

CSC 481: Resource Description Framework (RDF)

Based on Chapter 3 of “Semantic Web for the Working Ontologist - Modeling in RDF, RDFS and OWL” by Allenmag and Handler

Rodrigo Canaan
Assistant Professor
Computer Science Department
Cal Poly, San Luis Obispo
rcanaan@calpoly.edu

Note: book is available at Cal Poly Library

Note: book is available at Cal Poly Library

- https://csu-calpoly.primo.exlibrisgroup.com/permalink/01CALS_PSU/1qh1nk7/alma991067617147802901

Note: book is available at Cal Poly Library

- https://csu-calpoly.primo.exlibrisgroup.com/permalink/01CALS_PSU/1qh1nk7/alma991067617147802901

Note: book is available at Cal Poly Library

- https://csu-calpoly.primo.exlibrisgroup.com/permalink/01CALS_PSU/1qh1nk7/alma991067617147802901
- <https://ebookcentral.proquest.com/lib/calpoly/detail.action?docID=689813&pq-origsite=primo#>

Main considerations for semantic web

Main considerations for semantic web

- The AAA slogan (**A**nyone can say **A**nything about **A**ny topic)

Main considerations for semantic web

- The AAA slogan (**A**nyone can say **A**nything about **A**ny topic)
- Open world assumption

Main considerations for semantic web

- The AAA slogan (**A**nyone can say **A**nything about **A**ny topic)
- Open world assumption
- Non-unique naming

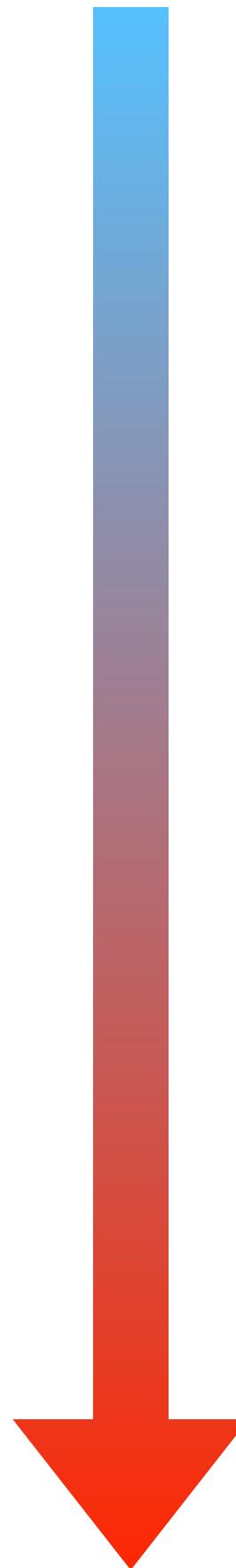
Main considerations for semantic web

- The AAA slogan (**A**nyone can say **A**nything about **A**ny topic)
- Open world assumption
- Non-unique naming
- Network effect

Semantic Web Standards

By level of expressivity

Less expressive



- **RDF - Resource Description Framework.** Models data as triplets (subject, predicate, object)
- **RDFS - RDF Schema language.** Enhances RDF with notions of classes, subclasses and properties.
- **OWL - Web Ontology Language.** adds more vocabulary for describing properties and classes (e.g. disjointness, cardinality, symmetry of properties...)
 - Related to Description Logics (ch. 9 of the Brachman & Levesque book)

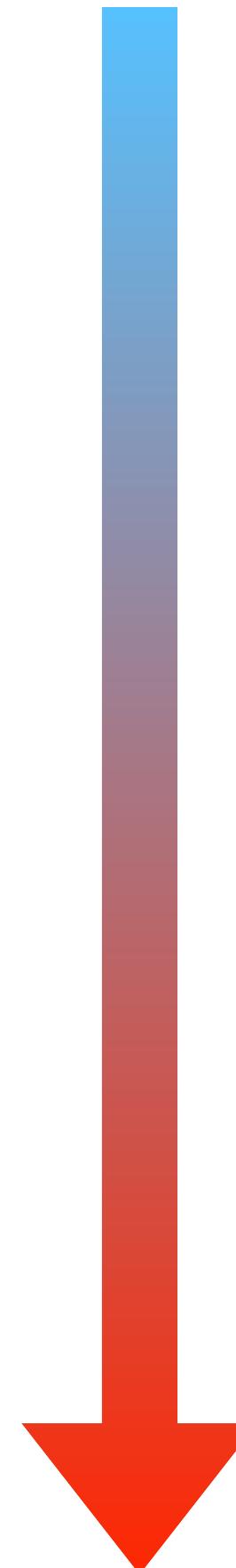
More expressive

Semantic Web Standards

By level of expressivity

Less expressive

Our focus today



- **RDF - Resource Description Framework.** Models data as triplets (subject, predicate, object)
- **RDFS - RDF Schema language.** Enhances RDF with notions of classes, subclasses and properties.
- **OWL - Web Ontology Language.** adds more vocabulary for describing properties and classes (e.g. disjointness, cardinality, symmetry of properties...)
 - Related to Description Logics (ch. 9 of the Brachman & Levesque book)

More expressive

What is RDF?

- RDF is a **Data Model** - a way of thinking, organizing and exchanging data
- In other words, it is an abstraction, not a specific language/syntax
- Knowledge represented as a directed graph composed of triples
- Triples defined by:
 - A node for the subject
 - A node for the predicate
 - A node for the object
- **RDFS - RDF Schema language.** Enhances RDF with notions of classes, subclasses and properties.
- **OWL - Web Ontology Language.** adds more vocabulary for describing properties and classes (e.g. disjointness, cardinality, symmetry of properties...)
 - Related to Description Logics (ch. 9 of the Brachman & Levesque book)

From tabular data to triples

Table 3.1 Tabular Data about Elizabethan Literature and Music

ID	Title	Author	Medium	Year
1	<i>As You Like It</i>	Shakespeare	Play	1599
2	<i>Hamlet</i>	Shakespeare	Play	1604
3	<i>Othello</i>	Shakespeare	Play	1603
4	“Sonnet 78”	Shakespeare	Poem	1609
5	<i>Astrophil and Stella</i>	Sir Phillip Sidney	Poem	1590
6	<i>Edward II</i>	Christopher Marlowe	Play	1592
7	<i>Hero and Leander</i>	Christopher Marlowe	Poem	1593
8	<i>Greensleeves</i>	Henry VIII Rex	Song	1525

From tabular data to triples

Tabular Data about Elizabethan Literature and Music			
Title	Author	Medium	Year
<i>As You Like It</i>	Shakespeare	Play	1599
<i>Hamlet</i>	Shakespeare	Play	1604
<i>Othello</i>	Shakespeare	Play	1603
"Sonnet 78"	Shakespeare	Poem	1609
<i>Astrophil and Stella</i>	Sir Phillip Sidney	Poem	1590
<i>Edward II</i>	Christopher Marlowe	Play	1592
<i>Hero and Leander</i>	Christopher Marlowe	Poem	1593
<i>Greensleeves</i>	Henry VIII Rex	Song	1525

From tabular data to triples

Tabular Data about Elizabethan Literature and Music			
Title	Author	Medium	Year
<i>As You Like It</i>	Shakespeare	Play	1599
<i>Hamlet</i>	Shakespeare	Play	1604
<i>Othello</i>	Shakespeare	Play	1603
"Sonnet 78"	Shakespeare	Poem	1609
<i>Astrophil and Stella</i>	Sir Phillip Sidney	Poem	1590
<i>Edward II</i>	Christopher Marlowe	Play	1592
<i>Hero and Leander</i>	Christopher Marlowe	Poem	1593
<i>Greensleeves</i>	Henry VIII Rex	Song	1525

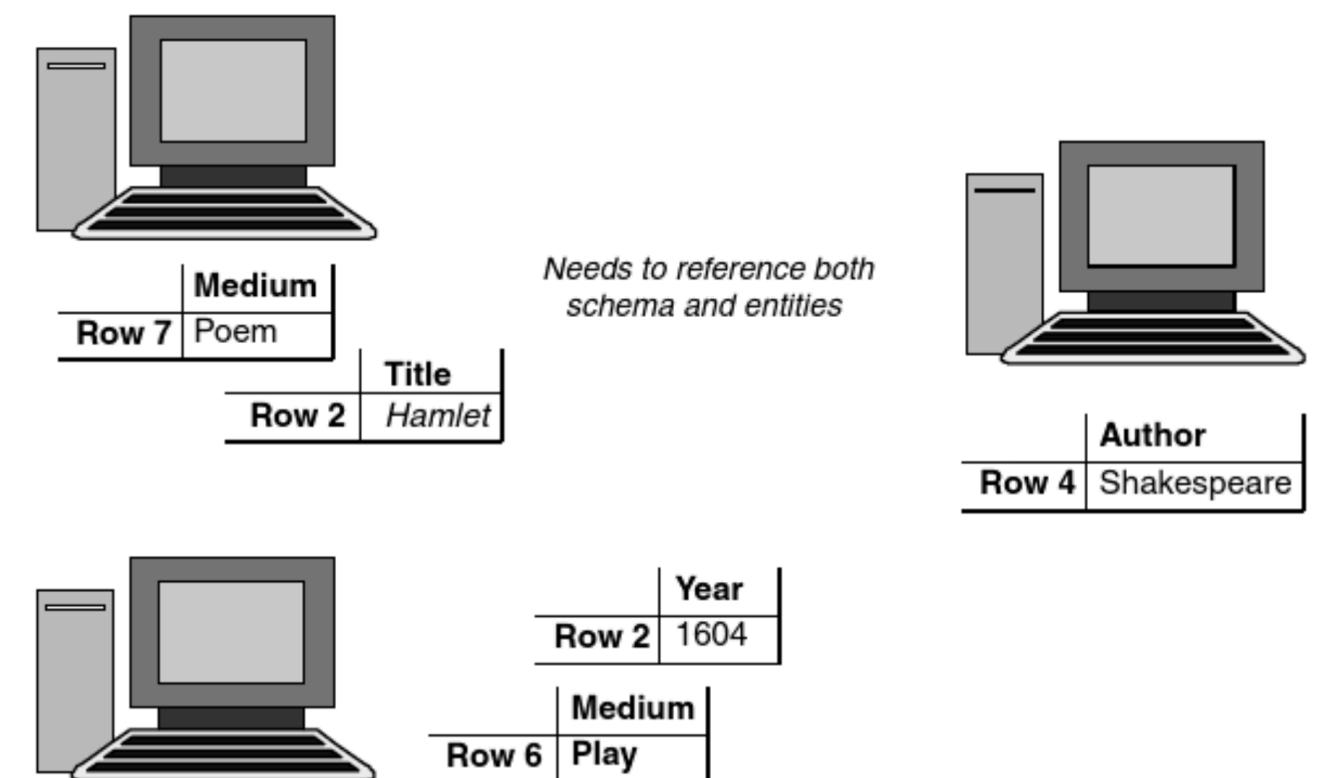


FIGURE 3.3

Distributing data across the Web, cell by cell.

