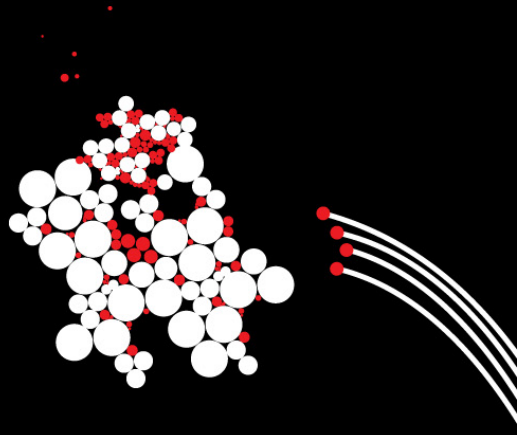
UNIVERSITY OF TWENTE.

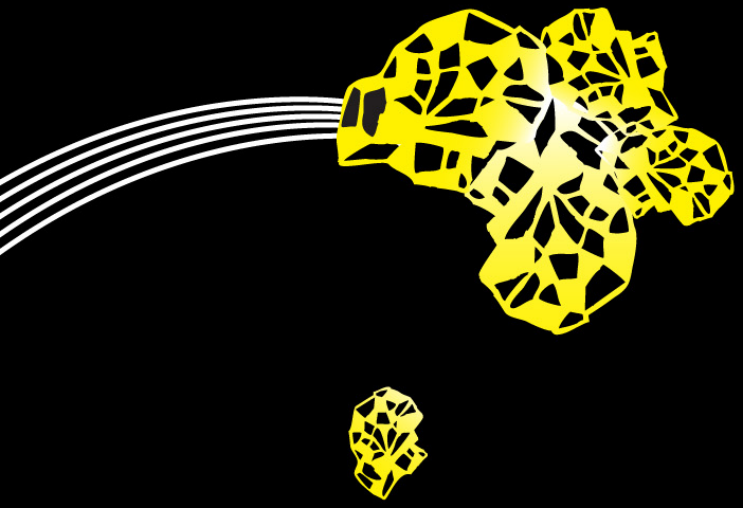


Data Science

Topic SEMI

Semi-Structured Data

MAURICE VAN KEULEN

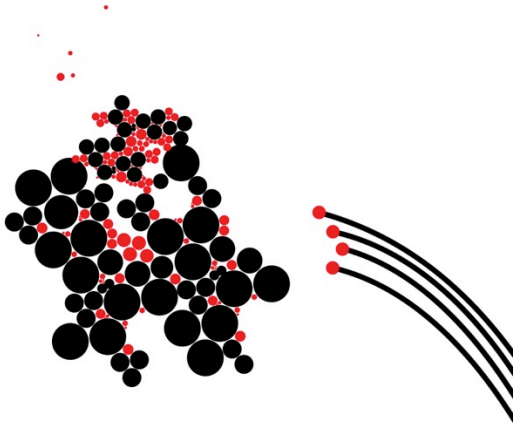


AGENDA

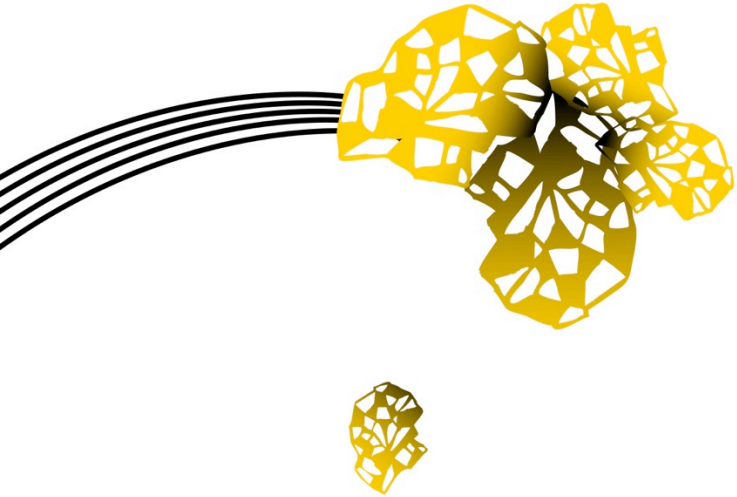
- History of web standards
- XML / JSON
  - Relational to XML with SQL/XML
  - XML Databases (XPath / XQuery / JSONiq)

- What is (the purpose) the Semantic Web
  - Integrating data on the web
  - Linked Open Data (LOD)
  - Data : RDF; and querying : SPARQL

UNIVERSITY OF TWENTE.



HISTORY OF WEB STANDARDS



it's 1967…

…Charles Goldfarb goes to work

IBM Almaden Research Center

The picture can't be displayed.

(Law) office automation

- Separate programs should interoperate
  - text editor,
  - document retrieval (IBM STAIRS),
  - presentation software,
  - etc.

Specific coding

```
.roman16
.ce
.bf
Introduction
.roman12
.ju
.ll 39
.ss
The work of authors,
publishers, and
researchers involves,
in varying degrees,
three recognizable
```

Formatting commands:

.roman16 use font size 16
.ce center next line
.ju justify right margin
.ll line length
.ss single spacing

Generic coding: GML (1969)

- Generic coding:
  - use "**`heading`**" instead of "**`.bf.ce.roman16`**"
- Generalized Markup Language (GML)
  - Charles Goldfarb
  - Edward Mosher, and
  - Raymond Lorie

Raymond Lorie …

…went on to develop System-R!

Standardized Generalized Markup Language (SGML)

- International standard ISO 8879 (1986)
  - Markup should describe a document's structure rather than its physical characteristics, and
  - Markup should be rigorous so that it can be unambiguously understood by a program *or* a human interpreter.

