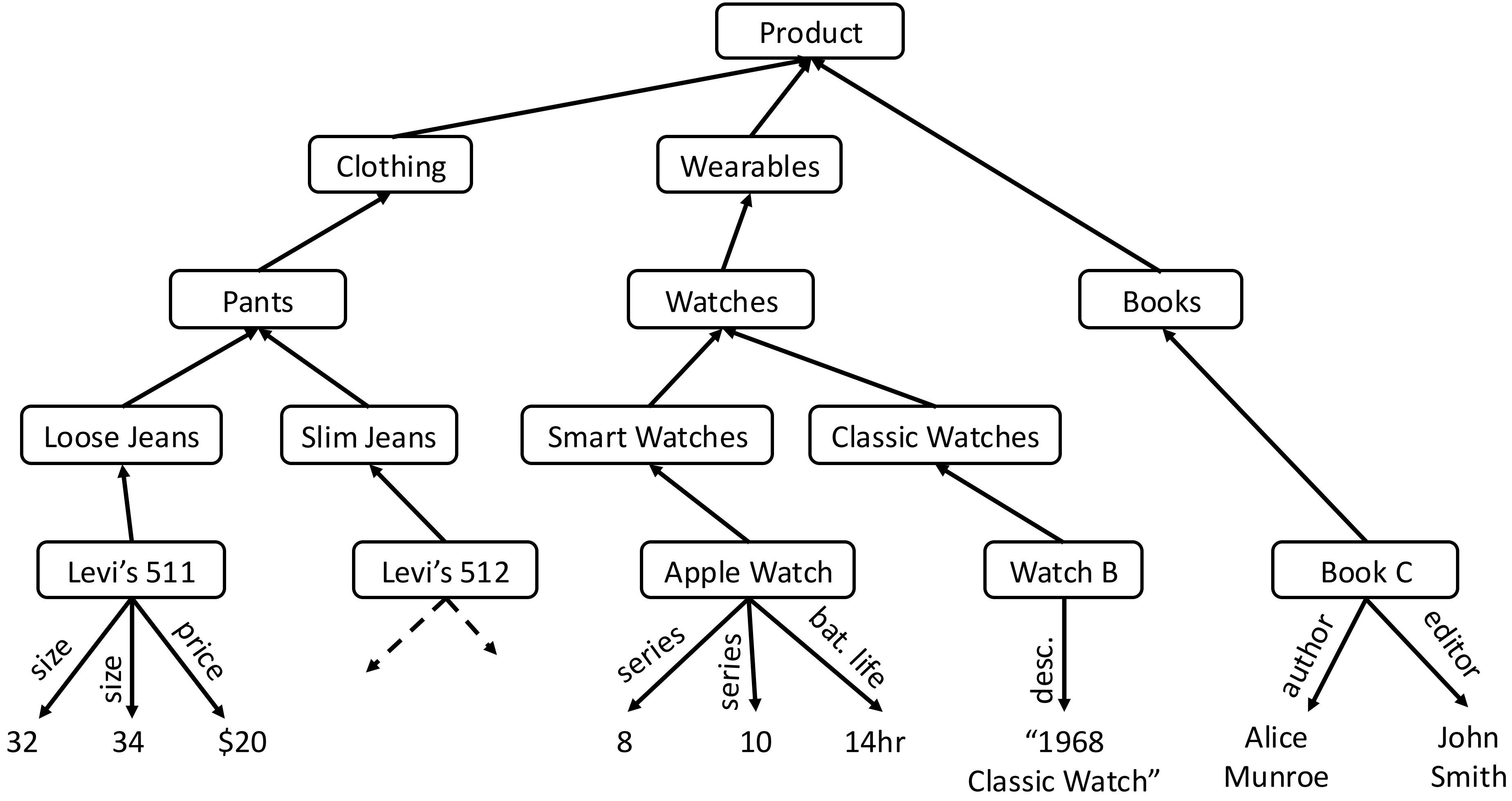
- 4 screen-size
- 10 series
- 3rd party apps
- multiple OS
- …

- 1 screen-size
- 1 series
- waterproof
- …

- Too much irregularity and complexity

Object-oriented/Graph-based data model

- Objects/nodes
- Type hierarchies
- Irregularities in data & connections of objects



RDF component 1: Resources & IRIs

- Resources are objects to represent:
  - Entities
  - Types
  - Properties

gc:Levis511 gc:Levis512 gc:JeansY gc:AppleWatch

gc:LooseJeans gc:Jeans gc:Pants gc:Watches gc:price

LooseJean
name:Levi’s 511
price: $20

- International Resource Identifiers (IRIs): URL-like object identifiers
  - namespace:localID
  - gc: → “http://global-corps.io/rdf-ex#”
  - gc:Levis511 → “http://global-corps.io/rdf-ex#Levis511”

RDF component 2: Literals

- Data values:
  - 20
  - “Foo bar”
  - 01/01/2024

RDF component 3: Triples

- (subject, verb/predicate, object) sentences

(Levi's 511) (is a type of) (LooseJeans)