From tabular data to triples

Table 3.1 Tabular Data about Elizabethan Literature and Music

ID	Title	Author	Medium	Year
1	<i>As You Like It</i>	Shakespeare	Play	1599
2	<i>Hamlet</i>	Shakespeare	Play	1604
3	<i>Othello</i>	Shakespeare	Play	1603
4	“Sonnet 78”	Shakespeare	Poem	1609
5	<i>Astrophil and Stella</i>	Sir Phillip Sidney	Poem	1590
6	<i>Edward II</i>	Christopher Marlowe	Play	1592
7	<i>Hero and Leander</i>	Christopher Marlowe	Poem	1593
8	<i>Greensleeves</i>	Henry VIII Rex	Song	1525

From tabular data to triples

Tabular Data about Elizabethan Literature and Music			
Title	Author	Medium	Year
<i>As You Like It</i>	Shakespeare	Play	1599
<i>Hamlet</i>	Shakespeare	Play	1604
<i>Othello</i>	Shakespeare	Play	1603
"Sonnet 78"	Shakespeare	Poem	1609
<i>Astrophil and Stella</i>	Sir Phillip Sidney	Poem	1590
<i>Edward II</i>	Christopher Marlowe	Play	1592
<i>Hero and Leander</i>	Christopher Marlowe	Poem	1593
<i>Greensleeves</i>	Henry VIII Rex	Song	1525

From tabular data to triples

Tabular Data about Elizabethan Literature and Music			
Title	Author	Medium	Year
<i>As You Like It</i>	Shakespeare	Play	1599
<i>Hamlet</i>	Shakespeare	Play	1604
<i>Othello</i>	Shakespeare	Play	1603
"Sonnet 78"	Shakespeare	Poem	1609
<i>Astrophil and Stella</i>	Sir Phillip Sidney	Poem	1590
<i>Edward II</i>	Christopher Marlowe	Play	1592
<i>Hero and Leander</i>	Christopher Marlowe	Poem	1593
<i>Greensleeves</i>	Henry VIII Rex	Song	1525

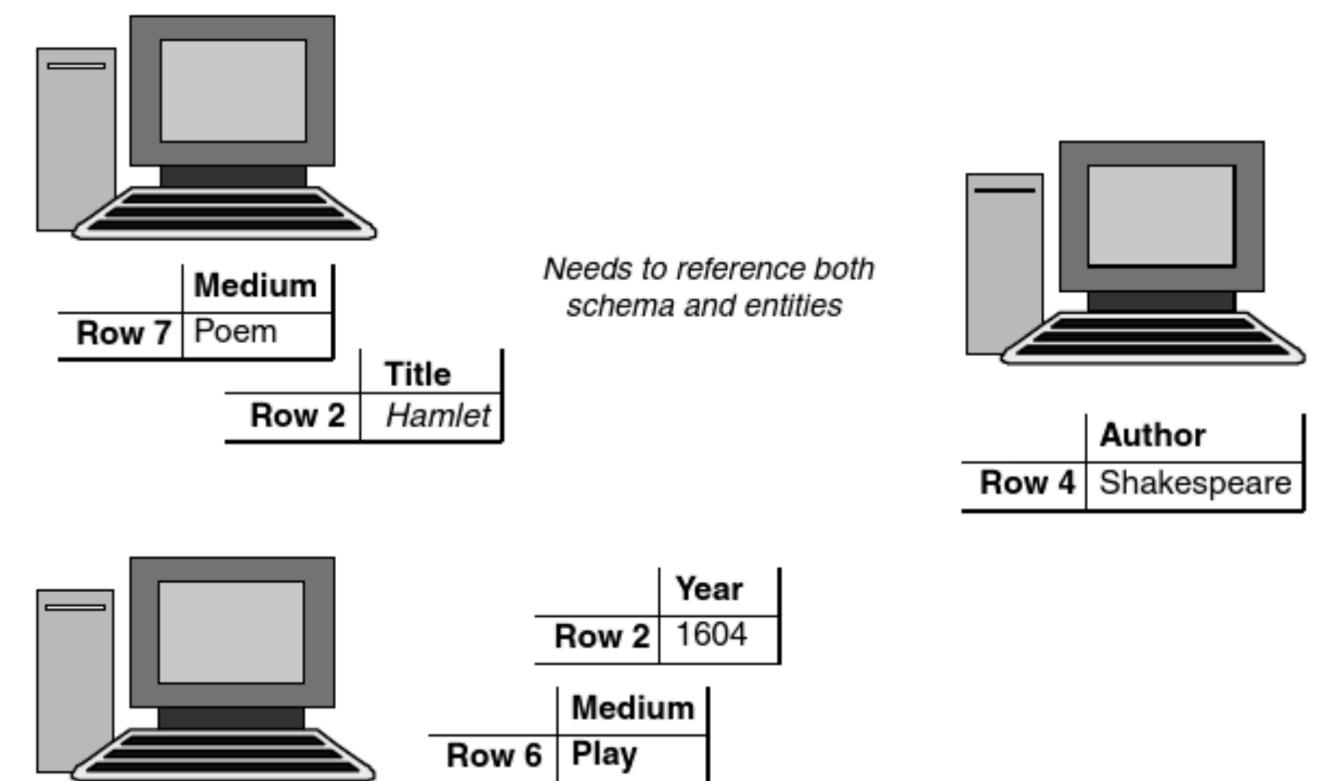
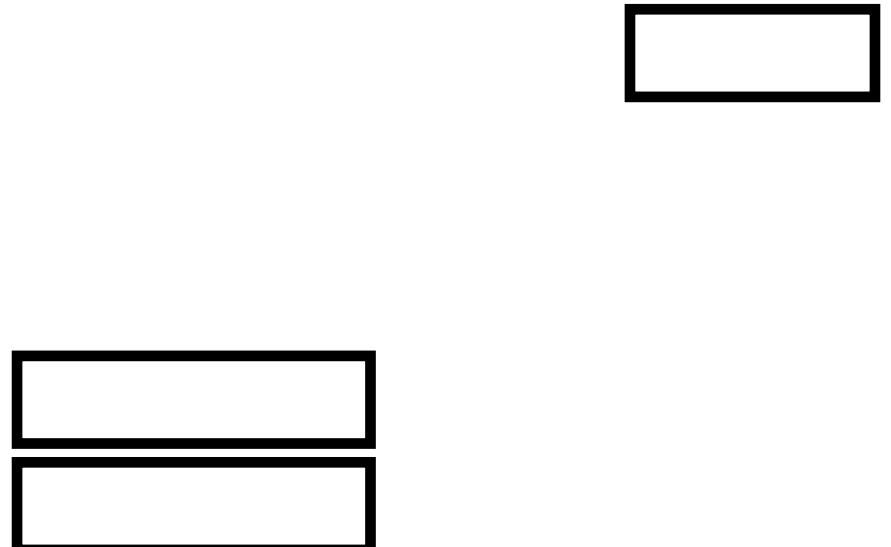


FIGURE 3.3

Distributing data across the Web, cell by cell.

Triples as building blocks of RDF

Table 3.3 Sample Triples

Subject	Predicate	Object
Shakespeare	wrote	King Lear
Shakespeare	wrote	Macbeth
Anne Hathaway	married	Shakespeare
Shakespeare	livedIn	Stratford
Stratford	isIn	England
Macbeth	setIn	Scotland
England	partOf	UK
Scotland	partOf	UK

Triples as building blocks of RDF

Table 3.3 Sample Triples

Subject	Predicate	Object
Shakespeare	wrote	King Lear
Shakespeare	wrote	Macbeth
Anne Hathaway	married	Shakespeare
Shakespeare	livedIn	Stratford
Stratford	isIn	England
Macbeth	setIn	Scotland
England	partOf	UK
Scotland	partOf	UK

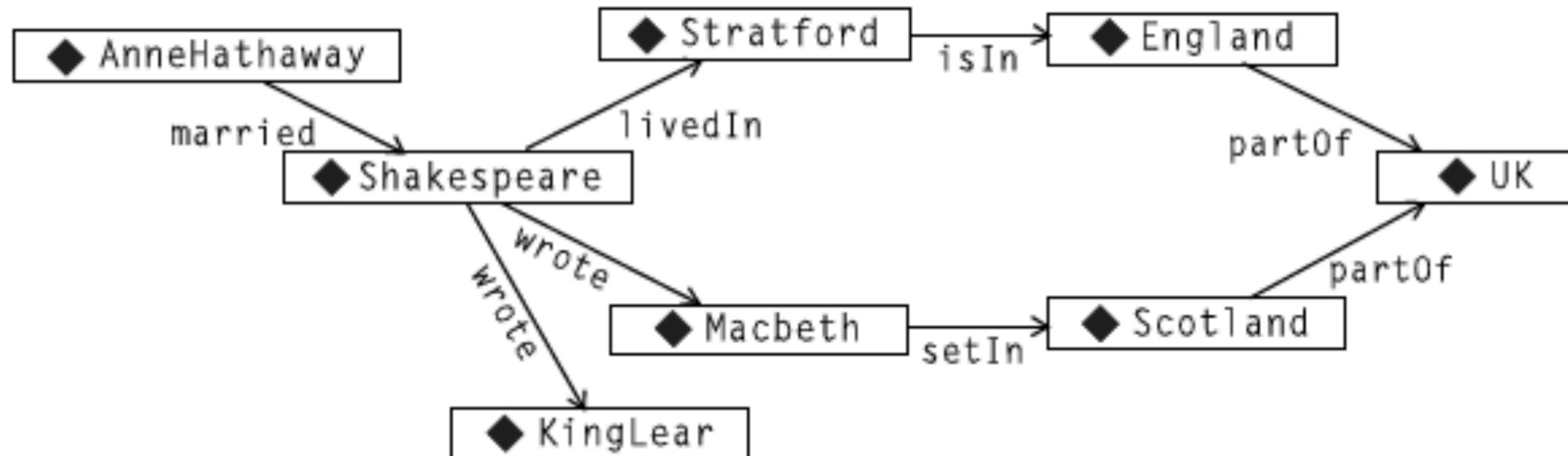


FIGURE 3.4

Graph display of triples from Table 3.3. Eight triples appear as eight labeled edges.

Merging data from multiple sources

Table 3.4 Triples about Shakespeare's Plays

Subject	Predicate	Object
Shakespeare	Wrote	<i>As You Like It</i>
Shakespeare	Wrote	<i>Henry V</i>
Shakespeare	Wrote	<i>Love's Labour's Lost</i>
Shakespeare	Wrote	<i>Measure for Measure</i>
Shakespeare	Wrote	<i>Twelfth Night</i>
Shakespeare	Wrote	<i>The Winter's Tale</i>
Shakespeare	Wrote	<i>Hamlet</i>
Shakespeare	Wrote	<i>Othello</i>
Shakespeare	Wrote	etc.

(a)