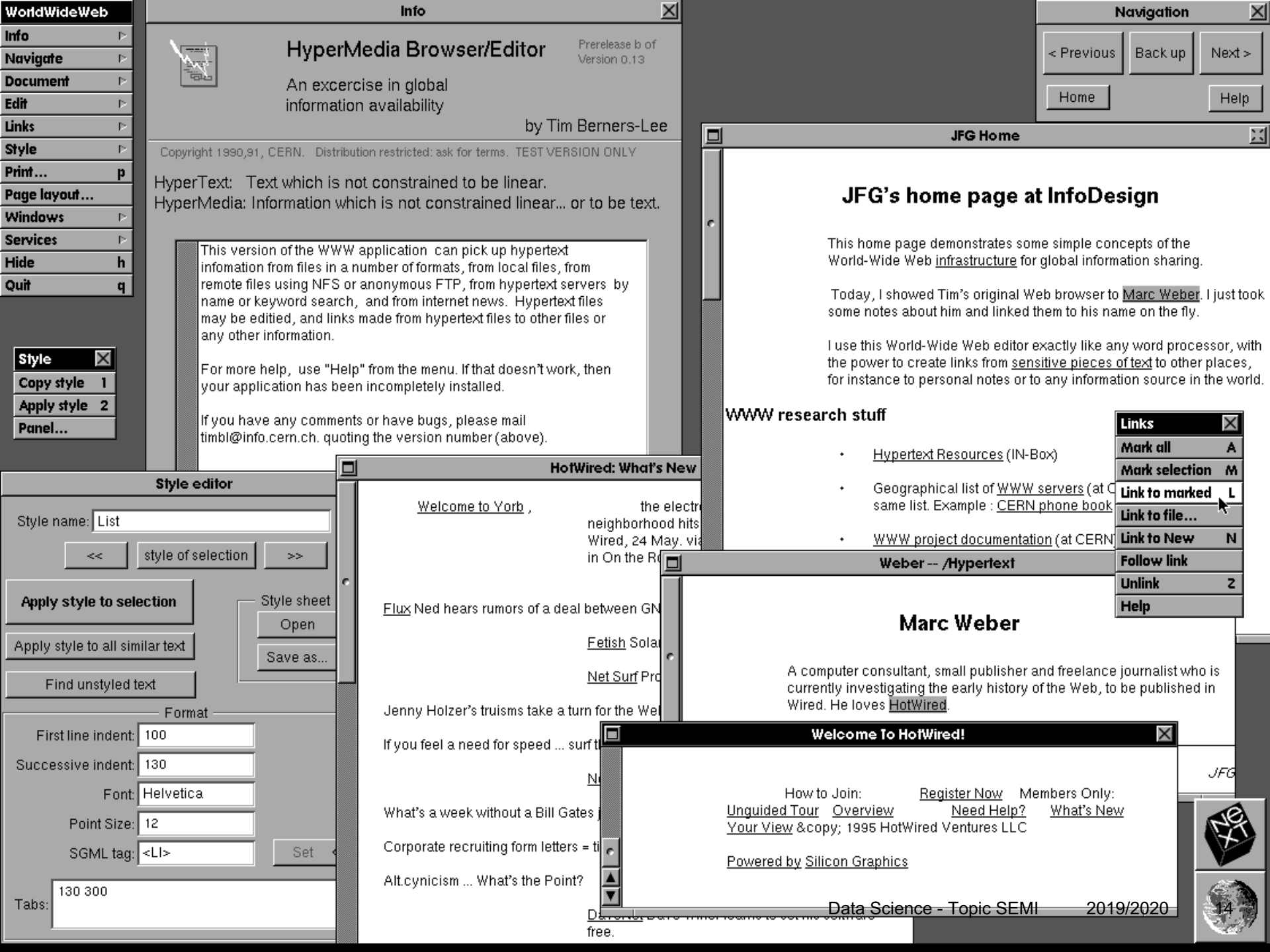
… it's 1989…

…Tim Berners-Lee goes to work

The picture can't be displayed.

Berners-Lee proposes the WWW

- HTML based on SGML
- Communication via anonymous ftp
- hyper text
- editing


WorldWideWeb
Info
Navigate
Document
Edit
Links
Style
Print... p
Page layout...
Windows
Services
Hide h
Quit q
Info
HyperMedia Browser/Editor
Prerelease b of Version 0.13
An excercise in global information availability
by Tim Berners-Lee
Copyright 1990,91, CERN. Distribution restricted: ask for terms. TEST VERSION ONLY
HyperText: Text which is not constrained to be linear.
HyperMedia: Information which is not constrained linear... or to be text.
This version of the WWW application can pick up hypertext infomation from files in a number of formats, from local files, from remote files using NFS or anonymous FTP, from hypertext servers by name or keyword search, and from internet news. Hypertext files may be editied, and links made from hypertext files to other files or any other information.
For more help, use "Help" from the menu. If that doesn't work, then your application has been incompletely installed.
If you have any comments or have bugs, please mail timbl@info.cern.ch. quoting the version number (above).
Navigation
< Previous
Back up
Next >
Home
Help
JFG Home
JFG's home page at InfoDesign
This home page demonstrates some simple concepts of the World-Wide Web infrastructure for global information sharing.
Today, I showed Tim's original Web browser to Marc Weber. I just took some notes about him and linked them to his name on the fly.
I use this World-Wide Web editor exactly like any word processor, with the power to create links from sensitive pieces of text to other places, for instance to personal notes or to any information source in the world.
WWW research stuff
Hypertext Resources (IN-Box)
Geographical list of WWW servers (at C
same list. Example : CERN phone book
WWW project documentation (at CERN
Style
Copy style 1
Apply style 2
Panel...
Links
Mark all A
Mark selection M
Link to marked L
Link to file...
Link to New N
Follow link
Unlink Z
Help
Style editor
Style name: List
<<
style of selection
>>
Apply style to selection
Style sheet
Open
Save as...
Apply style to all similar text
Find unstyled text
Format
First line indent: 100
Successive indent: 130
Font: Helvetica
Point Size: 12
SGML tag: <LI>
Set
Tabs: 130 300
HotWired: What's New
Welcome to Yorb ,
the electr
neighborhood hits
Wired, 24 May. via
in On the R
Flux Ned hears rumors of a deal between GN
Fetish Sola
Net Surf Pr
Jenny Holzer's truisms take a turn for the Wel
If you feel a need for speed ... surf t
N
What's a week without a Bill Gates
Corporate recruiting form letters = t
Alt.cynicism ... What's the Point?
free.
Weber -- /Hypertext
Marc Weber
A computer consultant, small publisher and freelance journalist who is currently investigating the early history of the Web, to be published in Wired. He loves HotWired.
Welcome To HotWired!
How to Join:
Register Now
Members Only:
Unguided Tour
Overview
Need Help?
What's New
Your View
© 1995 HotWired Ventures LLC
Powered by Silicon Graphics
JFG

HTML

- 1992: HTML 1.0 'standard' based on SGML, first web browsers
- 1994: Berners-Lee, founds the World Wide Web Consortium (W3C)

(and Bill Gates confesses that *"the Internet mania had taken him by surprise."*)

The picture can't be displayed.

the birth of XML… 1998

- 1996: Extensible Markup Language (XML)
  - simple dialect of SGML
  - easy implementation
  - interoperability with both SGML and HTML

- 1998: XML 1.0 "W3C recommendation"

now, it's 2017…

http://en.wikipedia.org/wiki/Category:XML-based_standards

What is XML today?

- A syntax for document publishing (Web data, Journals, Newspapers)
- A syntax for data-exchange and enterprise application integration (B2B applications, e-commerce, web services)
- Many standards (NewsML, MathML, XForms, SVG, GML, RSS, Atom, SOAP, etc.)
- Many generic tools and programming libraries

- A standard for a new *data model* (XML databases, semi-structured data, hierarchical model)

XML DATABASE PRODUCTS

Microsoft SQL Server™
The enterprise relational database management and analysis system

xyleme

TIMBER

software AG
THE XML COMPANY

xindice
(ZEEN-DEE-CHAY)

X-HIVE

BASE X

XML:DB

DB2 Information Management Software

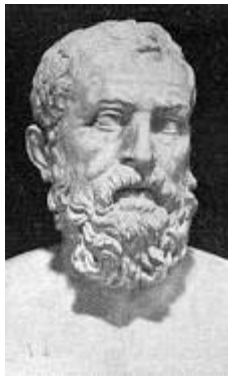
ORACLE
BERKELEY DB
XML Edition
ORACLE

Pathfinder
MonetDB

eXist

XHTML: Draconian error handling

- XHTML (minus mime-type loophole) is not backwards compatible with HTML.
- Draconian error handling for all content served with XHTML MIME type.
- XForms needs HTML MIME type and is not backwards compatible with HTML forms.