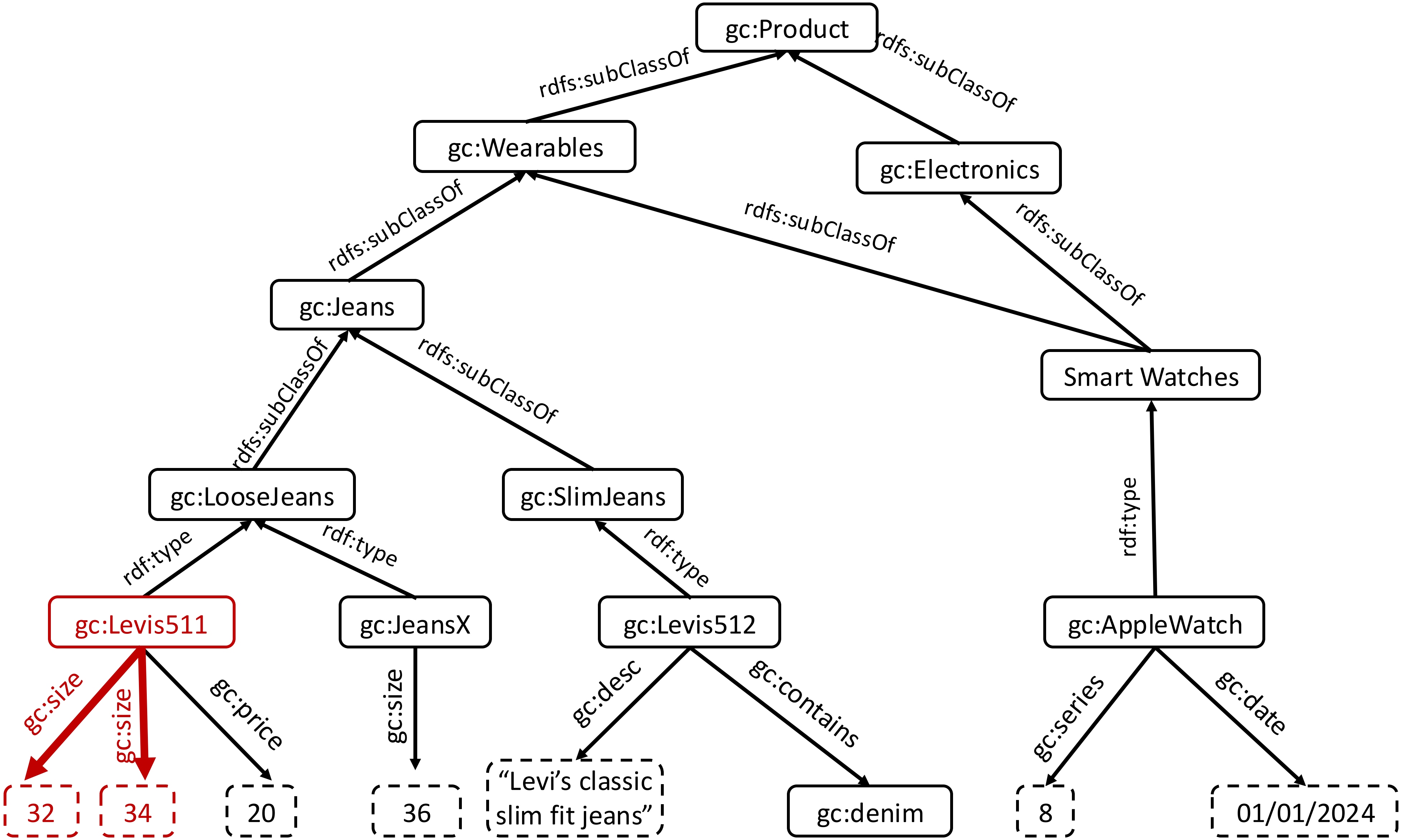
subject verb object

<gc:Levis511, rdf:type, gc:LooseJeans>

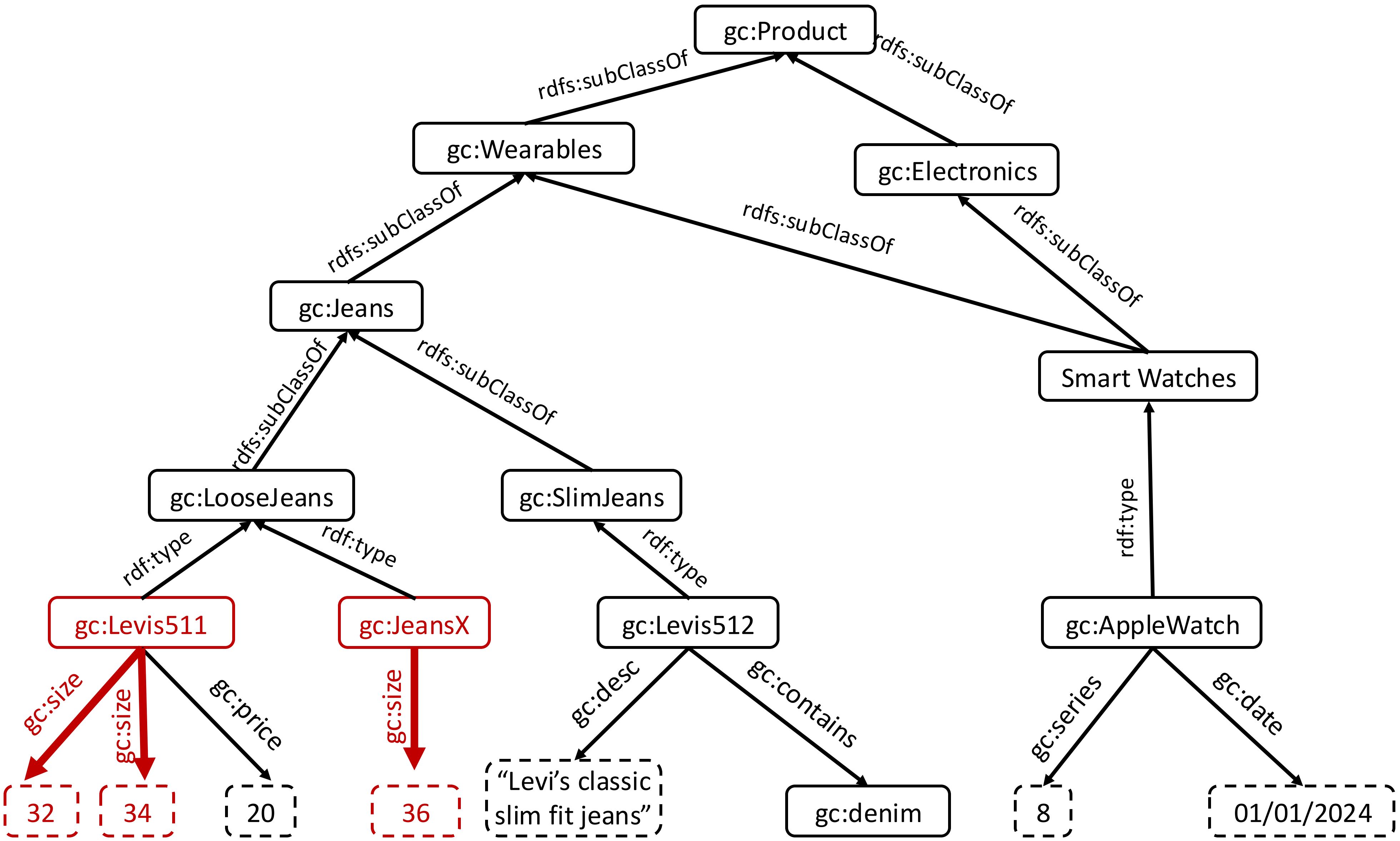
(Levi's 511) (has a price of) (32)