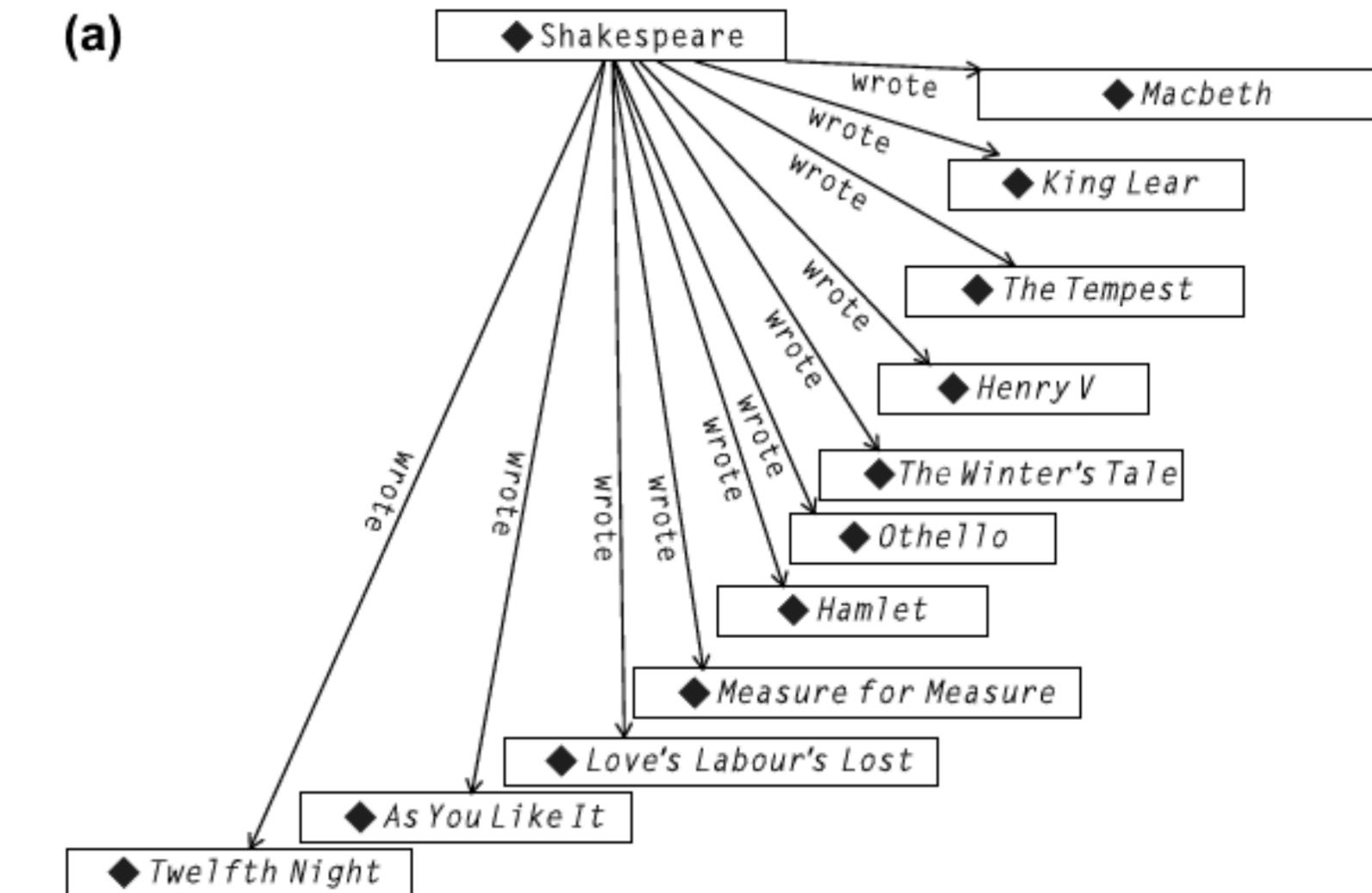
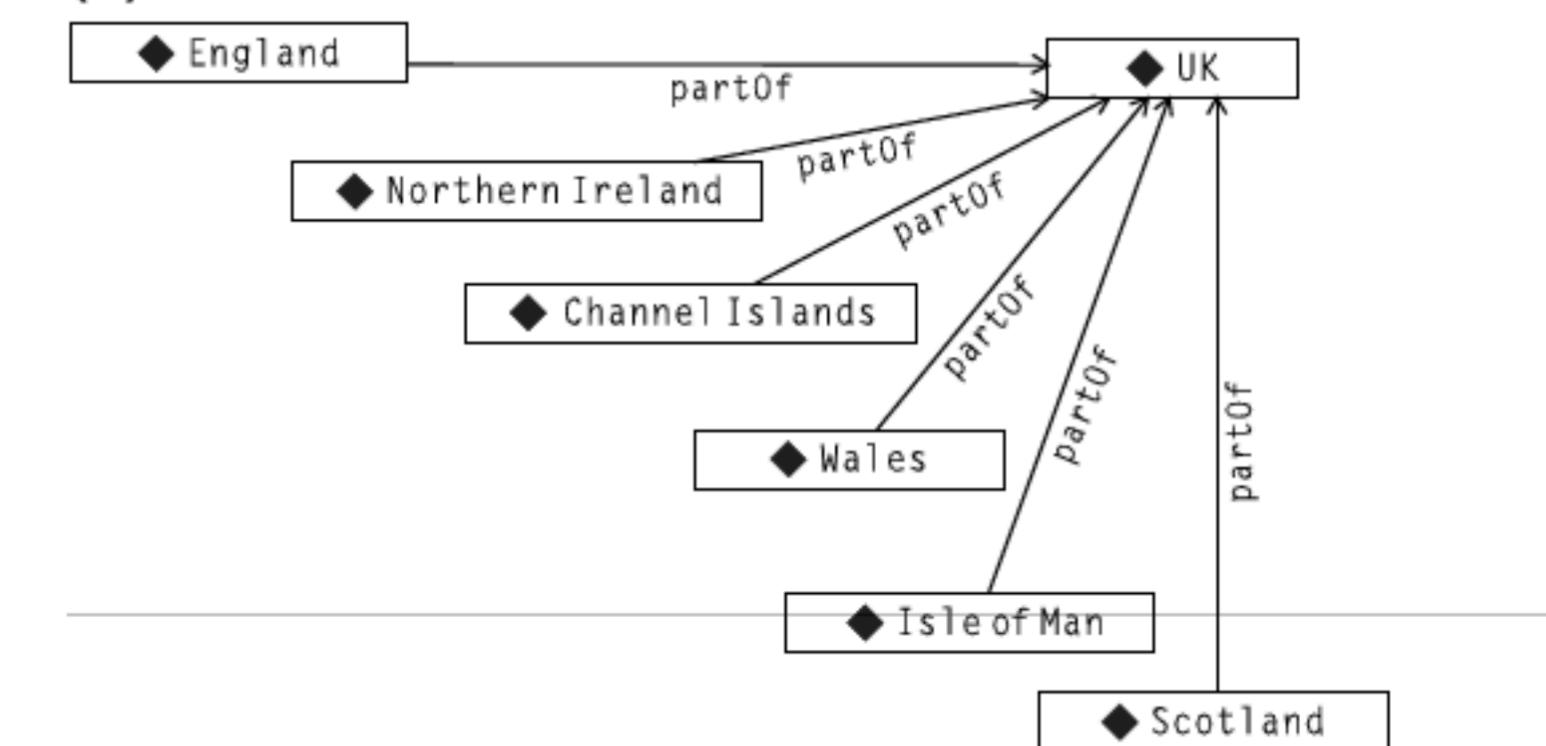


Table 3.5 Triples about the Parts of the United Kingdom

Subject	Predicate	Object
Scotland	part Of	The UK
England	part Of	The UK
Wales	part Of	The UK
Northern Ireland	part Of	The UK
Channel Islands	part Of	The UK
Isle of Man	part Of	The UK

(b)



Merging data from multiple sources

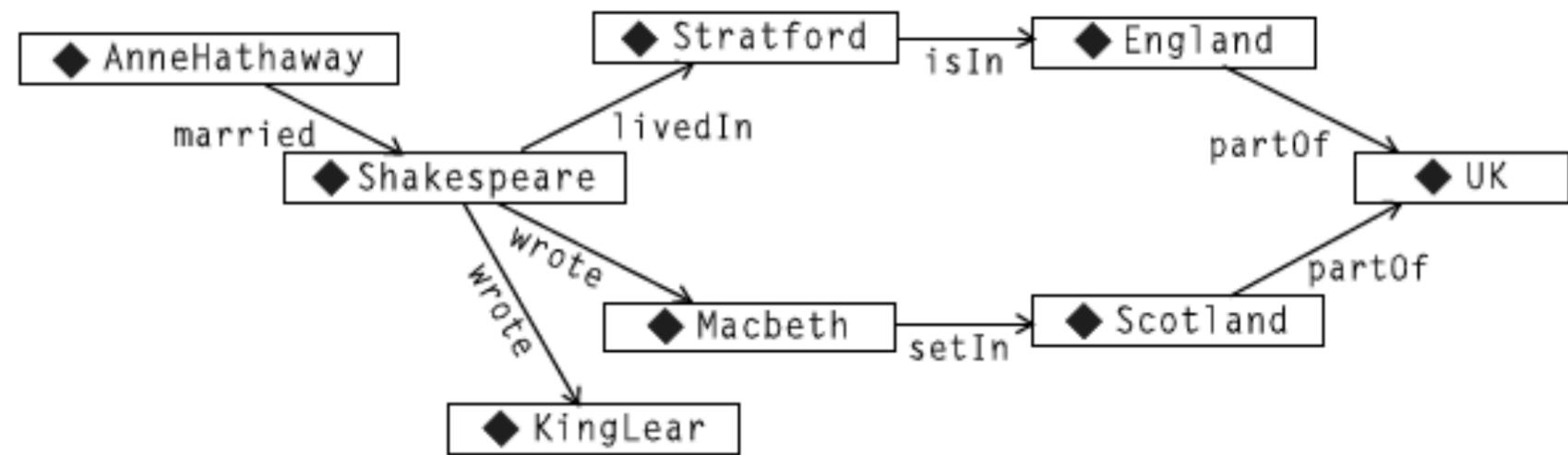
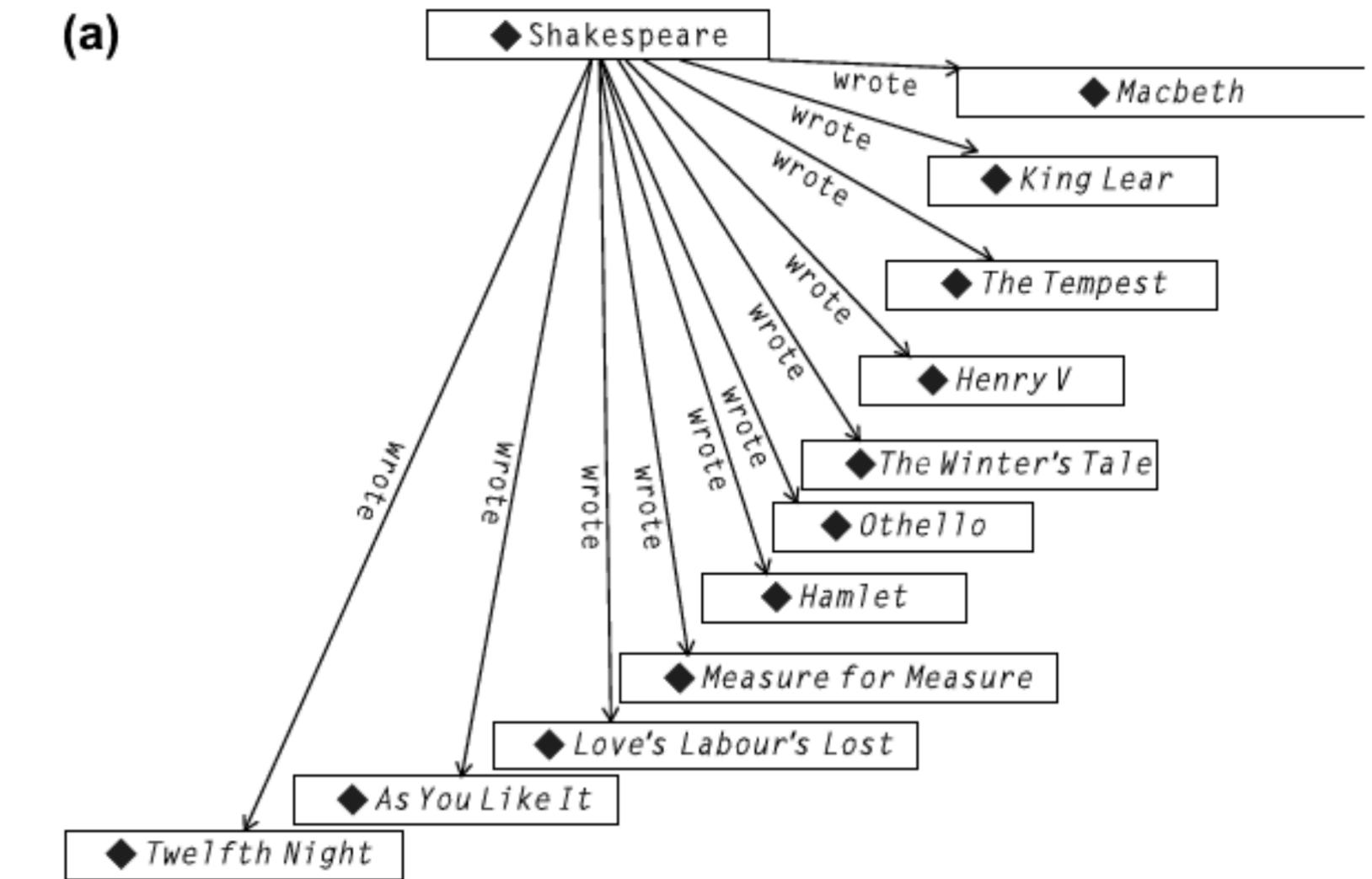
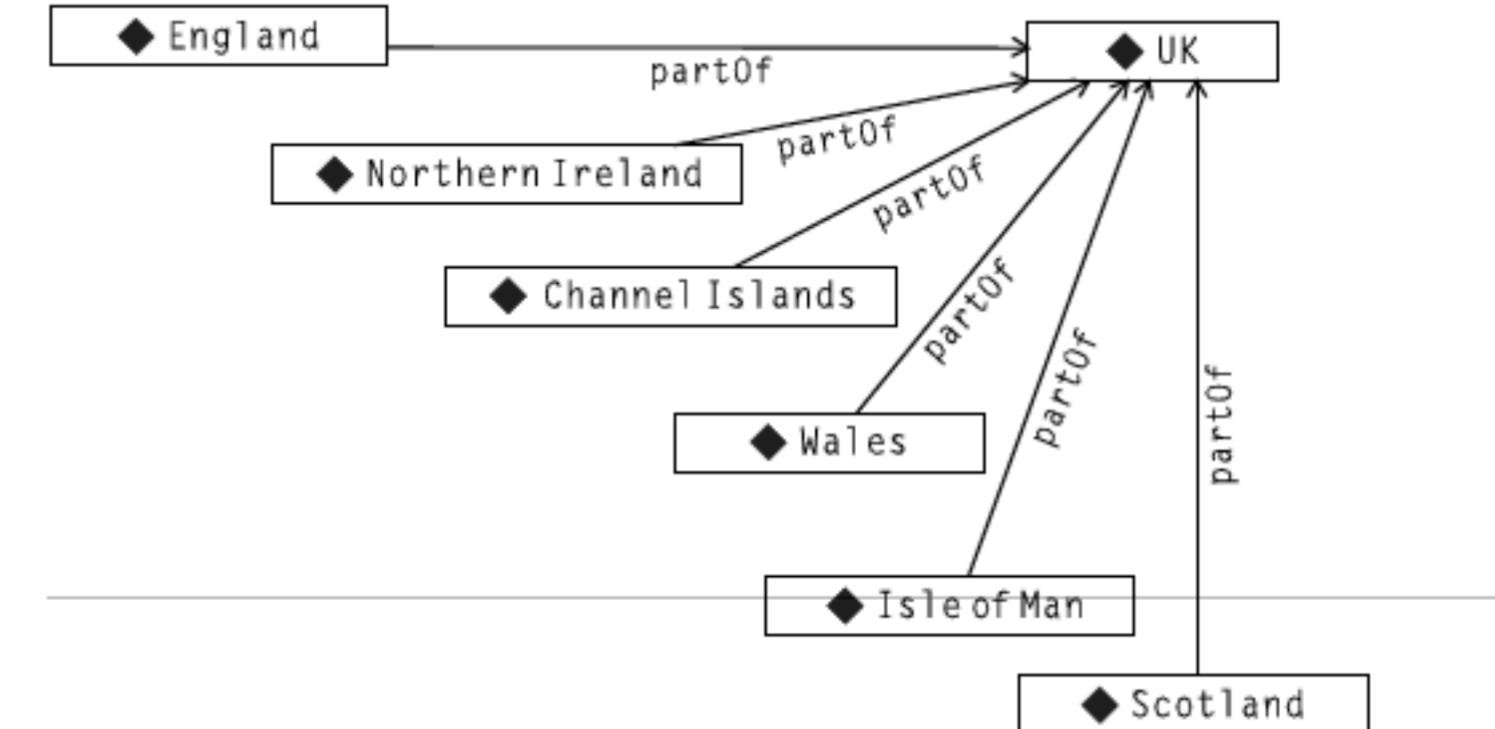