Draco (Δράκων, circa 7th century BC) was legislator of Athens in Ancient Greece.

The picture can't be displayed.

October 2009: HTML5 is not XML

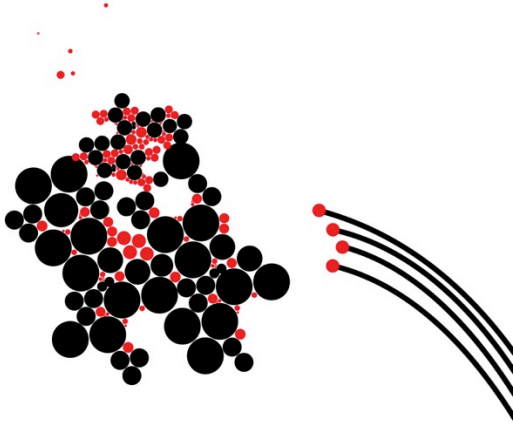
- W3C stops further development of XHTML

While we recognize the value of the XHTML 2 Working Group's contributions over the years, after discussion with the participants, W3C management has decided to allow the Working Group's charter to expire at the end of 2009 and not to renew it.

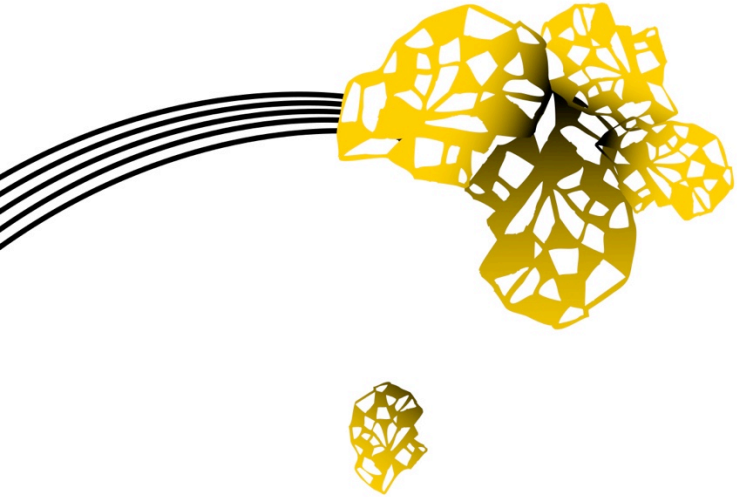
2020: R.I.P. XML ?

- JSON
  - Javascript Object notation
  - Increasingly used in HTML5 responsive web apps (ironically using the javascript function XMLhttpRequest())

UNIVERSITY OF TWENTE.



XML



EXAMPLE XML DOCUMENT

```
<company>
  <name>Foobar Corporation</name>
  <ticker_symbol>FOO</ticker_symbol>
  <description>Foobar Corporation is a maker of <EM>Foo(TM)</EM>
  and Foobar(TM) products and a leading software company with a 300
   Billion dollar revenue in 1999. It is located in Alaska.
  </description>
  <business_code>Software</business_code>
  <partners>
    <partner>YouNameItWeIntegrateIt.com</partner>
    <partner>TheAppCompany Inc.</partner>
  </partners>
  <competitors>
    <competitor name="Gorilla Corporation"/>
  </competitors>
</company>
```

Opening tag

Closing tag

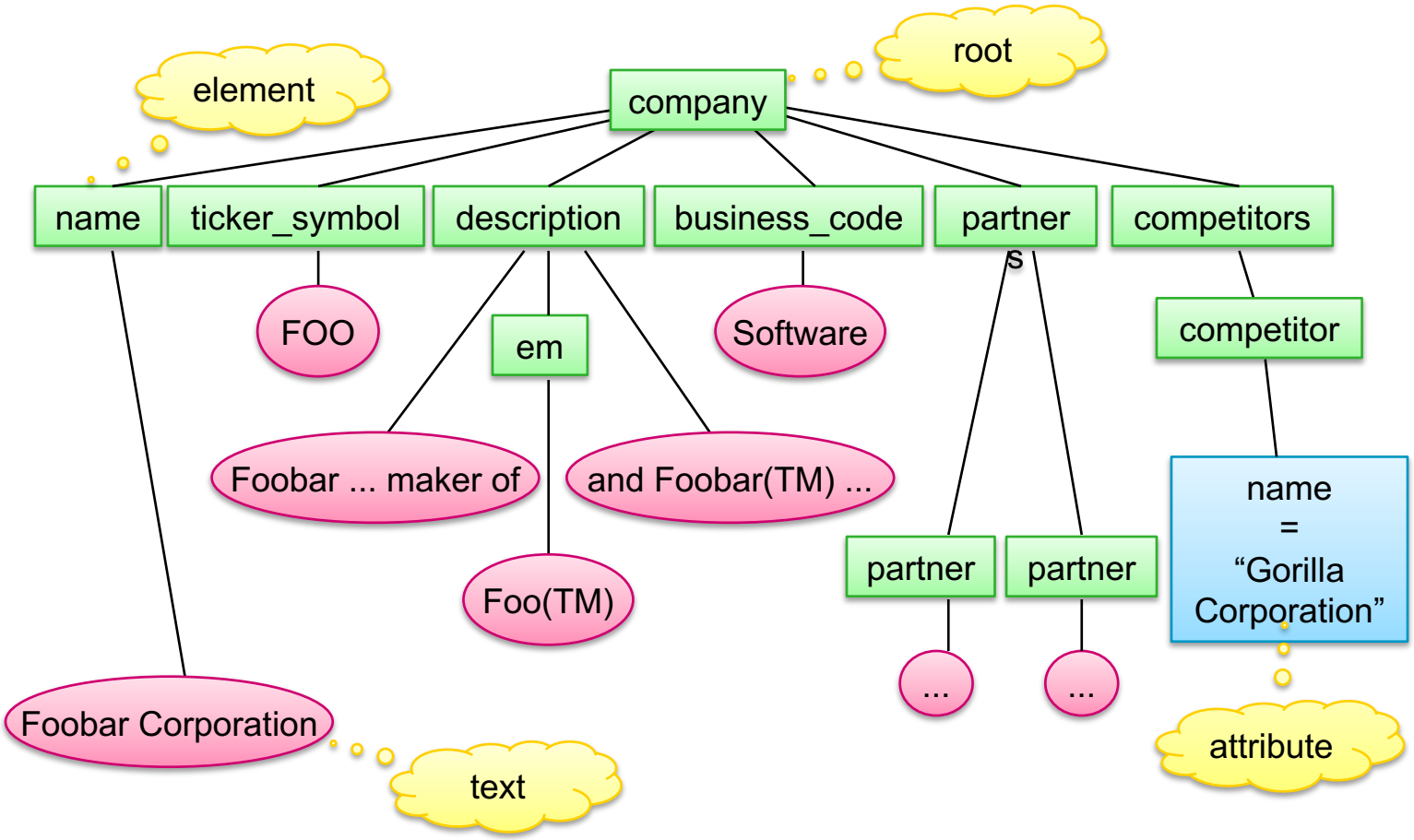
Mixed content

Shorthand for no embedded elements or text

Attribute

- Markup tags are chosen purely for **logic structure**!
- **Well-formedness**: start and end tags need to match precisely and be properly nested