<gc:Levis511, gc:price, 20>

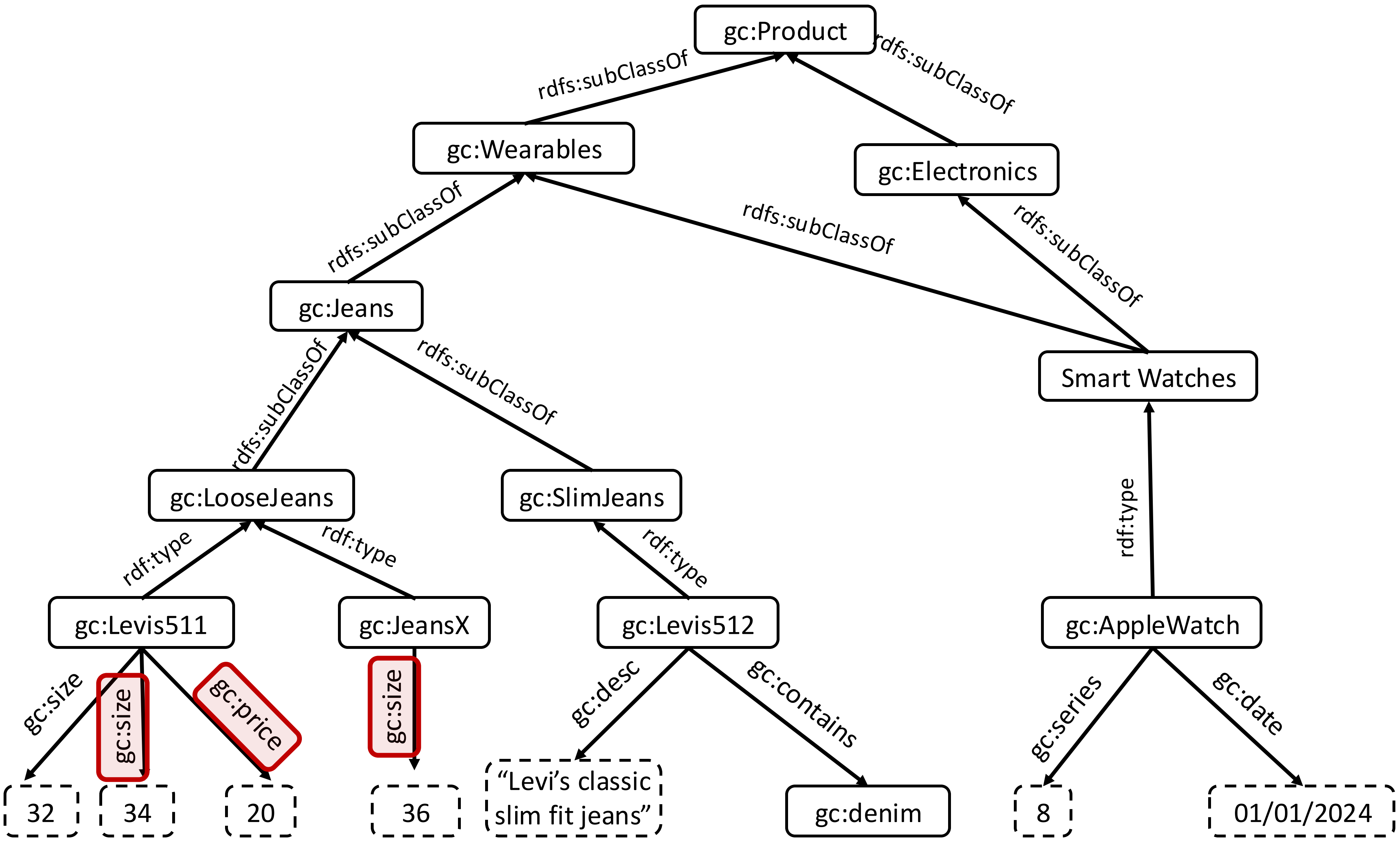
Irregular properties across Resources



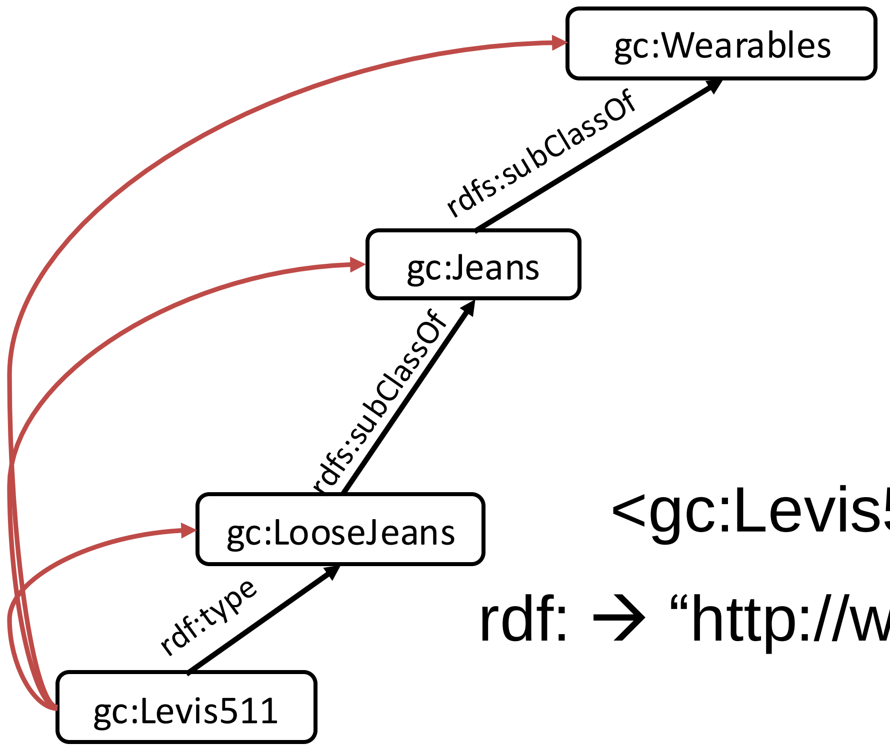
Irregular properties across Resources



Irregular properties across Resources



Type hierarchies



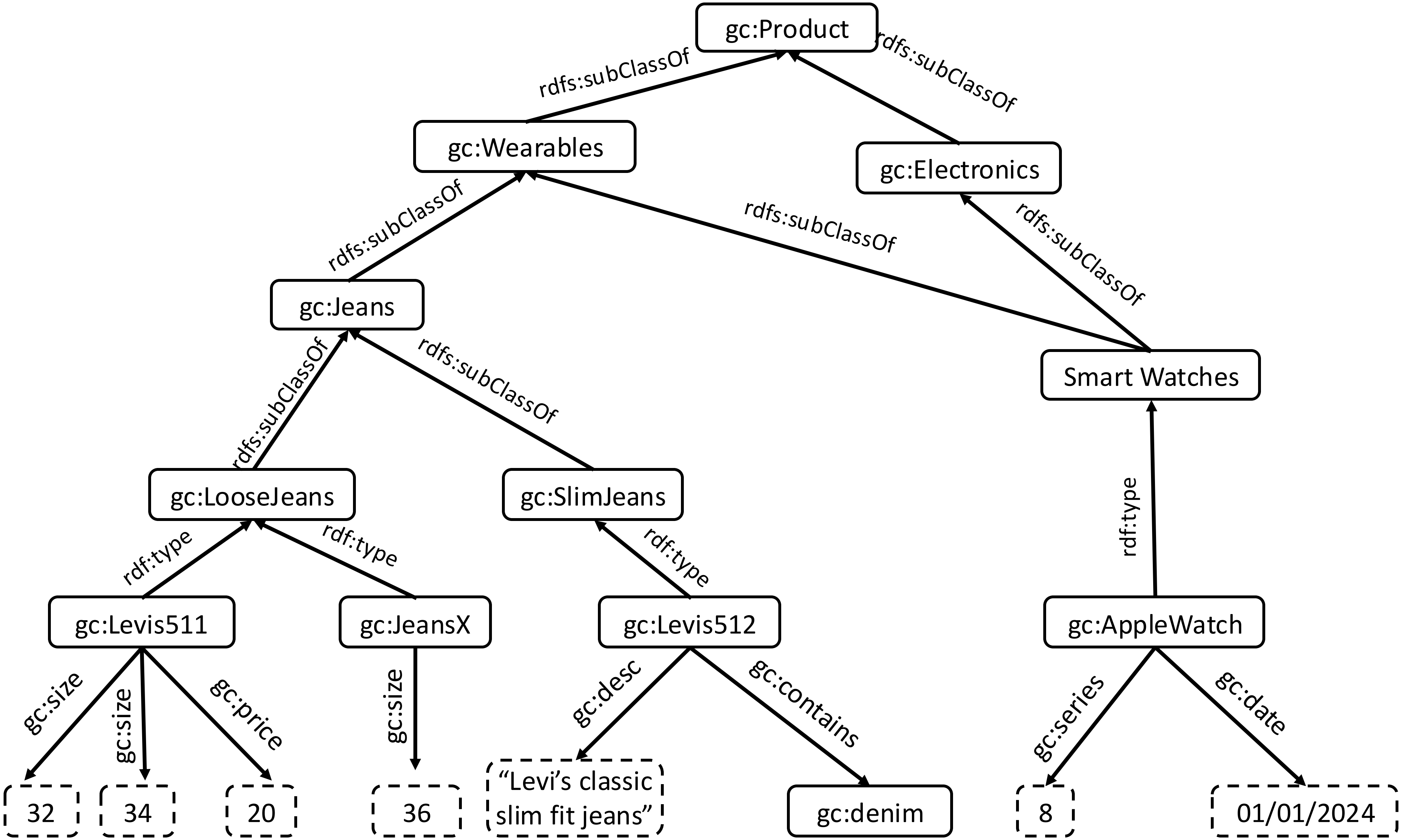
<gc:Levis511, rdf:type, gc:LooseJeans>

rdf: → "http://www.w3.org/1999/02/22-rdf-syntax-ns#"