FIGURE 3.4

Graph display of triples from Table 3.3. Eight triples appear as eight labeled edges.



(b)



Merging data from multiple sources

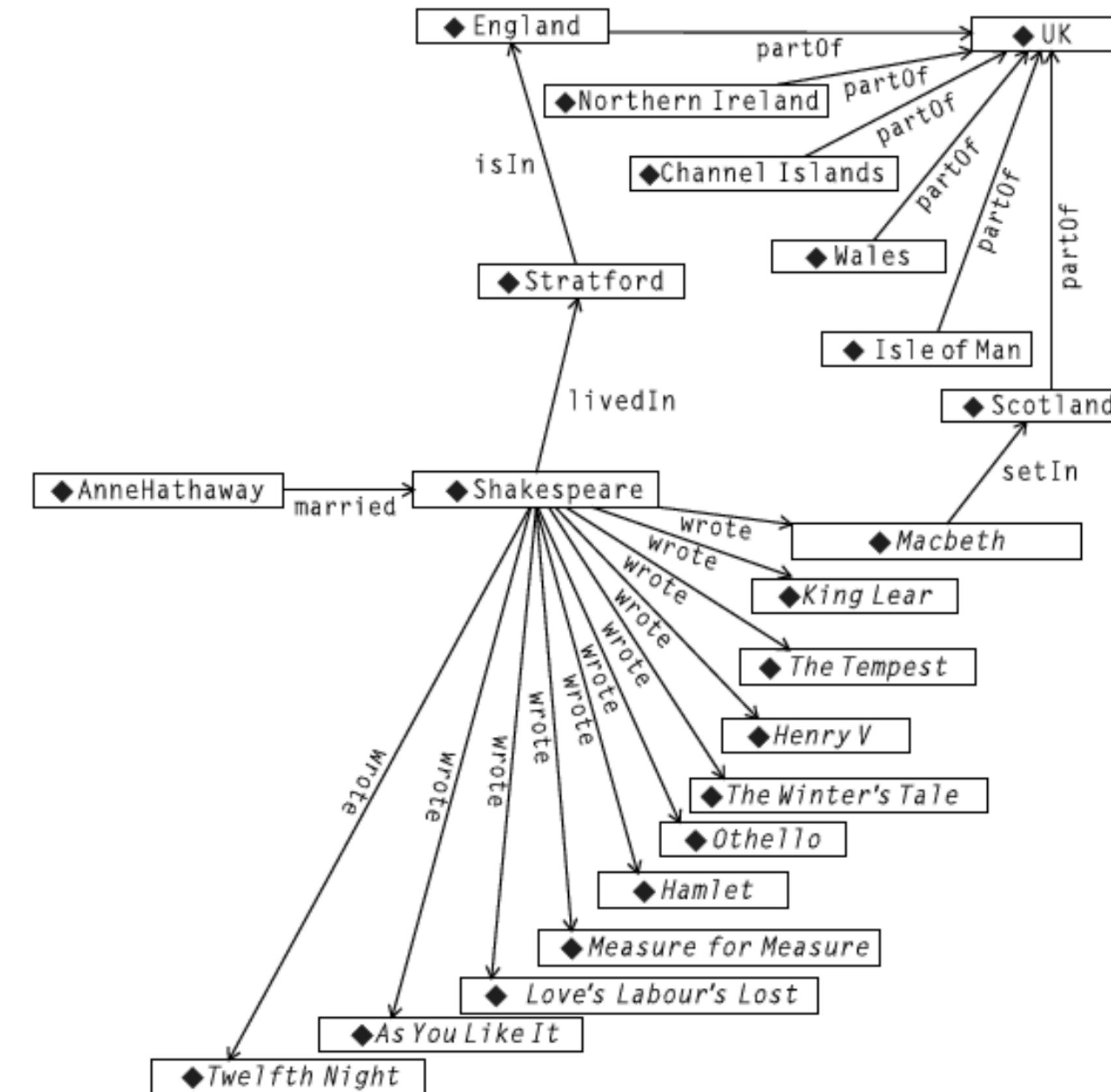


FIGURE 3.6

Combined graph of all triples about Shakespeare and the United Kingdom.

Merging data from multiple sources

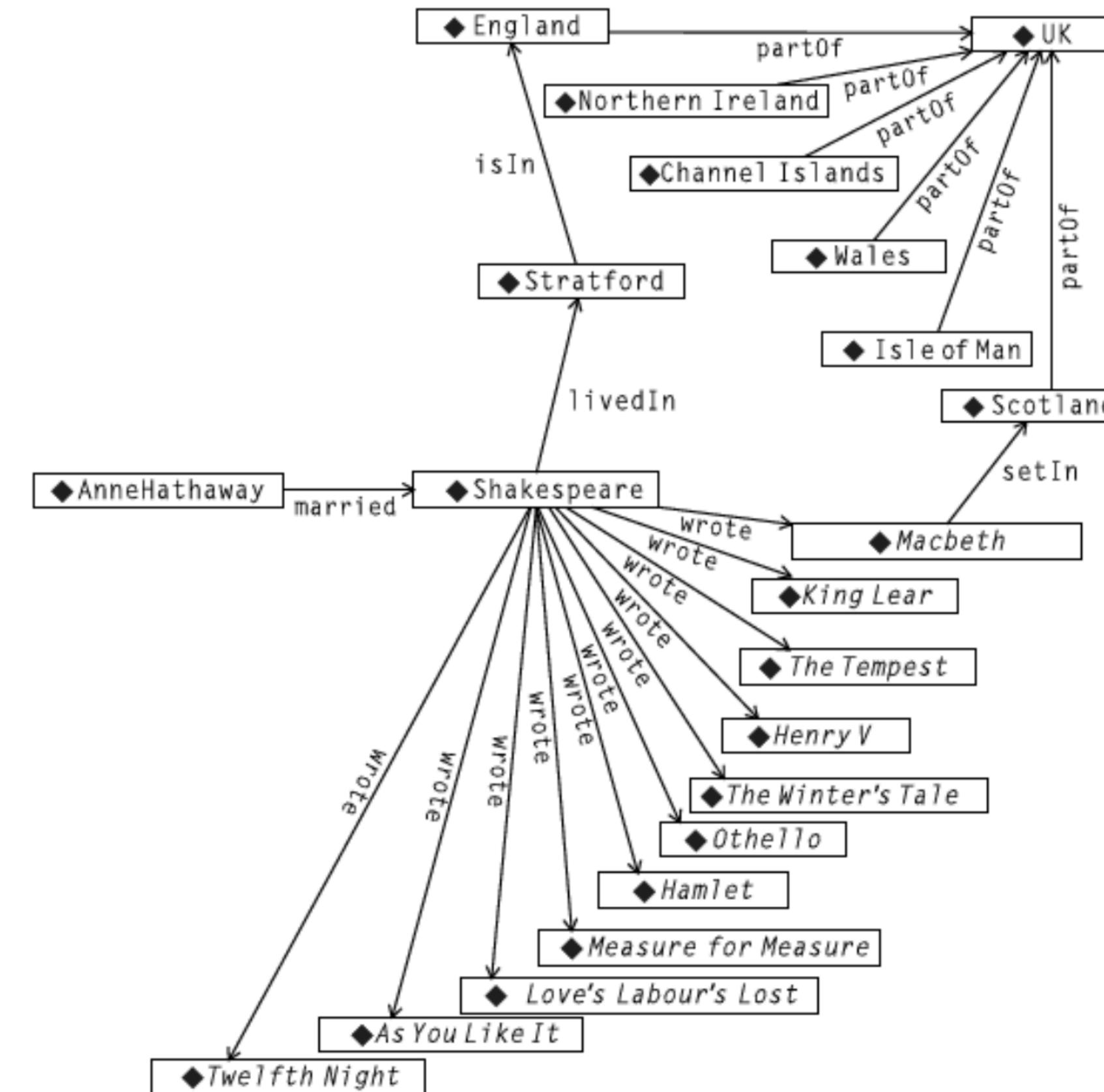


FIGURE 3.6

Combined graph of all triples about Shakespeare and the United Kingdom.