XML DOCUMENTS ARE TREES

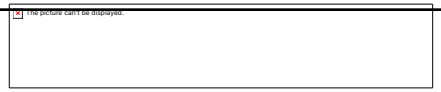


DATA- VS. DOCUMENT-CENTRIC XML

- Data-centric
  - Very structured
  - Use stems from data exchange from databases
- Document-centric
  - Semi-structured
  - Mixed content
  - Fulltext search important
  - Use stems from exchange of (formatted) documents

```
<site>
  <item ID="I001">
    <name>Chair</name>
    <description>This chair is in good
      condition ...
    </description>
  </item>
  <item ID="I002">
    <name>Table</table>
    <description>...</description>
  </item>
  ...
</site>
```

```
<memo>
  <author>John Doe</author>
  <title>...</title>
  <body>
  This memo is meant for all persons
  responsible for
  <list bullets=1>
    <item>either <em>customers</em> abroad,
    <item>or <em>suppliers</em> abroad.
  </list>
  ...
</memo>
```

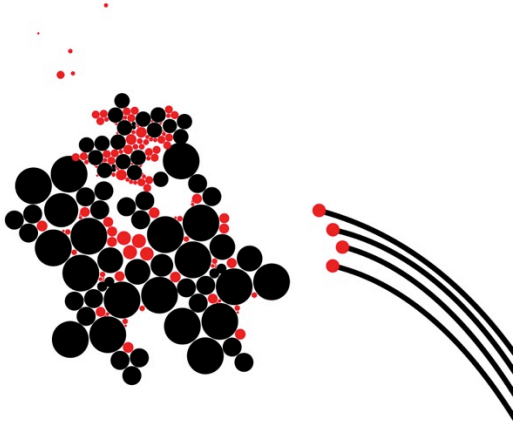


DTD: EXAMPLE

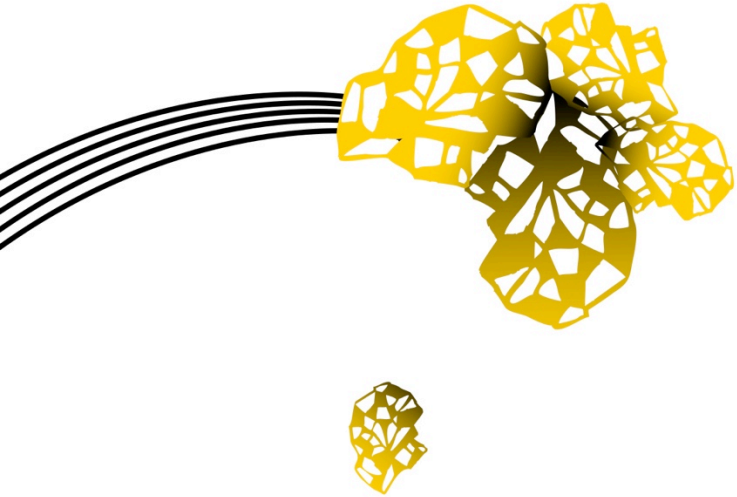
```
<!DOCTYPE bookstore [
  <!ELEMENT bookstore (book)*>
  <!ELEMENT book (title, author+, price?)>
  <!ATTLIST book genre CDATA #REQUIRED>
  <!ELEMENT title (#PCDATA)>
  <!ELEMENT author ( name | (first-name, last-name))>
  <!ELEMENT price (#PCDATA)>
  <!ELEMENT name (#PCDATA)>
  <!ELEMENT first-name (#PCDATA)>
  <!ELEMENT last-name (#PCDATA)>
]>

<bookstore>
  <book genre="autobiography">
    <title>The Autobiography of Benjamin Franklin</title>
    <author> <first-name>Benjamin</first-name>
             <last-name>Franklin</last-name> </author>
    <price>8.99</price>
  </book>
  <book genre="novel"> ... </book>
</bookstore>
```

UNIVERSITY OF TWENTE.



JSON



WHAT IS JSON

- JSON: **J**ava**S**cript **O**bject **N**otation
- JSON is a syntax for storing and exchanging data
- JSON is lightweight (compared to XML)
- JSON is text, written with JavaScript object notation, but language independent nonetheless
- JSON is "self-describing"

JSON DATA MODEL

key

value

- Curly braces hold **objects**
  {“name” : “Maurice”,
  “room” : “ZI-2013” }
- Square brackets hold **arrays**
  [1,2,3]
- Can be **arbitrarily nested**
  {“name” : “Maurice”,
  “room” : “ZI-2013”,
  “phone” : [{ ”nr” : “053-4893688”, “kind” : “work” },
  { “nr” : “…”, “kind” : “home” }]
  }
- Values must be one of the following **data types**: string, number, object (JSON object), array, boolean, or null

JSON EXAMPLE:
A TWEET (SIMPLIFIED) FROM THE TWITTER API

object

{"created_at":"Tue Feb 19 18:42:07 +0000 2013",
"id":303937526471725056,
"text":"Batavierenrace door #Ulft: Eind april klinkt jaarlijks het startschot voor de grootste estafetteloop v... http:\/\/t.co\/hijGD3pk #Enschede",
"user":{"id":258430204,
"name":"Enschede Nieuws",
"screen_name":"nieuws_enschede",
"description":"Al het nieuws over Enschede",
"followers_count":1344, "friends_count":17, "listed_count":18, …},
"retweet_count":0,
"entities":{"hashtags":[{"text":"Ulft", "indices":[20,25]},
{"text":"Enschede","indices":[127,136]}],
"urls":[{"url":"http:\/\/t.co\/hijGD3pk",
"expanded_url":"http:\/\/bit.ly\/UE0MCq",
"display_url":"bit.ly\/UE0MCq","indices":[106,126]}],
"user_mentions":[]},
"retweeted":false,
"possibly_sensitive":false}

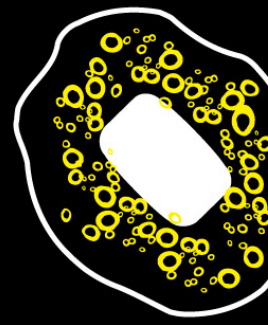
array

SOME GENERAL REMARKS ABOUT JSON

- JSON data is also a tree!
- No standardized query language … although
  - Relational databases have been extended with JSON functionality
  - XML databases / XQuery have been extended to also work for JSON
  - There are XQuery-inspired query languages
- There are proposals for schema languages for JSON to facilitate validation

➢ But be aware: only very limited standardization; developed by community not by committees

UNIVERSITEIT TWENTE.