rdfs:subClassOf

rdfs: → "http://www.w3.org/2000/01/rdf-schema#"

Multiple types/super-classes



Uniform data + schema modeling

LooseJeans		
name	size	price

Uniform data + schema modeling

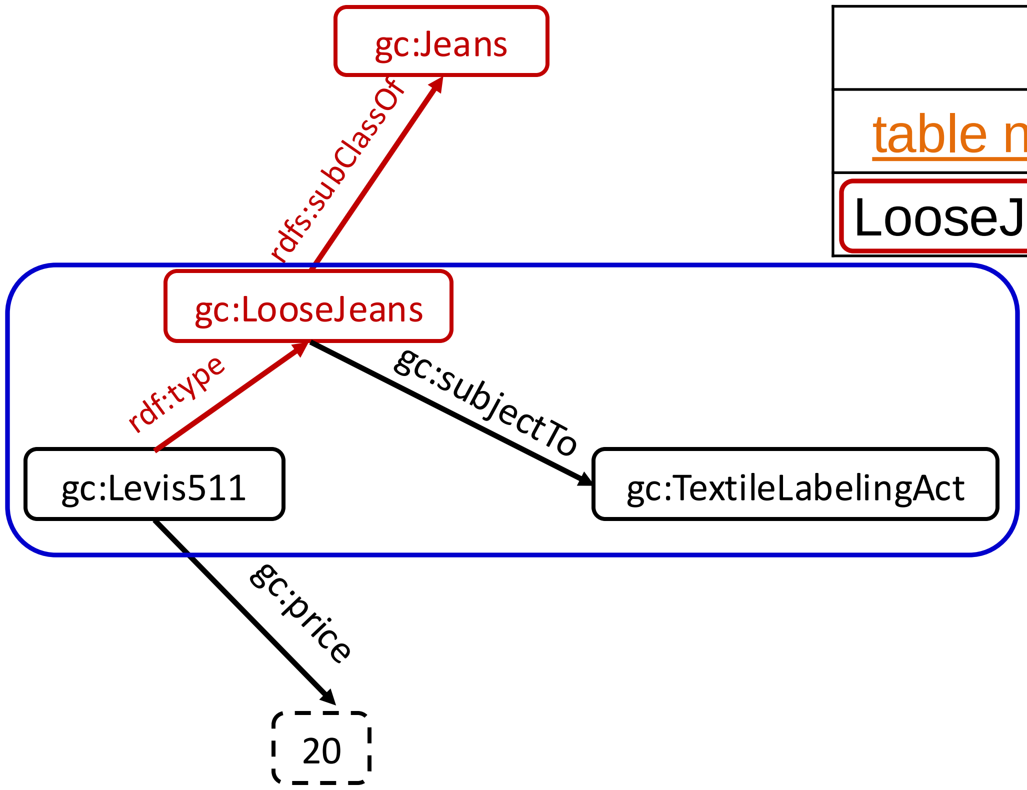
```
CREATE TABLE LooseJeans (
    name VARCHAR,
    size INT,
    price INT)

INSERT INTO LooseJeans (...)
VALUES (“Levi’s 511, 32, 20)
```