(As long as we know which nodes in the graph of each source match)

Merging data from multiple sources

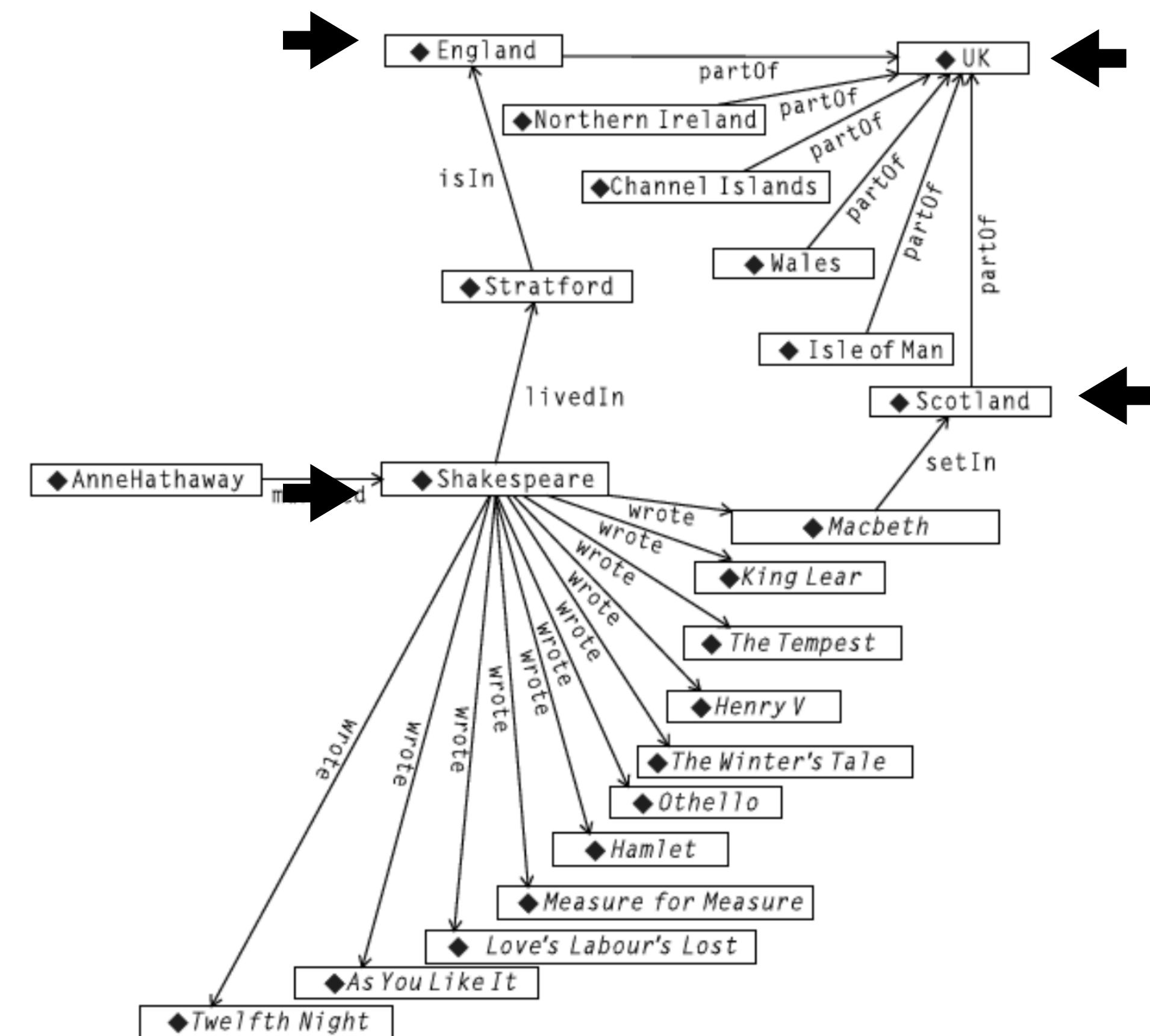


FIGURE 3.6

Combined graph of all triples about Shakespeare and the United Kingdom.

(As long as we know which nodes in the graph of each source match)

When are two resources the same?

When are two resources the same?

- A given name in natural language (e.g. “Washington”) may refer to many different concepts: the president, the state, the capital of the US...

When are two resources the same?

- A given name in natural language (e.g. “Washington”) may refer to many different concepts: the president, the state, the capital of the US...
- Within RDF, any identifiable concept (node in the graph) is called a “resource”

When are two resources the same?

- A given name in natural language (e.g. “Washington”) may refer to many different concepts: the president, the state, the capital of the US...
- Within RDF, any identifiable concept (node in the graph) is called a “resource”
- Two resources are the same if they share the same **Uniform Resource Identifier (URI)**

URIs vs URL

URIs vs URL

- **URI:** Uniform Resource *Identifier*

URIs vs URL

- **URI:** Uniform Resource *Identifier*
 - Focus on the model

URIs vs URL

- **URI:** Uniform Resource *Identifier*
 - Focus on the model

URIs vs URL

- **URI:** Uniform Resource *Identifier*
 - Focus on the model
- **URL:** Uniform Resource *Locator*

URIs vs URL

- **URI:** Uniform Resource *Identifier*
 - Focus on the model
- **URL:** Uniform Resource *Locator*
 - Focus on retrieval

URIs vs URL

- **URI:** Uniform Resource *Identifier*
 - Focus on the model
- **URL:** Uniform Resource *Locator*
 - Focus on retrieval

URIs vs URL

- **URI:** Uniform Resource *Identifier*
 - Focus on the model
- **URL:** Uniform Resource *Locator*
 - Focus on retrieval
- URL is a special case of URI (every URL is an URI but not vice-versa)

URIs example and abreviation

URIs example and abbreviation

- A fully expressed URI looks like an URL. Example:

URIs example and abbreviation

- A fully expressed URI looks like an URL. Example:
- [https://dbpedia.org/page/William Shakespeare](https://dbpedia.org/page/William_Shakespeare) is the URI of the DBpedia entry on William Shakespeare

URIs example and abbreviation

- A fully expressed URI looks like an URL. Example:
- [https://dbpedia.org/page/William Shakespeare](https://dbpedia.org/page/William_Shakespeare) is the URI of the DBpedia entry on William Shakespeare
- For ease of readability, it is also common to express it with a *qname* , consisting of a namespace and identifier

URIs example and abreviation

URIs example and abbreviation

- The binding between a namespace and the URI is local to each application.

URIs example and abbreviation

- The binding between a namespace and the URI is local to each application.
- Ex:

URIs example and abbreviation

- The binding between a namespace and the URI is local to each application.

- Ex:

after declaring (in turtle notation)

URIs example and abbreviation

- The binding between a namespace and the URI is local to each application.
- Ex:

after declaring (in turtle notation)

```
@prefix dbr:<http://dbpedia.org/resource/> .
```

URIs example and abbreviation

- The binding between a namespace and the URI is local to each application.
- Ex:

after declaring (in turtle notation)

```
@prefix dbr:<http://dbpedia.org/resource/> .
```

URIs example and abbreviation

- The binding between a namespace and the URI is local to each application.
- Ex:

after declaring (in turtle notation)

```
@prefix dbr:<http://dbpedia.org/resource/> .
```

[https://dbpedia.org/page/William Shakespeare](https://dbpedia.org/page/William_Shakespeare) becomes
dbr:William_Shakespeare

URIs example and abbreviation

- The binding between a namespace and the URI is local to each application.
- Ex:

after declaring (in turtle notation)

```
@prefix dbr:<http://dbpedia.org/resource/> .
```

[https://dbpedia.org/page/William Shakespeare](https://dbpedia.org/page/William_Shakespeare) becomes
dbr:William_Shakespeare

URIs example and abbreviation

- The W3C also defines a few standard namespaces with important predicates:

rdf: Indicates identifiers used in RDF. The set of identifiers defined in the standard is quite small and is used to define types and properties in RDF. The global URI for the *rdf* namespace is [*http://www.w3.org/1999/02/22-rdf-syntax-ns#*](http://www.w3.org/1999/02/22-rdf-syntax-ns#).

rdfs: Indicates identifiers used for the RDF Schema language, RDFS. The scope and semantics of the symbols in this namespace are the topics of future chapters. The global URI for the *rdfs* namespace is [*http://www.w3.org/2000/01/rdf-schema#*](http://www.w3.org/2000/01/rdf-schema#).

owl: Indicates identifiers used for the Web Ontology Language, OWL. The scope and semantics of the symbols in this namespace are the topics of future chapters. The global URI for the *owl* namespace is [*http://www.w3.org/2002/07/owl#*](http://www.w3.org/2002/07/owl#).

See:

<https://www.w3.org/1999/02/22-rdf-syntax-ns>

<https://www.w3.org/2000/01/rdf-schema#>

<https://www.w3.org/2002/07/owl#Thing>

Important RDF resources

rdf:type

- Provides an elementary typing system
- Types themselves can also have types

Table 3.9 Using `rdf:type` to Describe Playwrights

Subject	Predicate	Object
lit:Shakespeare	<code>rdf:type</code>	lit:Playwright
lit:Ibsen	<code>rdf:type</code>	lit:Playwright
lit:Simon	<code>rdf:type</code>	lit:Playwright
lit:Miller	<code>rdf:type</code>	lit:Playwright
lit:Marlowe	<code>rdf:type</code>	lit:Playwright
lit:Wilder	<code>rdf:type</code>	lit:Playwright

Table 3.10 Defining Types of Names

Subject	Predicate	Object
lit:Playwright	<code>rdf:type</code>	bus:Profession
bus:Profession	<code>rdf:type</code>	hr:Compensation

Important RDF resources

rdf:property

- Indicates that something is to be used as a predicate rather than as a subject or object
- Typically not enforced by inference engines
- Still useful for reasoning, semantic clarity and documentation

Table 3.11 `rdf:Property` Assertions for Tables 3.5 to 3.8

Subject	Predicate	Object
lit:wrote	<code>rdf:type</code>	<code>rdf:Property</code>
geo:partOf	<code>rdf:type</code>	<code>rdf:Property</code>
bio:married	<code>rdf:type</code>	<code>rdf:Property</code>
bio:livedIn	<code>rdf:type</code>	<code>rdf:Property</code>
bio:livedWith	<code>rdf:type</code>	<code>rdf:Property</code>
geo:isIn	<code>rdf:type</code>	<code>rdf:Property</code>

RDF and Tabular Data

How to convert tabular data to RDF?

Product						
ID	Model Number	Division	Product Line	Manufacture Location	SKU	Available
1	ZX-3	Manufacturing support	Paper machine	Sacramento	FB3524	23
2	ZX-3P	Manufacturing support	Paper machine	Sacramento	KD5243	4
3	ZX-3S	Manufacturing support	Paper machine	Sacramento	IL4028	34
4	B-1430	Control engineering	Feedback line	Elizabeth	KS4520	23
5	B-1430X	Control engineering	Feedback line	Elizabeth	CL5934	14
6	B-1431	Control engineering	Active sensor	Seoul	KK3945	0
7	DBB-12	Accessories	Monitor	Hong Kong	ND5520	100
8	SP-1234	Safety	Safety valve	Cleveland	HI4554	4
9	SPX-1234	Safety	Safety valve	Cleveland	OP5333	14

RDF and Tabular Data

RDF and Tabular Data

- The whole database becomes a namespace (in the example, ‘mfg:’)

RDF and Tabular Data

- The whole database becomes a namespace (in the example, ‘mfg:’)
- Each table becomes a type (in the example, ‘mfg:Product’)

RDF and Tabular Data

- The whole database becomes a namespace (in the example, ‘mfg:’)
- Each table becomes a type (in the example, ‘mfg:Product’)
- Each line gets each own URI

RDF and Tabular Data

- The whole database becomes a namespace (in the example, ‘mfg:’)
- Each table becomes a type (in the example, ‘mfg:Product’)
- Each line gets each own URI
 - e.g. mfg:Product 1, mfg:Product2, etc

RDF and Tabular Data

Table 3.14 Triples Representing Type of Information from Table 3.12

Subject	Predicate	Object
mfg:Product1	rdf:type	mfg:Product
mfg:Product2	rdf:type	mfg:Product
mfg:Product3	rdf:type	mfg:Product
mfg:Product4	rdf:type	mfg:Product
mfg:Product5	rdf:type	mfg:Product
mfg:Product6	rdf:type	mfg:Product
mfg:Product7	rdf:type	mfg:Product
mfg:Product8	rdf:type	mfg:Product
mfg:Product9	rdf:type	mfg:Product

RDF and Tabular Data

RDF and Tabular Data

- For each line, we state that element to be of the appropriate type

RDF and Tabular Data

- For each line, we state that element to be of the appropriate type
 - ▶ e.g. mfg:Product1 rdf:type mfg:product

RDF and Tabular Data

- For each line, we state that element to be of the appropriate type
 - ▶ e.g. mfg:Product1 rdf:type mfg:product
- Each column becomes a property, each cell becomes a triple

RDF and Tabular Data

- For each line, we state that element to be of the appropriate type
 - ▶ e.g. mfg:Product1 rdf:type mfg:product
- Each column becomes a property, each cell becomes a triple
 - ▶ e.g. mfg:Product1 mfg:Product_ModelNo “ZX-3”