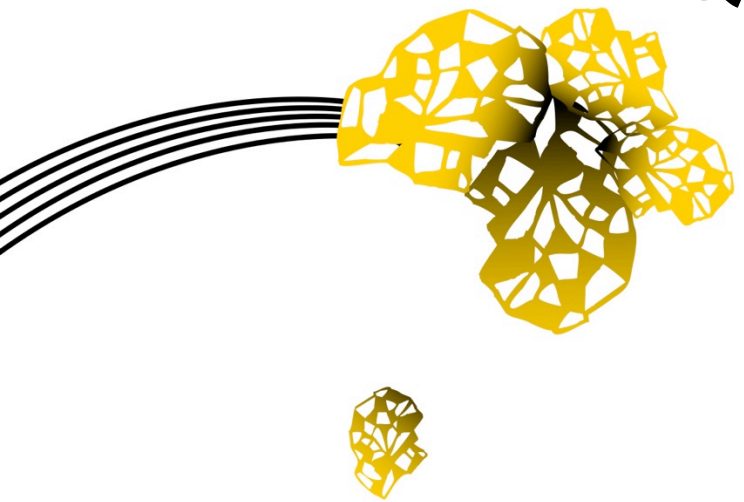
RELATIONAL TO XML WITH SQL/XML

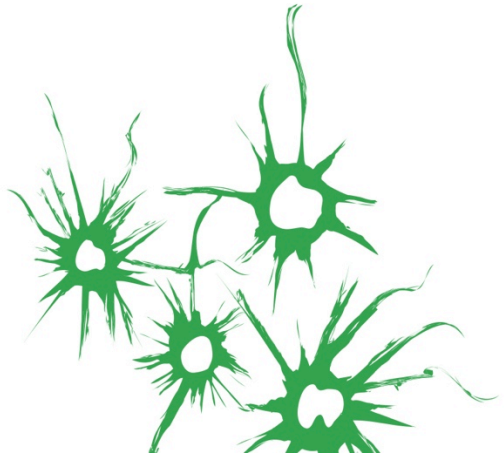
(ASSIGNMENT 2)

WHAT IS SQL/XML

- Extension to SQL for producing XML fragments as query result
  - Part of SQL 2003 standard
- Constructor functions generate XML
  - XMLELEMENT
  - XMLATTRIBUTES
  - XMLAGG
  - XMLFOREST
  - etc.

UNIVERSITY OF TWENTE.



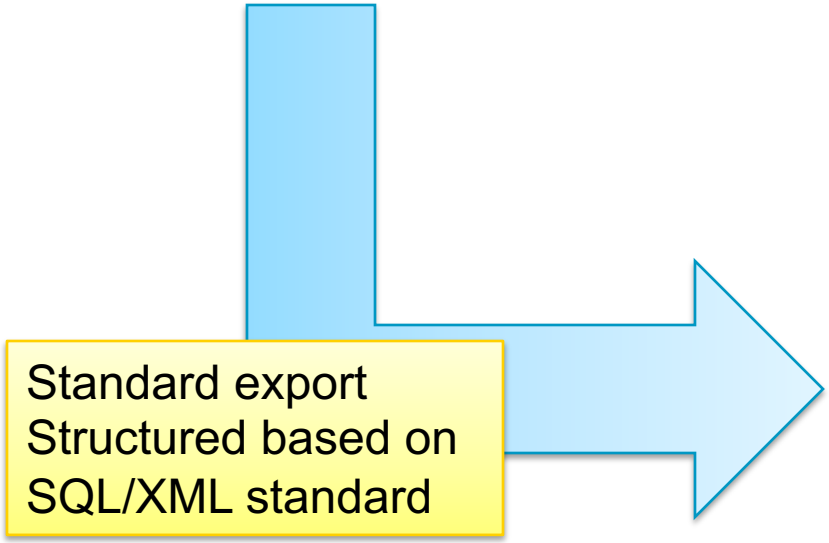


(1) STANDARD EXPORT + TRANSFORMATION IN XQUERY

STANDARD EXPORT

DEFAULT VIEW OR CANNONICAL MAPPING

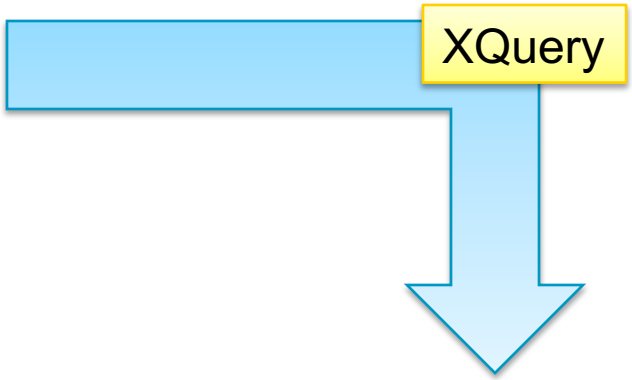
id	fname	lname	group
1	Klaas	Sikkel	SCS
2	Djoerd	Hiemstra	DB
3	Luis	Ferreira Perez	SCS
4	Maurice	van Keulen	DB




```
<database>
 <teachers>
  <row>
   <id>1</id>
   <fname>Klaas</fname>
   <lname>Sikkel</lname>
   <group>SCS</group>
  </row>
  <row>
   <id>2</id>
   <fname>Djoerd</fname>
   <lname>Hiemstra</lname>
   <group>DB</group>
  </row>
  …
 </teachers>
 …
</database>
```

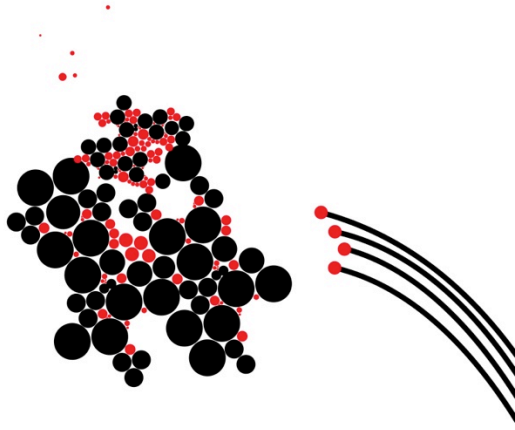
TRANSFORM STANDARD EXPORT WITH XQUERY

```
<database>
  <teachers>
    <row>
      <id>1</id>
      <fname>Klaas</fname>
      <lname>Sikkel</lname>
      <group>SCS</group>
    </row>
    <row>
      <id>2</id>
      <fname>Djoerd</fname>
      <lname>Hiemstra</lname>
      <group>DB</group>
    </row>
    …
  </teachers>
  …
</database>
```