LooseJeans		
name	size	price
Levi’s 511	32	20
…	…	…

Uniform data + schema modeling

LooseJeans		
name	size	price
Levi's 511	32	20
…	…	…

Regulations	
table name	reg. name
LooseJeans	Textile Labeling Act

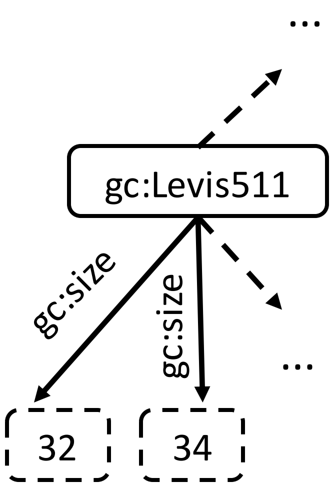


Querying RDF databases

Which sizes of Levi's 511 jeans are sold?

```
SELECT ?x
WHERE {
    gc:Levis-511 gc:size ?x .
}
```

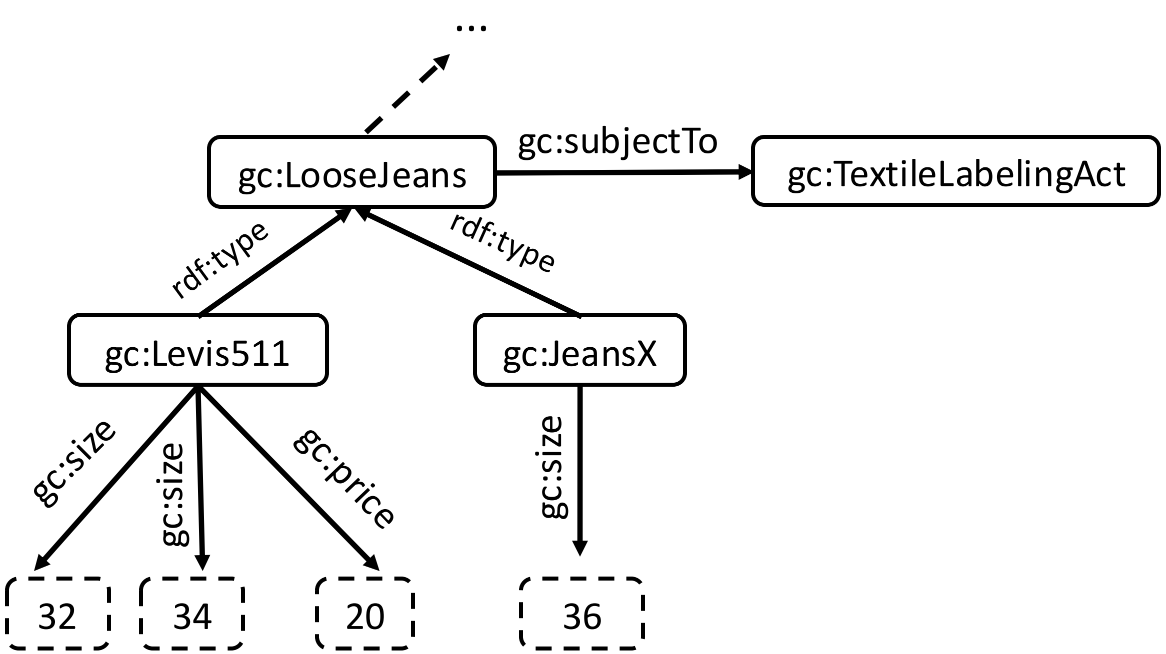
?x
32
34



Example data+schema querying

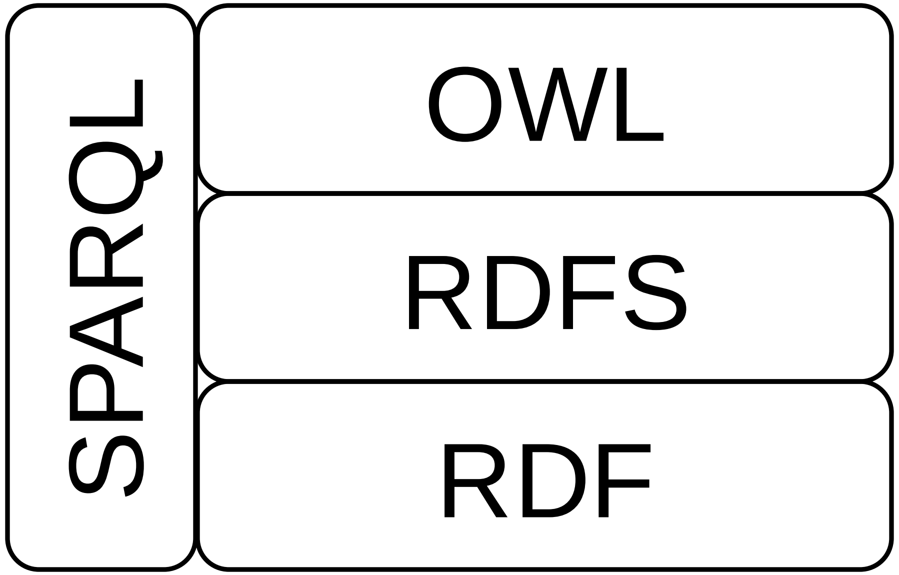
Which products are subject to Textile Labeling Act?

```
SELECT ?p
WHERE {
    ?p rdf:type ?c .
    ?c gc:subjectTo gc:TextileLabelingAct .
}
```