RDF and Tabular Data

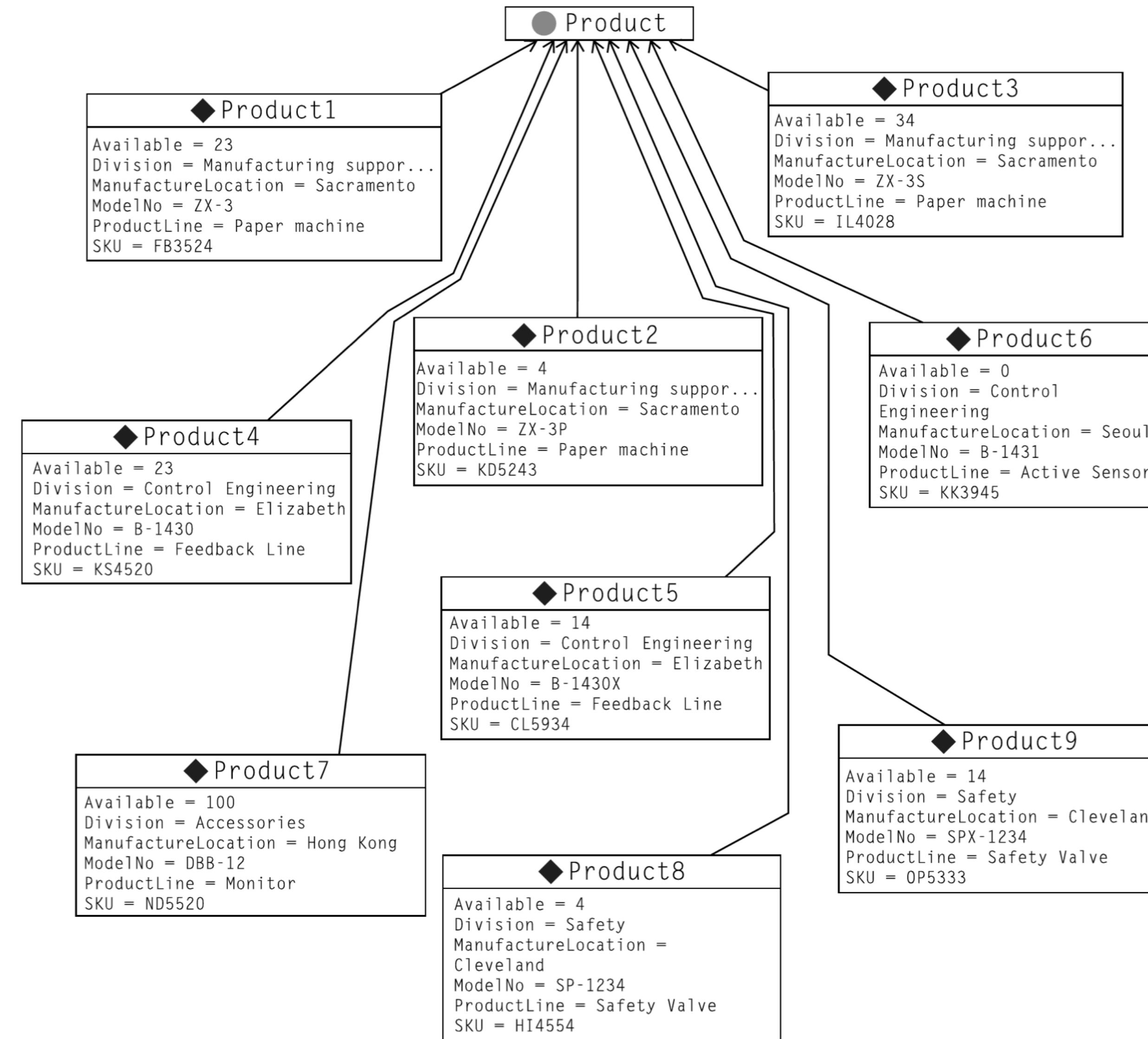
- For each line, we state that element to be of the appropriate type
 - ▶ e.g. mfg:Product1 rdf:type mfg:product
- Each column becomes a property, each cell becomes a triple
 - ▶ e.g. mfg:Product1 mfg:Product_ModelNo “ZX-3”
- Object of each triple may be a literal data type (Integer, String) rather than a resource

RDF and Tabular Data

Table 3.13 Triples Representing Some of the Data in Table 3.12

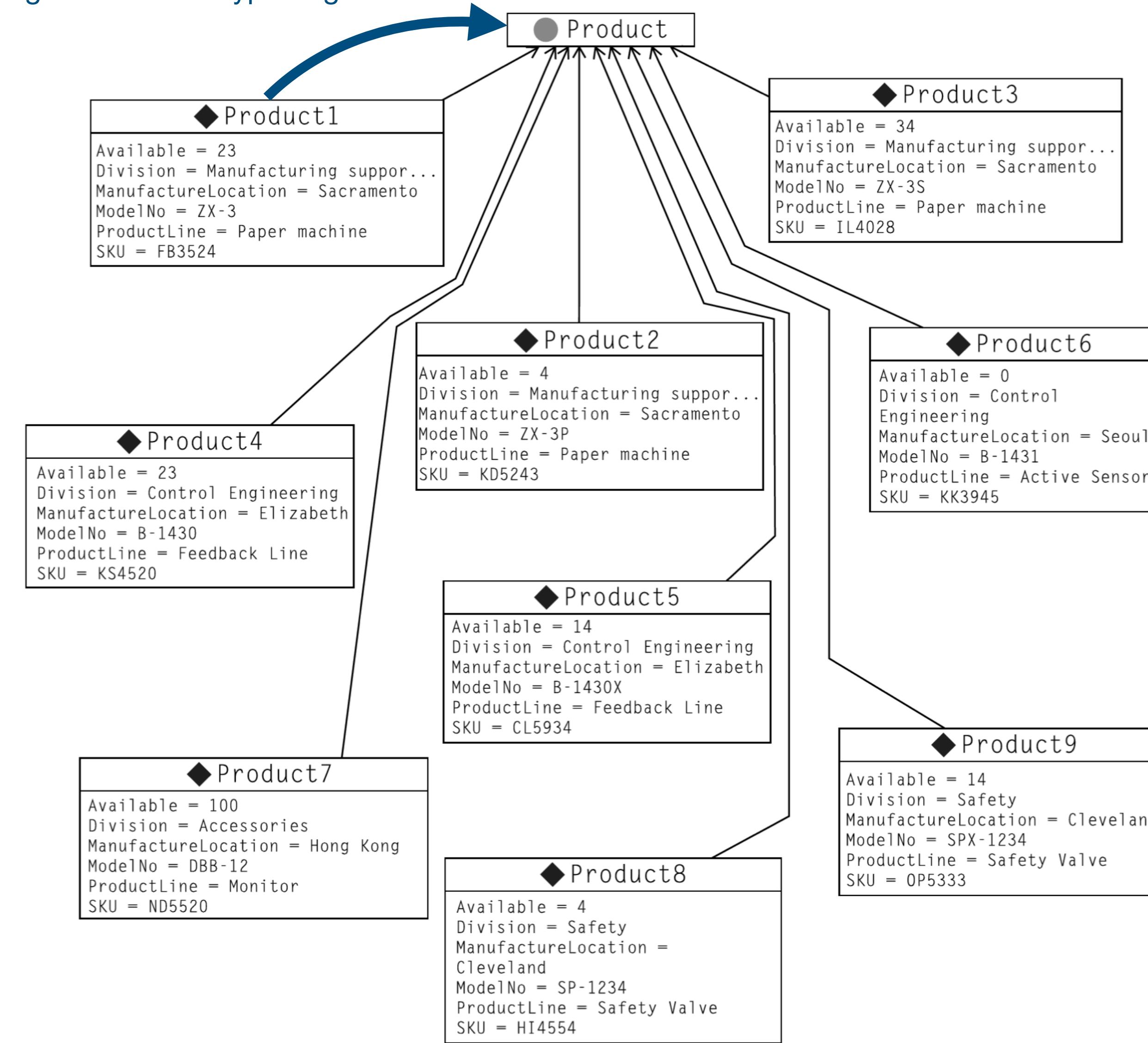
Subject	Predicate	Object
mfg:Product1	mfg:Product_ID	1
mfg:Product1	mfg:Product_ModelNo	ZX-3
mfg:Product1	mfg:Product_Division	Manufacturing support
mfg:Product1	mfg:Product_Product_Line	Paper machine
mfg:Product1	mfg:Product_Manufacture_Location	Sacramento
mfg:Product1	mfg:Product_SKU	FB3524
mfg:Product1	mfg:Product_Available	23
mfg:Product2	mfg:Product_ID	2
mfg:Product2	mfg:Product_ModelNo	ZX-3P
mfg:Product2	mfg:Product_Division	Manufacturing support
mfg:Product2	mfg:Product_Product_Line	Paper machine
mfg:Product2	mfg:Product_Manufacture_Location	Sacramento
mfg:Product2	mfg:Product_SKU	KD5243
mfg:Product2	mfg:Product_Available	4...

RDF and Tabular Data



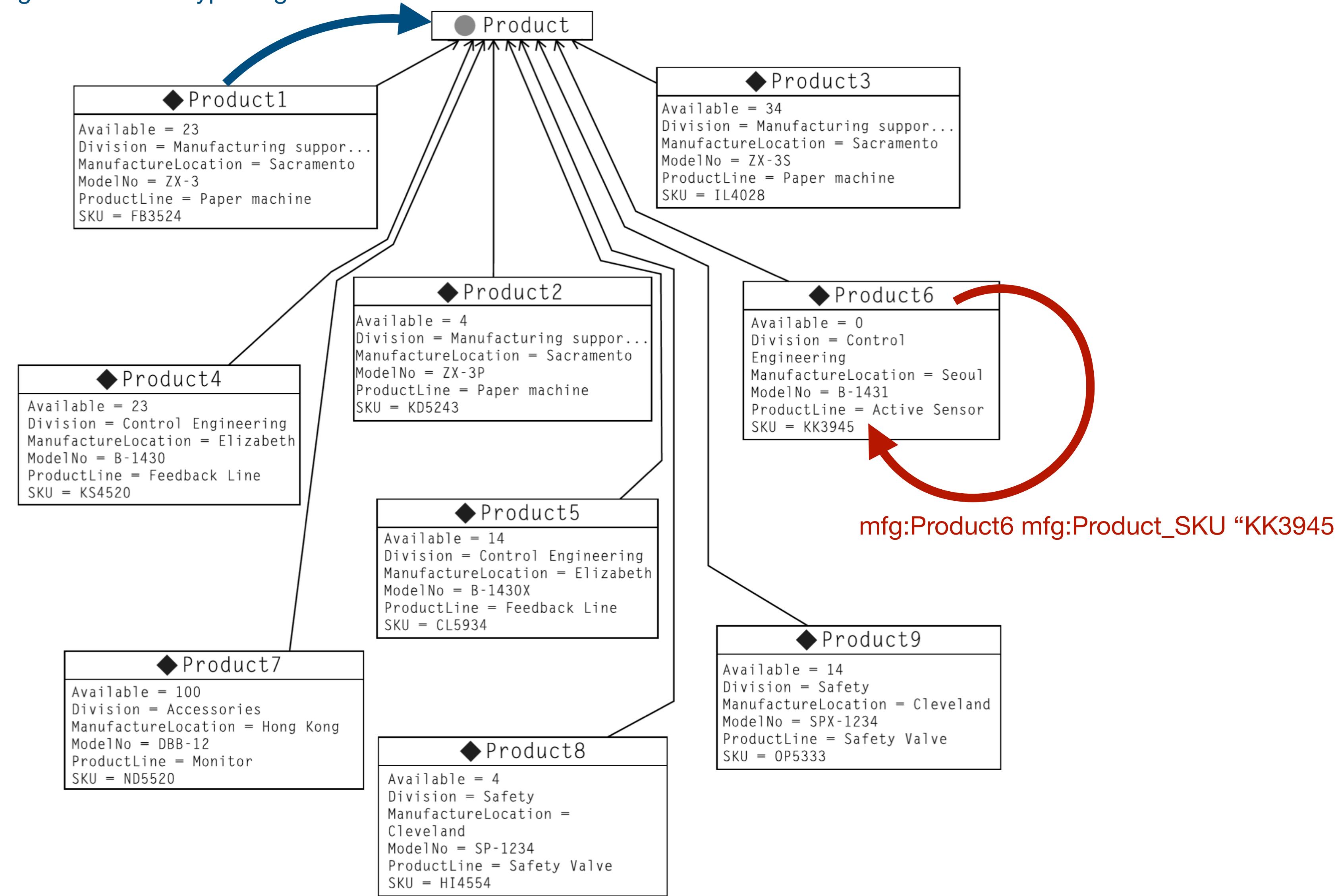
RDF and Tabular Data

mfg:Product1 rdf:type mfg:Product



RDF and Tabular Data

mfg:Product1 rdf:type mfg:Product



What about higher-order relationships?