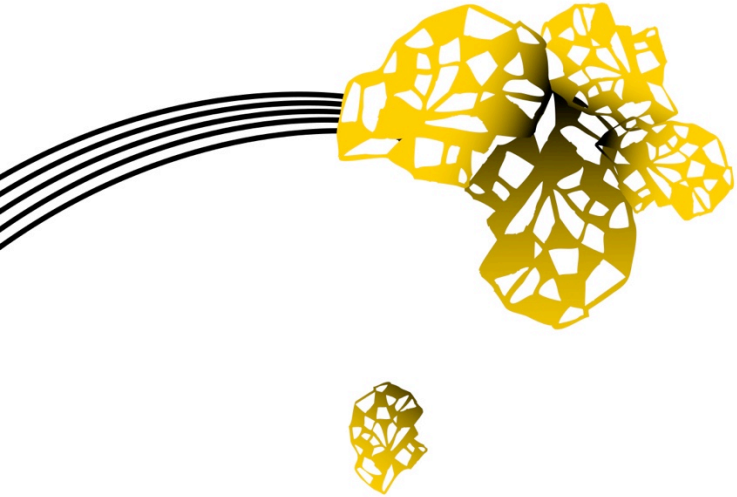



```
<result>
  <lecturers chair="DB">
    <name>Hiemstra</name>
    <name>van Keulen</name>
  </lecturers>
  <lecturers chair="SCS">
    <name>Sikkel</name>
    <name>Ferreira Perez</name>
  </lecturers>
</result>
```

UNIVERSITY OF TWENTE.



(2) CUSTOM MAPPINGS



SQL (NOT XML)

id	fname	lname	group
1	Klaas	Sikkel	SCS
2	Djoerd	Hiemstra	DB
3	Luis	Ferreira Perez	SCS
4	Maurice	van Keulen	DB

```
SELECT id, fname || ' ' || lname AS result
FROM teachers
```

id	result
1	Klaas Sikkel
2	Djoerd Hiemstra
3	Luis Ferreira Perez
4	Maurice van Keulen

SQL/XML: CONSTRUCTING ELEMENTS

id	fname	lname	group
1	Klaas	Sikkel	SCS
2	Djoerd	Hiemstra	DB
3	Luis	Ferreira Perez	SCS
4	Maurice	van Keulen	DB

```
SELECT id,
  XMLELEMENT(NAME lecturers,
    fname || ' ' || lname) AS result
FROM teachers
```

id	result
1	<lecturers>Klaas Sikkel</lecturers>
2	<lecturers>Djoerd Hiemstra </lecturers>
3	<lecturers>Luis Ferreira Perez </lecturers>
4	<lecturers>Maurice van Keulen </lecturers>

SQL/XML: CONSTRUCTING ATTRIBUTES

id	fname	lname	group
1	Klaas	Sikkel	SCS
2	Djoerd	Hiemstra	DB
3	Luis	Ferreira Perez	SCS
4	Maurice	van Keulen	DB

```
SELECT id,
  XMLELEMENT(NAME lecturers,
    XMLATTRIBUTES(id,lname AS name)
  AS result
FROM teachers
```

id	result
1	<lecturers id="1" name="Sikkel"/>
2	<lecturers id="2" name="Hiemstra"/>
3	<lecturers id="3" name="Ferreira Perez"/>
4	<lecturers id="4" name="van Keulen"/>

SQL/XML: CONSTRUCTING NESTED ELEMENTS

id	fname	lname	group
1	Klaas	Sikkel	SCS
2	Djoerd	Hiemstra	DB
3	Luis	Ferreira Perez	SCS
4	Maurice	van Keulen	DB

```
SELECT id,
  XMLELEMENT(NAME lecturers, 'Teacher '
    XMLELEMENT(NAME name, lname))
  AS result
FROM teachers
```

id	result
1	<lecturers>Teacher <name>Sikkel</name></lecturers>
2	<lecturers>Teacher <name>Hiemstra</name></lecturers>
3	<lecturers>Teacher <name>Ferreira Perez</name></lecturers>
4	<lecturers>Teacher <name>van Keulen</name></lecturers>

SQL/XML: XMLFOREST

id	fname	lname	group
1	Klaas	Sikkel	SCS
2	Djoerd	Hiemstra	DB
3	Luis	Ferreira Perez	SCS
4	Maurice	van Keulen	DB

```
SELECT id,
   XMLFOREST(lname AS lecturers,
      group AS chair)
   AS result
FROM teachers
```

id	result
1	<lecturers>Sikkel</lecturers><chair>SCS</chair>
2	<lecturers>Hiemstra</lecturers><chair>DB</chair>
3	<lecturers>Ferreira Perez</lecturers><chair>SCS</chair>
4	<lecturers>van Keulen</lecturers><chair>DB</chair>

SQL/XML: XMLAGG

id	fname	lname	group
1	Klaas	Sikkel	SCS
2	Djoerd	Hiemstra	DB
3	Luis	Ferreira Perez	SCS
4	Maurice	van Keulen	DB

```
SELECT group,
  XMLELEMENT(NAME lecturers,
    XMLATTRIBUTES(group AS chair),
    XMLAGG(
      XMLELEMENT(NAME name,lname)
    ))
  AS result
FROM teachers
GROUP BY group
```

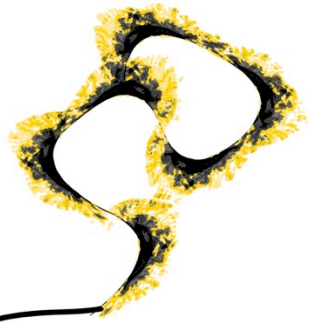
group	result
DB	<lecturers chair="DB"><name>Hiemstra</name> <name>van Keulen</name></lecturers>
SCS	<lecturers chair="SCS"><name>Sikkel</name> <name>Ferreira Perez</name></lecturers>

SQL → JSON

- There are similar operators in major DBMS products
- Not standardized
- Not as well supported as XML yet
  … but already quite impressive what the newest versions of PostgreSQL can do

UNIVERSITY OF TWENTE.