?p
gc:Levis511
gc:JeansX

RDF standards



RDF standards

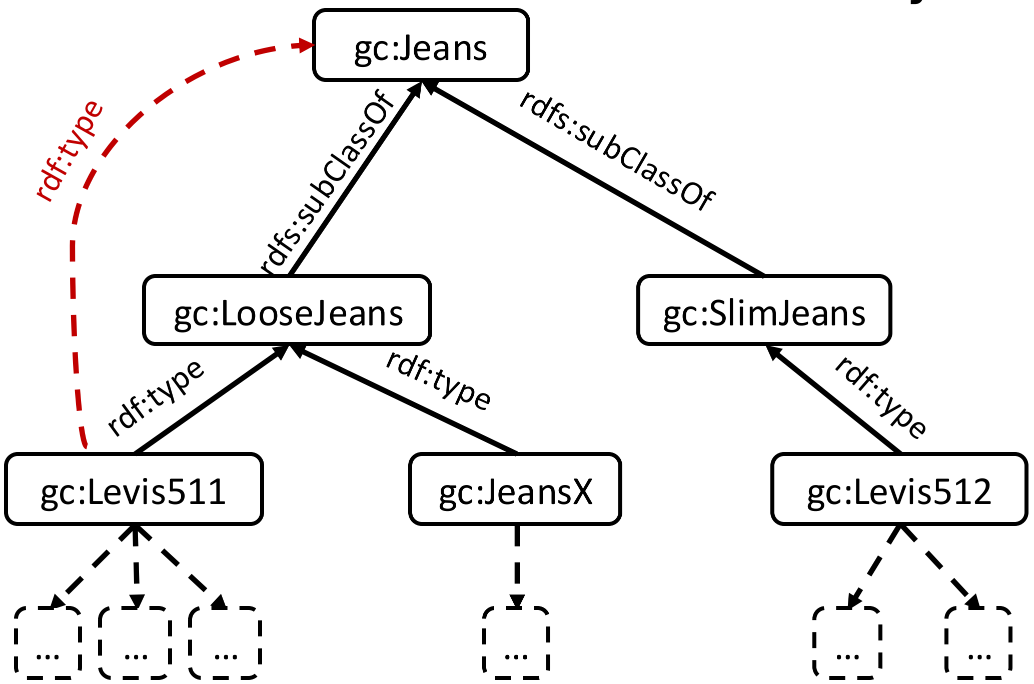
- Standardized vocabulary, i.e. IRIs. If a DBMS implements these standards:
  - Facilitates basic reasoning & infer new triples in a database
- rdf:type → http://www.w3.org/1999/02/22-rdf-syntax-ns#type
  - Types of resources
- rdfs:subClassOf: http://www.w3.org/2000/01/rdf-schema#subClassOf
  - Sub/super-class relationships

RDF standards

Inference Rule

(x rdf:type A) & (A rdfs:subClassOf B) => (x rdf:type B)

```
SELECT ?p
WHERE {
    ?p rdf:type gc:Jeans .
}
```



?p
gc:Levis511
gc:JeansX
gc:Levis512

OWL inference example

- owl:sameAs → http://www.w3.org/2002/07/owl#sameAs

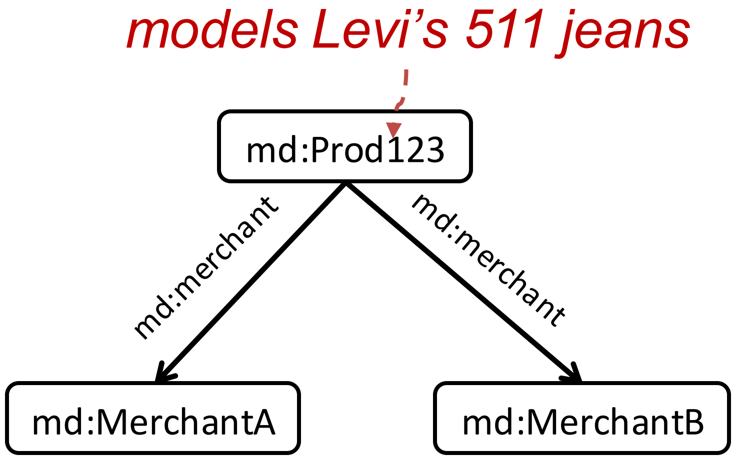
  <md:Prod123, md:merchant, md:MerchantA>