What about higher-order relationships?

- How do we state the following with triplets?

What about higher-order relationships?

- How do we state the following with triplets?
 - “Shakespeare wrote Hamlet”

What about higher-order relationships?

- How do we state the following with triplets?
 - “Shakespeare wrote Hamlet”
 - “Shakespeare wrote Hamlet in 1604”

What about higher-order relationships?

- How do we state the following with triplets?
 - “Shakespeare wrote Hamlet”
 - “Shakespeare wrote Hamlet in 1604”
 - “Wikipedia states that Shakespeare wrote Hamlet in 1604”

What about higher-order relationships?

What about higher-order relationships?

- “Shakespeare wrote Hamlet”

What about higher-order relationships?

- “Shakespeare wrote Hamlet”
 - Can be represented in rdf similarly to English

What about higher-order relationships?

- “Shakespeare wrote Hamlet”
 - Can be represented in rdf similarly to English
 - ▶ :Shakespeare :wrote :Hamlet

What about higher-order relationships?

What about higher-order relationships?

- “Shakespeare wrote Hamlet in 1604”

What about higher-order relationships?

- “Shakespeare wrote Hamlet in 1604”
 - Implicit reification

What about higher-order relationships?

- “Shakespeare wrote Hamlet in 1604”
 - Implicit reification
 - ▶ :n1 :author :Shakespeare

What about higher-order relationships?

- “Shakespeare wrote Hamlet in 1604”
 - Implicit reification
 - ▶ :n1 :author :Shakespeare
 - ▶ :n1 :title :Hamlet

What about higher-order relationships?

- “Shakespeare wrote Hamlet in 1604”
 - Implicit reification
 - ▶ :n1 :author :Shakespeare
 - ▶ :n1 :title :Hamlet
 - ▶ :n1 :publicationDate 1604

What about higher-order relationships?

- “Shakespeare wrote Hamlet in 1604”
 - Implicit reification
 - ▶ :n1 :author :Shakespeare
 - ▶ :n1 :title :Hamlet
 - ▶ :n1 :publicationDate 1604
 - Reification is implicit because the identifier n1 still represents Hamlet (an object in the domain)

What about higher-order relationships?

What about higher-order relationships?

- “Wikipedia states Shakespeare wrote Hamlet in 1604”

What about higher-order relationships?

- “Wikipedia states Shakespeare wrote Hamlet in 1604”
- Consider the simpler “Wikipedia states Shakespeare wrote Hamlet”

What about higher-order relationships?

- “Wikipedia states Shakespeare wrote Hamlet in 1604”
- Consider the simpler “Wikipedia states Shakespeare wrote Hamlet”
 - Explicit reification

What about higher-order relationships?

- “Wikipedia states Shakespeare wrote Hamlet in 1604”
- Consider the simpler “Wikipedia states Shakespeare wrote Hamlet”
 - Explicit reification
 - ▶ :n2 rdf:subject :Shakespeare

What about higher-order relationships?

- “Wikipedia states Shakespeare wrote Hamlet in 1604”
- Consider the simpler “Wikipedia states Shakespeare wrote Hamlet”
 - Explicit reification
 - ▶ :n2 rdf:subject :Shakespeare
 - ▶ :n2 rdf: predicate :wrote

What about higher-order relationships?

- “Wikipedia states Shakespeare wrote Hamlet in 1604”
- Consider the simpler “Wikipedia states Shakespeare wrote Hamlet”
 - Explicit reification
 - ▶ :n2 rdf:subject :Shakespeare
 - ▶ :n2 rdf:predicate :wrote
 - ▶ :n2 rdf:object :Hamlet

What about higher-order relationships?

- “Wikipedia states Shakespeare wrote Hamlet in 1604”
- Consider the simpler “Wikipedia states Shakespeare wrote Hamlet”
 - Explicit reification
 - ▶ :n2 rdf:subject :Shakespeare
 - ▶ :n2 rdf:predicate :wrote
 - ▶ :n2 rdf:object :Hamlet
 - ▶ :n2 :attributedTo :Wikipedia

What about higher-order relationships?

- “Wikipedia states Shakespeare wrote Hamlet in 1604”
- Consider the simpler “Wikipedia states Shakespeare wrote Hamlet”
 - Explicit reification
 - ▶ :n2 rdf:subject :Shakespeare
 - ▶ :n2 rdf:predicate :wrote
 - ▶ :n2 rdf:object :Hamlet
 - ▶ :n2 :attributedTo :Wikipedia
 - Reification is explicit because n2 represents a statement (not an object)

What about higher-order relationships?

- “Wikipedia states Shakespeare wrote Hamlet in 1604”
- Consider the simpler “Wikipedia states Shakespeare wrote Hamlet”
 - Explicit reification
 - ▶ :n2 rdf:subject :Shakespeare
 - ▶ :n2 rdf:predicate :wrote
 - ▶ :n2 rdf:object :Hamlet
 - ▶ :n2 :attributedTo :Wikipedia
 - Reification is explicit because n2 represents a statement (not an object)
 - Similarly, we could construct another statement n3 (“Wikipedia states Hamlet was written in 1604”)

Modern uses of Semantic Web Technologies

Google Knowledge Graph

Google Knowledge Graph

文 A 24 languages ▾

Article Talk

Read Edit View history Tools ▾

From Wikipedia, the free encyclopedia

"Knowledge Graph" redirects here. For the general concept in information science, see [Knowledge graph](#). For other uses, see [Knowledge Graph \(disambiguation\)](#).



A request that this article title be changed to [Knowledge Graph](#) is under discussion.

Please do not move this article until the discussion is closed.

The **Google Knowledge Graph** is a [knowledge base](#) from which [Google](#) serves relevant information in an infobox beside its [search results](#). This allows the user to see the answer in a glance, as an [instant answer](#). The data is generated automatically from a variety of sources, covering places, people, businesses, and more. [1][2]

The information covered by Google's Knowledge Graph grew quickly after launch, tripling its data size within seven months (covering 570 million entities and 18 billion facts^[3]). By mid-2016, Google reported that it held 70 billion facts^[4] and answered "roughly one-third" of the 100 billion monthly searches they handled. By May 2020, this had grown to 500 billion facts on 5 billion entities.^[5]

There is no official documentation of how the Google Knowledge Graph is implemented.^[6] According to Google, its information is retrieved from many sources, including the [CIA World Factbook](#) and [Wikipedia](#).^[7] It is used to answer direct spoken questions in [Google Assistant](#)^[8] and [Google Home](#) voice queries.^[10] It has been criticized for providing answers with neither source attribution nor [citations](#).^[11]

History [edit]

Google announced its Knowledge Graph on May 16, 2012, as a way to significantly enhance the value of information returned by Google searches.^[7] Initially available only in English, it was expanded in December 2012 to [Spanish](#), [French](#), [German](#), [Portuguese](#), [Japanese](#), [Russian](#) and [Italian](#).^[12] [Bengali](#) support was added in March 2017.^[13]



Thomas Jefferson

3rd U.S. President

Thomas Jefferson was an American Founding Father, the principal author of the Declaration of Independence, and the third President of the United States. [Wikipedia](#)

Born: April 13, 1743, Shadwell, VA

Died: July 4, 1826, Charlottesville, VA

Presidential term: March 4, 1801 – March 4, 1809

Spouse: [Martha Jefferson](#) (m. 1772–1782)

Party: Democratic-Republican Party

Awards: AIA Gold Medal

Get updates about Thomas Jefferson

Keep me updated

People also search for



View 15+ more

Knowledge panel data about

Thomas Jefferson displayed on [Google Search](#), as of January 2015

Tagging in online shopping

The screenshot shows a product page for the Bose QuietComfort Ultra Wireless Noise Cancelling Over-the-Ear Headphones on the Best Buy website. The page features a large image of the headphones, a prominent price of \$429.00, and a financing option of \$35.75 per month. Below the main content, there are links for 'Finance Options' and 'Special Offers'. At the bottom of the page, a developer tools browser console is open, showing the selected JSON script element containing the product schema data.

```
!!--$-->
script id="product-schema" type="application/ld+json">> = $0
{
  "@context": "http://schema.org/",
  "@type": "Product",
  "name": "Bose - QuietComfort Ultra Wireless Noise Cancelling Over-the-Ear Headphones - Black",
  "image": [],
  "url": "https://www.bestbuy.com/site/bose-quietcomfort-ultra-wireless-noise-cancelling-over-the-ear-headphones-black/6554464.p?skuId=6554464",
  "Brand": {
    "name": "Bose"
  },
  "aggregateRating": {
    "ratingValue": "4.7",
    "reviewCount": 1824,
    "url": "https://www.bestbuy.com/site/reviews/name/6554464"
  },
  "Organization": {
    "name": "Best Buy"
  },
  "availableDeliveryMethod": [
    "PICKUP",
    "SHIPPING"
  ],
  "Offer": {
    "priceCurrency": "USD",
    "price": 371.99,
    "availability": "https://schema.org/InStock",
    "itemCondition": "https://schema.org/UsedCondition",
    "description": "Open-Box"
  },
  "Offer": {
    "priceCurrency": "USD",
    "price": 363.99,
    "availability": "https://schema.org/InStock",
    "itemCondition": "https://schema.org/UsedCondition",
    "description": "Open-Box"
  },
  "Offer": {
    "priceCurrency": "USD",
    "price": 348.99,
    "availability": "https://schema.org/InStock",
    "itemCondition": "https://schema.org/UsedCondition",
    "description": "Open-Box"
  },
  "Person": {
    "name": "StephenT"
  },
  "reviewBody": "Terrific noise canceling ability, decent bass when turned up on the equalizer in the app, comfortable faux leather padded motion tracking modes in the app, superb 40hr battery life, super clear built in mic. Great for everyday wear 8.5/10 could use a little more oomph in the bass department, n them.",
  "reviewRating": {
    "ratingValue": "5",
    "bestRating": "5"
  },
  "publisher": {
    "Organization": {
      "name": "Best Buy"
    }
  },
  "Review": {
    "name": "Supreme choice headphones. From the moment you put them on, you're greeted with unparalleled comfort-thanks to the plush ear cushions and lightweight design, making them perfect for long listening to music, podcasts, or watching movies, the sound experience is immersive and detailed.\n\nNoise cancellation on these headphones is next-level. It effectively seals out most ambient noise, allowing you to fully immerse yourself in your music or calls. The secure fit ensures they stay in place during physical activity like running or working out. Whether I'm on the course, at the gym, or commuting, I know they'll last all day without any interruptions.\n\nThe touch controls and seamless Bluetooth connectivity are also impressive. They're a fantastic investment for audiophiles, frequent travelers, or anyone who wants to enjoy high-quality sound on the go. The quietcomfort ultra came out on top.\n\nThe deciding factors for me were comfort, weight, features for the price, and noise cancellation. First off, these headphones are very comfortable and have a great overall fit. The weight is just right, not too heavy or too light. The noise cancellation is excellent, blocking out most ambient noise. The sound quality is rich and balanced, and the companion app offers plenty of customization options to tweak it just right.\n\nOverall, these headphones are a great investment. They're an investment in comfort and quality sound. Bose knocked it out of the park with these.", "ratingValue": "5", "bestRating": "5", "author": "StephenT", "datePublished": "2023-05-14T12:00:00Z", "url": "https://www.bestbuy.com/site/reviews/name/6554464#review-1824"}, "reviewRating": {
    "ratingValue": "5",
    "bestRating": "5"
  }
}
```

Source: BestBuy (access May 14, 2025)

LLMs and Knowledge Graphs

Modern research on integration (both ways) between KGs and LLMs

The screenshot shows a Google Scholar search interface. The search query is "knowledge graphs" and llms. The results page displays 14,700 articles. The sidebar includes filters for time (Any time, Since 2025, Since 2024, Since 2021, Custom range...), sorting (Sort by relevance, Sort by date), and document type (Any type, Review articles). It also has checkboxes for include patents and include citations, and a 'Create alert' button.