XML DATABASES

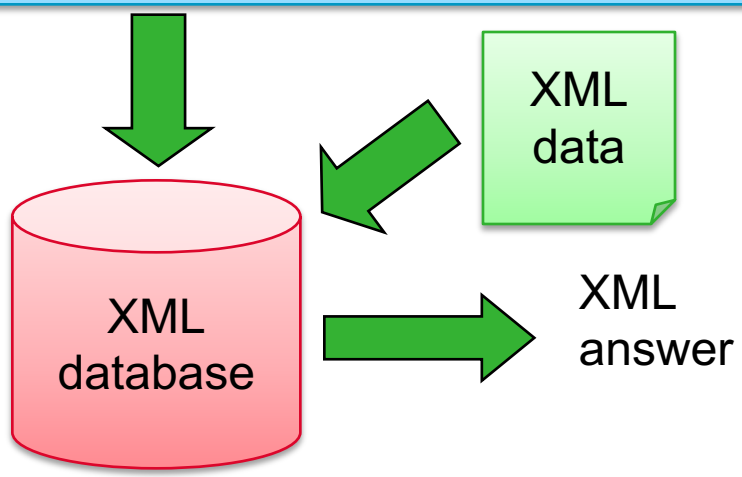


XML DATABASES

XQuery is for the XML world, what SQL is for the relational world

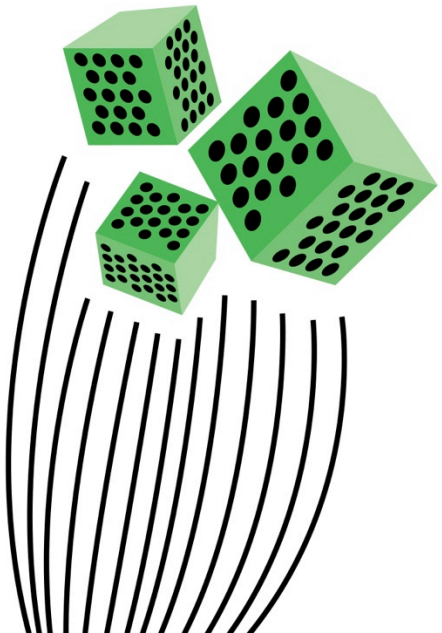
XQuery query

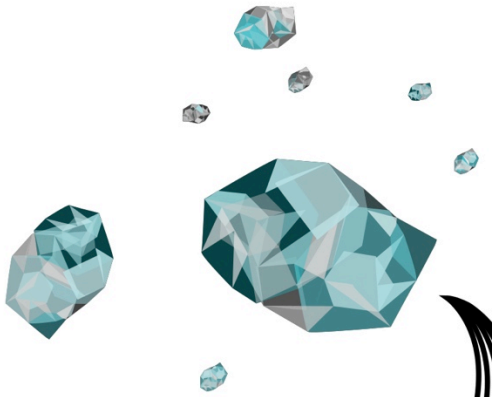
```
for $p in fn:doc("auction.xml")/site/people/person
return
  let $a :=
    for $t in fn:doc("auction.xml")/site/closed_auctions/closed_auction
    return
      if fn:data($t/buyer/person/text()) = fn:data($p/id/text())
      then $t
      else ()
  return <item> { <person> { $p/name/text() } </person>,
                  text { fn:count($a) } } </item>
```



UNIVERSITY OF TWENTE.

XPATH





XPATH BASICS

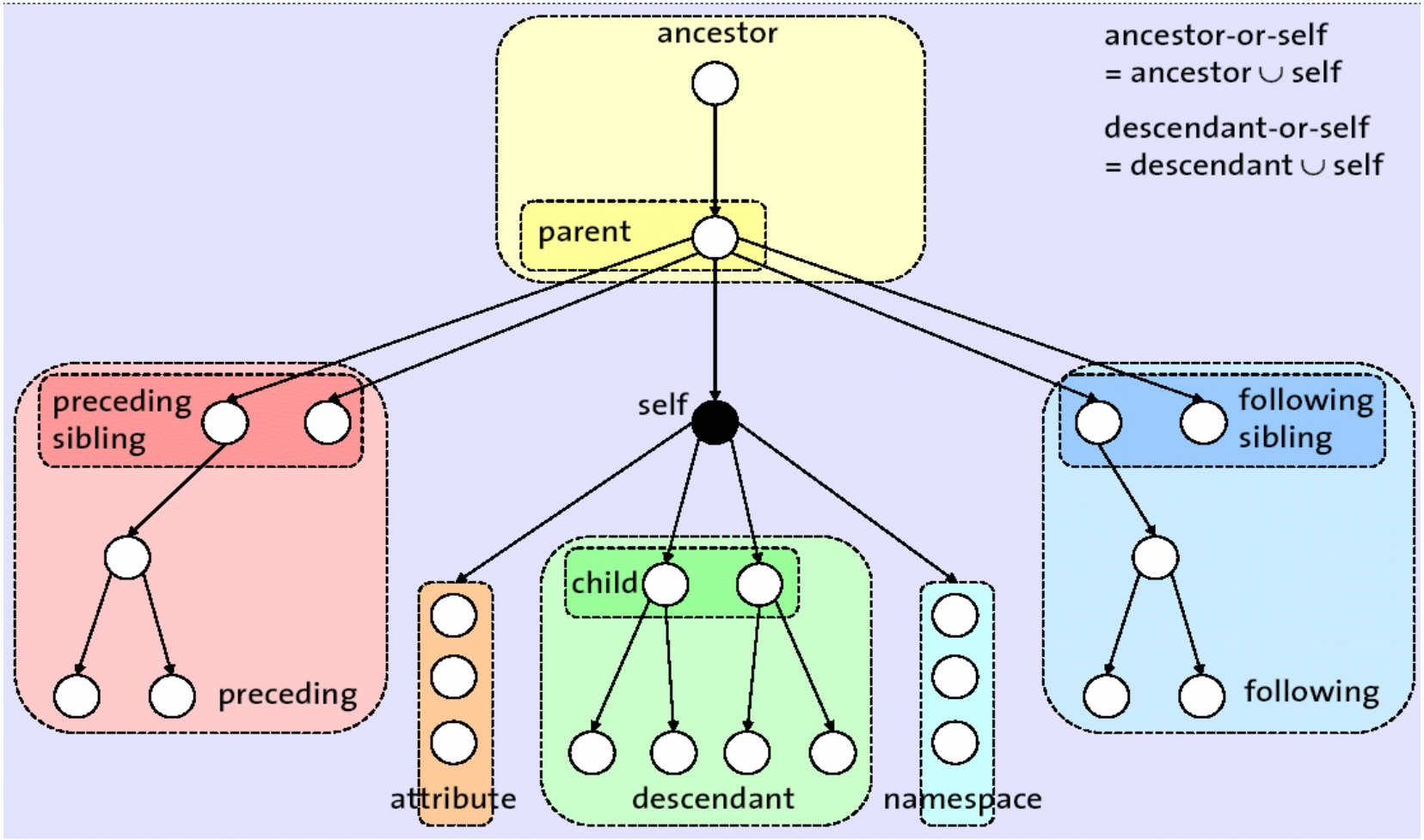
- Central concept is the *path expression*:
  - A sequence of location steps seperated by ‘/’
    /step/step/.../step
  - Query evaluatie from left to right
- A location step has the following form
  axis::nodetest[predicate]
  - A location step starts from a *context node*.
  - axis: parent, child, descendant, following, etc.
  - nodetest: or a name test (on the tag)
    or a kind test (attribute of text node).