  <md:Prod123, md:merchant, md:MerchantB>

  <md:MerchantA, md:locatedIn, md:Waterloo>

  …

- Goal: Integrate merchant db with product catalogue db



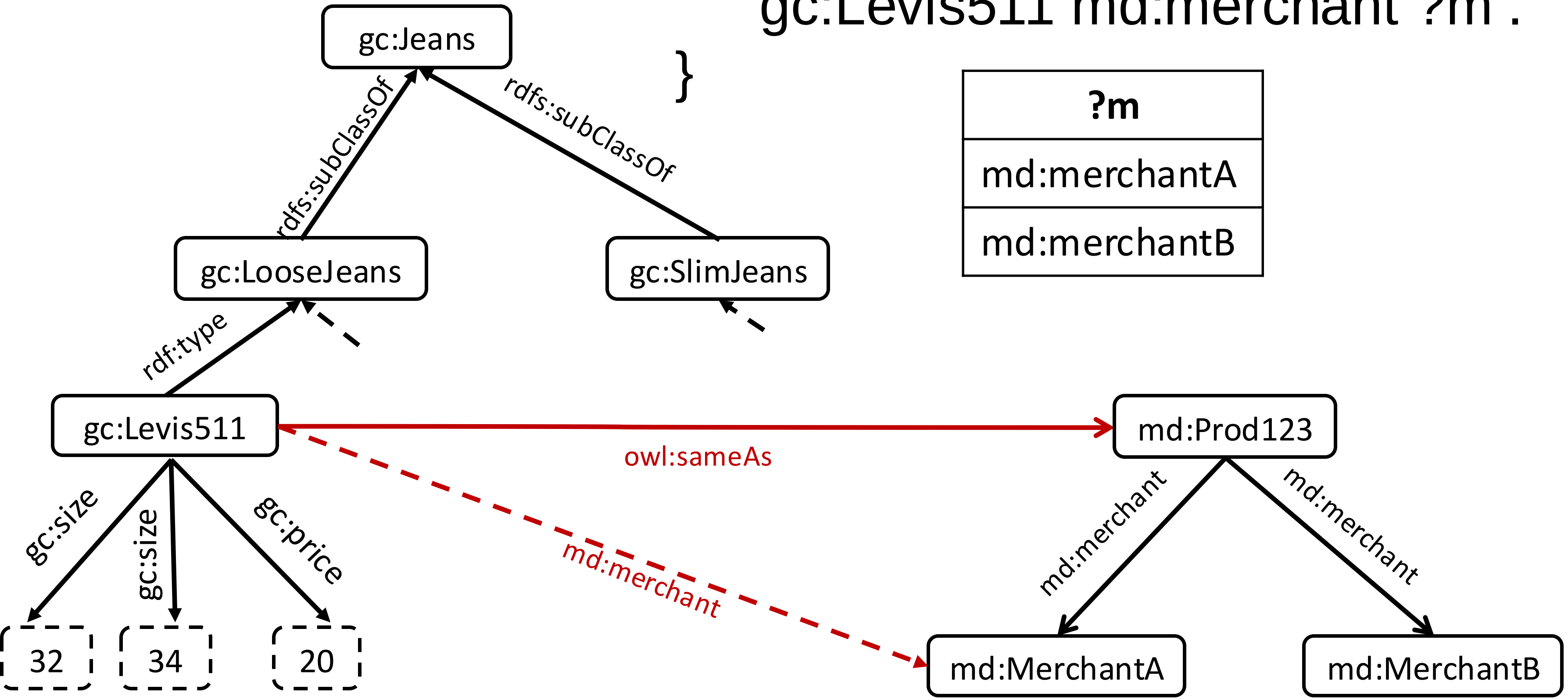
OWL inference example

Inference Rule

(x owl:sameAs y) & (y p o) => (x p o)

```
SELECT ?m
WHERE {
      gc:Levis511 md:merchant ?m .
}
```

?m
md:merchantA
md:merchantB



Custom transitive predicates: owl:TransitiveProperty

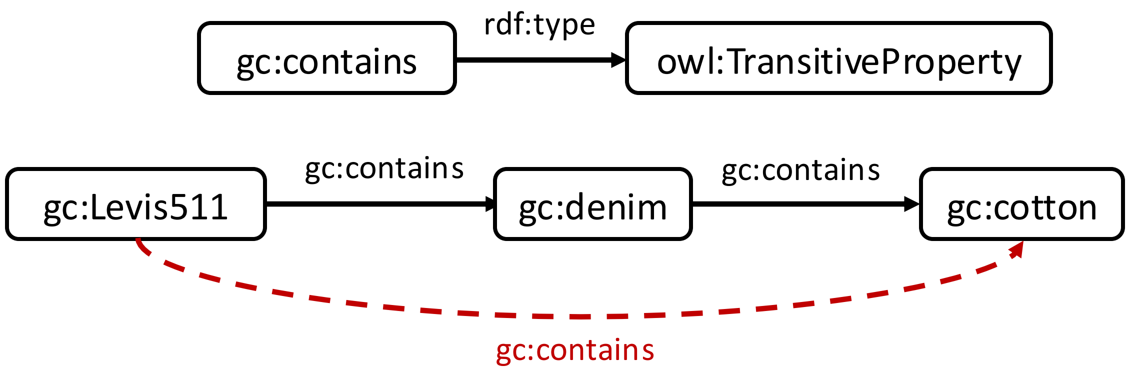
Inference Rule

(gc:contains rdf:type owl:transitiveProperty) & (x gc:contains A) & (A gc:contains B)

=> (x gc:contains B)

```
SELECT ?m
WHERE {
    gc:Levis511 gc:contains ?m .
}
```