Search Query: "knowledge graphs" and llms

Results: About 14,700 results (0.10 sec)

Filters:

- Any time
- Since 2025
- Since 2024
- Since 2021
- Custom range...
- Sort by relevance
- Sort by date
- Any type
- Review articles
- include patents
- include citations
- Create alert

Articles:

- Unifying large language models and knowledge graphs: A roadmap**
S Pan, L Luo, Y Wang, C Chen... - IEEE Transactions on ..., 2024 - ieexexplore.ieee.org
... solution is to incorporate knowledge graphs (KGs) into LLMs. ... LLM pre-training, which aims to inject knowledge into LLMs ... enhanced LLM inference, which enables LLMs to consider ...
☆ Save ⚡ Cite Cited by 1034 Related articles All 9 versions Web of Science: 176
- Knowledge solver: Teaching llms to search for domain knowledge from knowledge graphs**
C Feng, X Zhang, Z Fei - arXiv preprint arXiv:2309.03118, 2023 - arxiv.org
... Our approach helps LLMs search for necessary knowledge to perform tasks by harnessing LLMs' ... with the second group, with an emphasis on integrating Knowledge Graphs into LLMs. ...
☆ Save ⚡ Cite Cited by 64 Related articles All 2 versions
- A survey on augmenting knowledge graphs (KGs) with large language models (LLMs): models, evaluation metrics, benchmarks, and challenges**
N Ibrahim, S Aboulela, A Ibrahim, R Kashef - Discover Artificial Intelligence, 2024 - Springer
... LLMs, which integrates knowledge graphs to enhance LLMs performance and interpretability; LLMs-augmented KG, whereby LLMs ... of Knowledge Graphs; and Synergized LLMs + KG, ...
☆ Save ⚡ Cite Cited by 15 Related articles All 3 versions
- Can knowledge graphs reduce hallucinations in llms?: A survey**
G Agrawal, T Kumarage, Z Alghamdi, H Liu - arXiv preprint arXiv ..., 2023 - arxiv.org
... LLM augmentation methods utilizing structured knowledge from knowledge graphs. Specifically, our emphasis is on addressing hallucinations in LLMs through KG integration. ...
☆ Save ⚡ Cite Cited by 122 Related articles All 7 versions
- [PDF] Research trends for the interplay between large language models and knowledge graphs**
H Khorashadizadeh, FZ Amara, M Ezzabady... - arXiv preprint arXiv ..., 2024 - vldb.org
... This survey investigates the synergistic relationship between Large Language Models (LLMs) and Knowledge Graphs (KGs), which is crucial for advancing AI's understanding, ...
☆ Save ⚡ Cite Cited by 18 Related articles All 4 versions
- Combining knowledge graphs and large language models**
A Kau, X He, A Nambissan, A Astudillo, H Yin... - arXiv preprint arXiv ..., 2024 - arxiv.org
... This work collected 28 papers outlining methods for KG-powered LLMs, LLM-based KGs, and LLM-KG hybrid approaches. We systematically analysed and compared these approaches ...
☆ Save ⚡ Cite Cited by 25 Related articles All 2 versions

Links:

- [PDF] ieee.org
- Get It at Cal Poly
- [PDF] arxiv.org
- [PDF] springer.com
- Full View
- [PDF] arxiv.org
- [PDF] vldb.org
- [PDF] arxiv.org

Source: Google Scholar (access May 14, 2025)

Other large projects

Other large projects

- dbpedia.org - open source project extracting structured content from Wikimedia. One of the largest triple stores

Other large projects

- dbpedia.org - open source project extracting structured content from Wikimedia. One of the largest triple stores
- Wikidata Query Service - a SPARQL endpoint for queries on wikidata

Other large projects

- dbpedia.org - open source project extracting structured content from Wikimedia. One of the largest triple stores
- Wikidata Query Service - a SPARQL endpoint for queries on wikidata
- schema.org - A community-maintained ontology founded by Google, Microsoft, Yahoo and Yandex

Other large projects

- dbpedia.org - open source project extracting structured content from Wikimedia. One of the largest triple stores
- Wikidata Query Service - a SPARQL endpoint for queries on wikidata
- schema.org - A community-maintained ontology founded by Google, Microsoft, Yahoo and Yandex
- LOD (Linked Open Data) Cloud - lod-cloud.net

Other large projects

- dbpedia.org - open source project extracting structured content from Wikimedia. One of the largest triple stores
- Wikidata Query Service - a SPARQL endpoint for queries on wikidata
- schema.org - A community-maintained ontology founded by Google, Microsoft, Yahoo and Yandex
- LOD (Linked Open Data) Cloud - lod-cloud.net
- Biomedical databases:

Other large projects

- dbpedia.org - open source project extracting structured content from Wikimedia. One of the largest triple stores
- Wikidata Query Service - a SPARQL endpoint for queries on wikidata
- schema.org - A community-maintained ontology founded by Google, Microsoft, Yahoo and Yandex
- LOD (Linked Open Data) Cloud - lod-cloud.net
- Biomedical databases:
 - <https://geneontology.org/>

Turtle notation (serialization)

Serialization

Serialization

- How to represent triples on text (or in a file)?

Serialization

- How to represent triples on text (or in a file)?
- Fully writing down each triple (N-Triples) is clunky and not very readable

Serialization

- How to represent triples on text (or in a file)?
- Fully writing down each triple (N-Triples) is clunky and not very readable
- Turtle format is particularly suited for human readability

Serialization

- How to represent triples on text (or in a file)?
- Fully writing down each triple (N-Triples) is clunky and not very readable
- Turtle format is particularly suited for human readability
 - Our focus here

Serialization

- How to represent triples on text (or in a file)?
- Fully writing down each triple (N-Triples) is clunky and not very readable
- Turtle format is particularly suited for human readability
 - Our focus here
- Other formats, such as RDF/XML are more appropriate for web applications

Serialization

- How to represent triples on text (or in a file)?
- Fully writing down each triple (N-Triples) is clunky and not very readable
- Turtle format is particularly suited for human readability
 - Our focus here
- Other formats, such as RDF/XML are more appropriate for web applications
 - Full specification available: <https://www.w3.org/TR/rdf-syntax-grammar/>

Serialization

- How to represent triples on text (or in a file)?
- Fully writing down each triple (N-Triples) is clunky and not very readable
- Turtle format is particularly suited for human readability
 - Our focus here
- Other formats, such as RDF/XML are more appropriate for web applications
 - Full specification available: <https://www.w3.org/TR/rdf-syntax-grammar/>
- Most triple stores can provide triples in multiple formats

Turtle

Turtle

- Preambles binds local names and global URIs

Turtle

- Preambles binds local names and global URIs

```
@prefix mfg:  
<http://www.WorkingOntologist.com/Examples/Chapter3/Manufac  
turing#>  
@prefix rdf: http://www.w3.org/1999/02/22-rdf-syntax-ns#
```

Turtle

- Preambles binds local names and global URIs

```
@prefix mfg:  
<http://www.WorkingOntologist.com/Examples/Chapter3/Manufac  
turing#>  
@prefix rdf: http://www.w3.org/1999/02/22-rdf-syntax-ns#
```

- Remaining triples can be expressed using the local names

Turtle

- Preambles binds local names and global URIs

```
@prefix mfg:  
<http://www.WorkingOntologist.com/Examples/Chapter3/Manufac  
turing#>  
@prefix rdf: http://www.w3.org/1999/02/22-rdf-syntax-ns#
```

- Remaining triples can be expressed using the local names
 - Triples usually terminate with a period

Turtle

- Preambles binds local names and global URIs

```
@prefix mfg:  
<http://www.WorkingOntologist.com/Examples/Chapter3/Manufac  
turing#>  
@prefix rdf: http://www.w3.org/1999/02/22-rdf-syntax-ns#
```

- Remaining triples can be expressed using the local names
 - Triples usually terminate with a period

```
mfg:Product1 rdf:type mfg:Product .
```

Turtle

- Commas ',' separate triples with repeat subjects AND predicate

Turtle

- Commas ',' separate triples with repeat subjects AND predicate

```
lit:Shakespeare b:hasChild b:Susanna, b:Judith, b:Hamnet.
```

There are actually three triples represented here—namely:

```
lit:Shakespeare b:hasChild b:Susanna.
```

```
lit:Shakespeare b:hasChild b:Judith.
```

```
lit:Shakespeare b:hasChild b:Hamnet.
```

Turtle

Turtle

- `rdf:type` is often abbreviated by *a*

Turtle

- `rdf:type` is often abbreviated by *a*
 - Rather than say “X has type Y”, say “X is a Y”

Turtle

- `rdf:type` is often abbreviated by `a`
 - Rather than say “X has type Y”, say “X is a Y”

```
mfg:Product1 rdf:type mfg:Product.  
lit:Shakespeare rdf:type lit:Playwright.
```

Turtle

- `rdf:type` is often abbreviated by `a`
 - Rather than say “X has type Y”, say “X is a Y”

```
mfg:Product1 rdf:type mfg:Product.  
lit:Shakespeare rdf:type lit:Playwright.
```

```
mfg:Product1 a mfg:Product.  
lit:Shakespeare a lit:Playwright.
```

Turtle

Blank Nodes (“bnodes”)

Turtle

Blank Nodes (“bnodes”)

- What if we don't know the subject of a triple?

Turtle

Blank Nodes (“bnodes”)

- What if we don't know the subject of a triple?
 - e.g. existential quantifier in FOL

Turtle

Blank Nodes (“bnodes”)

- What if we don't know the subject of a triple?
 - e.g. existential quantifier in FOL
- We can always create a “dummy” identifier

Turtle

Blank Nodes (“bnodes”)

- What if we don't know the subject of a triple?
 - e.g. existential quantifier in FOL
- We can always create a “dummy” identifier
- Alternatively, we can express with bracket notation:

Turtle

Blank Nodes (“bnodes”)

- What if we don't know the subject of a triple?
 - e.g. existential quantifier in FOL
 - We can always create a “dummy” identifier
 - Alternatively, we can express with bracket notation:

lit:Sonnet78 lit:hasInspiration [a :Woman;
bio:livedIn geo:England] .

Turtle

Blank Nodes (“bnodes”)

- What if we don't know the subject of a triple?
 - e.g. existential quantifier in FOL
- We can always create a “dummy” identifier
- Alternatively, we can express with bracket notation:

```
lit:Sonnet78 lit:hasInspiration [a :Woman;  
                                bio:livedIn geo:England] .
```

- Due to lack of a UID, need to copy the whole bracketed statement

Summary

- RDF expresses data as triples
- Two elements have the same identity if they share the same URI
- RDF triples can be viewed as a graph
- To merge data, simply add up all triples from all sources
 - No need to worry about missing entries or matching up columns from tables

Key Concepts

RDF (Resource Description Framework)—This distributes data on the Web.

Triple—The fundamental data structure of RDF. A triple is made up of a subject, predicate, and object.

Graph—A nodes-and-links structural view of RDF data.

Merging—The process of treating two graphs as if they were one.

URI (Uniform Resource Indicator)—A generalization of the URL (Uniform Resource Locator), which is the global name on the Web.

namespace—A set of names that belongs to a single authority. Namespaces allow different agents to use the same word in different ways.

qname—An abbreviated version of a URI, it is made up of a namespace identifier and a name, separated by a colon.

rdf:type—The relationship between an instance and its type.

rdf:Property—The type of any property in RDF.

Reification—The practice of making a statement about another statement. It is done in RDF using `rdf:subject`, `rdf:predicate`, and `rdf:object`.

N-Triples, Turtle, RDF/XML—The serialization syntaxes for RDF.

Blank nodes—RDF nodes that have no URI and thus cannot be referenced globally. They are used to stand in for anonymous entities.