?m
gc:denim
gc:cotton

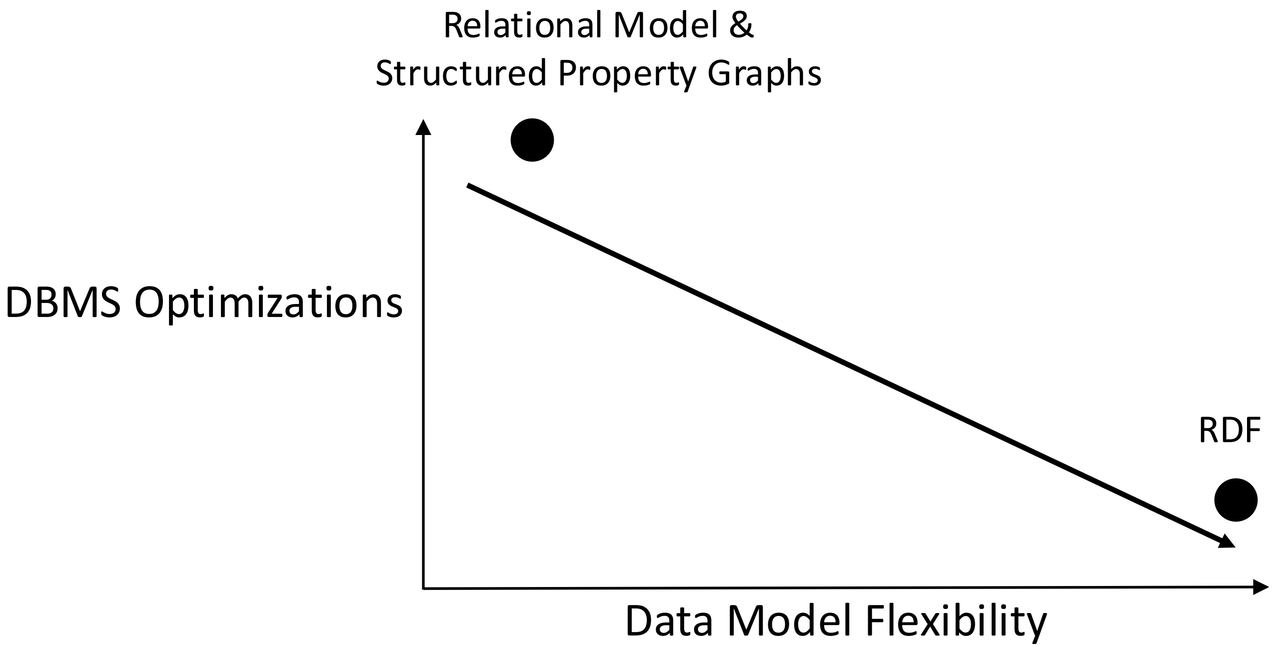


RDF cons

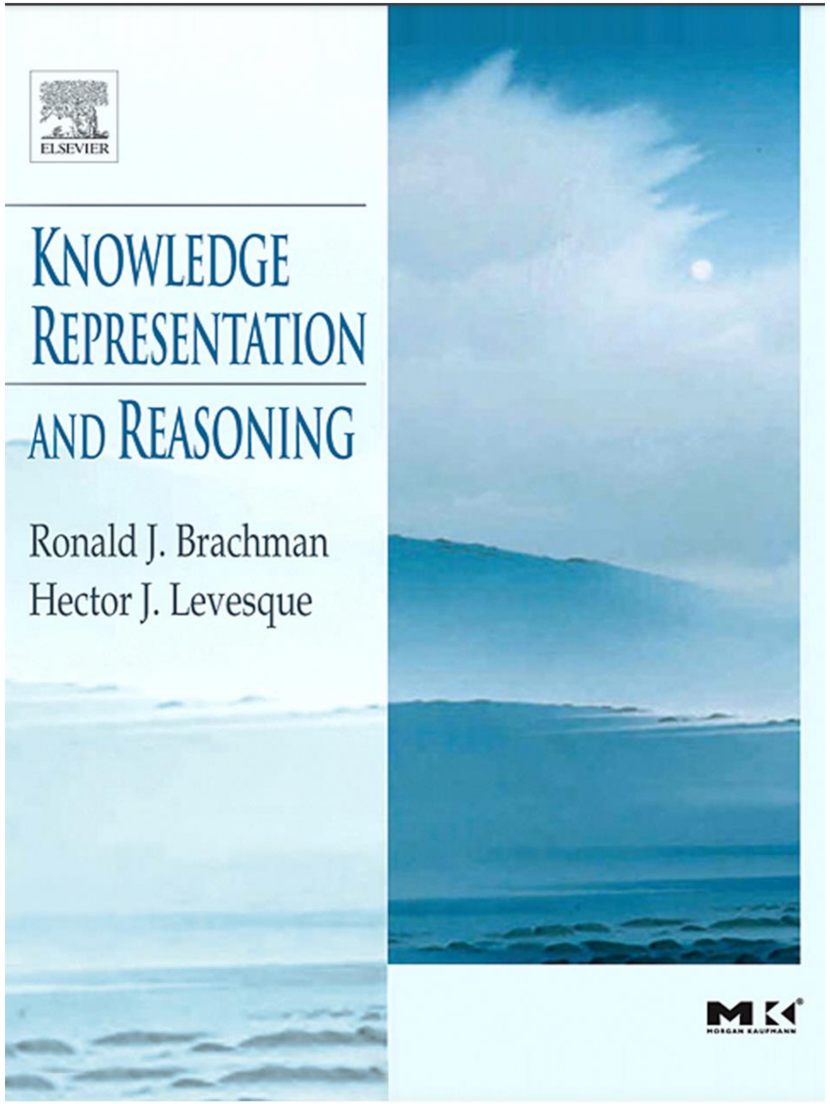
- Verbosity
  - Long IRIs
  - Repeated values
- Scalability & Performance
  - General principle:

<gc:Levis511, gc:size, 32>
<gc:Levis511, gc:size, 34>
<gc:Levis512, gc:size, 36>

product	size
gc:Levis511	32
gc:Levis511	34
gc:Levis512	36

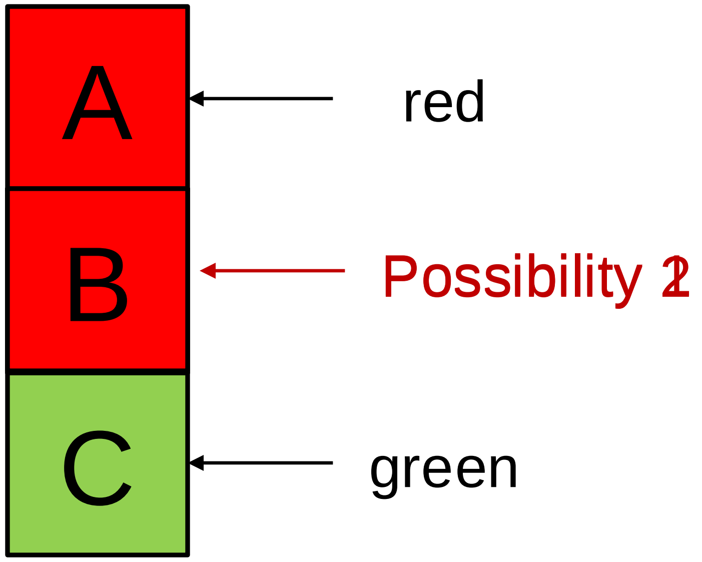


RDF & KRR in modern era of AI



RDF & KRR in modern era of AI

How many red boxes are above green boxes?




```
SELECT b1
WHERE {
    ?b1 abc:color abc:red .
    ?b1 abc:ontopOf ?b2 .
    ?b2 abc:color abc:green .
}
```

<abc:boxA, rdf:type, abc:box>

<abc:boxB, rdf:type, abc:box>

<abc:boxA, abc:onTopOf, abc:boxB>

…

<abc:boxA, abc:color, abc:red>

<abc:boxC, abc:color, abc:green>

- Restrictions: owl:someValuesFrom, owl:oneOf, owl:cardinality

Cyc

CYC

Getting from Generative AI to Trustworthy AI:
What LLMs might learn from Cyc

Doug Lenat
doug@cyc.com

Gary Marcus
gary.marcus@nyu.edu

July 31, 2023

Abstract

Generative AI, the most popular current approach to AI, consists of large language models (LLMs) that are trained to produce outputs that are *plausible*, but not necessarily *correct*. Although their abilities are often uncanny, they are lacking in aspects of reasoning, leading LLMs to be less than completely trustworthy. Furthermore, their results tend to be both unpredictable and uninterpretable.

We lay out 16 desiderata for future AI, and discuss an alternative approach to AI which could theoretically address many of the limitations associated with current approaches: AI educated with curated pieces of explicit knowledge and rules of thumb, enabling an inference engine to automatically deduce the logical entailments of all that knowledge. Even long arguments produced this way can be both trustworthy and interpretable, since the full step-by-step line of reasoning is always available, and for each step the provenance of the knowledge used can be documented and audited. There is however a catch: if the logical language is expressive enough to fully represent the meaning of anything we can say in English, then the inference engine runs much too slowly. That's why symbolic AI systems typically settle for some fast but much less expressive logic, such as knowledge graphs. We describe how one AI system, Cyc, has developed ways to overcome that tradeoff and is able to reason in higher order logic in real time.

We suggest that any trustworthy general AI will need to hybridize the approaches, the LLM approach and more formal approach, and lay out a path to realizing that dream.