XPATH EXAMPLES

```
<bookstore>
  <book genre="autobiography">
    <title>The Autobiography of Benjamin Franklin
    </title>
    <author> <first-name>Benjamin</first-name>
             <last-name>Franklin</last-name>
    </author>
    <price>8.99</price>
  </book>
  <book genre="novel"> ... </book>
</bookstore>
```

- All authors of all books: `/bookstore/book/author`
- All author elements wherever: `//author`
- All books written by Franklin:
  `//book[./author/last-name = "Franklin"]`
- First author of every book: `/bookstore/book/author[1]`
- The prices of all novels: `//book[@genre = "novel"]/price`

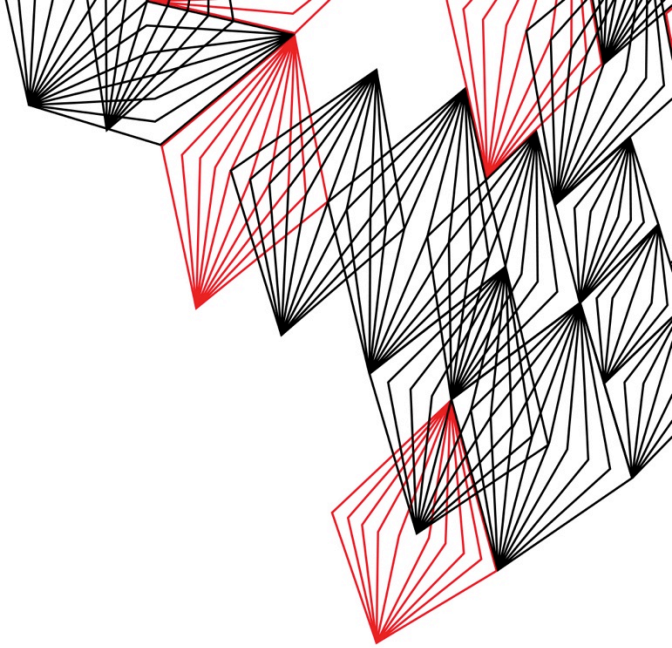
XPATH AXES


ancestor
ancestor-or-self
= ancestor ∪ self
descendant-or-self
= descendant ∪ self
parent
preceding
sibling
self
following
sibling
child
preceding
following
attribute
descendant
namespace

XPATH: A FEW DETAILS

- `//book[@genre = "novel"]/price`
  is shorthand for
  `/child::*/descendant-or-self::book[./attribute::genre = "novel"]/child::price`
- Select by position: `//author[./position()<3]`
- Existential: `//book[./price]`
  Note the difference with `//book[/price]`
- Text nodes: `//book/title/child::text()`
- Casting: `//book[number(./price) < number('8.50')]`
- Counting: `count(//book)`
- Order
  - The result of every location step is in *document order* and *without duplicates*
  - But positions count backwards for reverse axes

UNIVERSITY OF TWENTE.



XQUERY

XQUERY

- Full query language (comparable with PLSQL)
- XPath is a sublanguage
  - An XPath-expression is a valid XQuery query!
- Base form FLWOR (pronounce ‘flower’)
  - FOR/LET/WHERE/ORDER BY/RETURN
- Example:

```
<result>{
let $doc := doc("books.xml")
for $b in $doc/book
where $b/author
order by $b/title ascending
return <b>{$b/title}
          <nrofauthors>{count($b/author)}</nrofauthors>
        </b>
}</result>
```

XQUERY EXPLAINED

```
<result>{
  let $doc := doc("books.xml")
  for $b in $doc/book
  where $b/author
  order by $b/title ascending
  return <b>{$b/title}
            <nrofauthors>{count($b/author)}</nrofauthors>
         </b>
  }</result>
```

- LET produces a binding for variable $doc
- FOR iterates using a binding over an ordered list of book elements
- WHERE filters this list to only $b's for which there is a child element named author
- ORDER BY sorts the list on title value
- RETURN constructs for every $b a resulting XML fragment
- All fragments are collected in a resulting sequence
- <result> constructs an XML element. Content within tags is take literally. {…} is necessary to fill in (sub)query results

XQUERY: BIGGER EXAMPLE

```
for $d in doc("depts.xml")//deptno
let $e := doc("emps.xml")//employee[deptno = $d]
where count($e) >= 2
order by avg($e/salary) descending
return
  <big-dept>
     { $d,
        <headcount>{count($e)}</headcount>,
        <avgsal>{avg($e/salary)}</avgsal>
     }
  </big-dept>
```

CONDITIONAL EXPRESSIONS

```
<r>{
  for $a in doc("/XQuery/docs/xmach1/d2.xml")//section1
  return
    <s> {$a/@id}
        { if (count($a/paragraph)>2)
          then <bigone>{$a/head1}</bigone>
          else <smallone/>
        }
    </s>
}</r>
```

GROUP BY IN XQUERY

```
<answer>{
     for $emp in doc("payroll.xml")//employee
     let $a := $emp/age
     group by $a
     return
       <age-group age="{$a}">
         <cnt>{ count($emp) }</cnt>
         { for $e in $emp return $e/name }
       </age-group>
}</answer>
```

Group by changes loop variabele ($emp)
in a sequence of elements
(all with the same $a)

FLWOR EXPLAINED: TUPLE STREAMS

`for $emp in doc("payroll.xml")//employee`

- *Produces a stream of variable bindings*
  ($emp=<employee>…</employee>) ,
  ($emp=…) ,
  ($emp=…) , …

`let $a := $emp/age`

- *Adds a new binding to a tuple stream*
  ($emp=<employee>…</employee>, $a=<age>…</age>) ,
  ($emp=…) ,
  ($emp=…) , …

`group by $a`

- ($emp=(…), $a=<age>…</age>), …

etc.

FUNCTIONS

```
declare namespace foo = "http://foo-bar.org";

declare function foo:age-groups($emps as element()*)
   as element(age-group)*
{
  <answer>{
      for $emp in $emps
      let $a := $emp/age
      group by $a
      return
        <age-group age="{$a}">
          <cnt>{ count($emp) }</cnt>
          { for $e in $emp return $e/name }
        </age-group>
  }</answer>
};

foo:age-groups(doc("payroll.xml")//employee)
```

MODULES

Declaration of module (file test.xqm)

```
module namespace foo = “http://foo.org/test”;

declare function foo:func ...
```

Use of module

```
import module
   namespace bar = http://foo.org/test
   at “...test.xqm”;

... bar:func() ...
```

EVERYTHING IS A SEQUENCE!

- Sequence: `(1, 5, 8, 2, 2, 5)`
- Sequences are flat: above is equivalent with `( (1,5), (), (8,(2,2)), (5) )`
- Single value is equivalent with sequence with one element: `4 ≡ (4)`
- Empty sequence: `()`
  Often used as NULL, or as false
  `() + 4` is `();` `() and true` is `();` `() and false` is `false`
- Sequence can contain a mix of things: `(1, "foo", $b, $b/@id)`
- Order matters!
  - XPath expression result is in document order
  - FOR expression result is in sequence order
- String value of een sequence: `string(( (1,5),(),(8,(2,2)),(5)))` is `"1 5 8 2 2 5"`

XQUERY 3.0 / 3.1

‘New’ W3C recommendation: XQuery 3.0 (8 April 2014)
[candidate recommendation: XQuery 3.1 (18 Dec 2014)]

- `group by` clause, `count` clause
- tumbling window en sliding window in FLWOR
- `allowing empty` in FOR (outer join functionaliteit)
- `try/catch`
- Dynamic function calls
- Inline function expressions
- Private functions
- Switch expressions
- Computed namespace constructors
- Output declarations
- Annotations and annotation assertions
- XQuery 3.1: **maps and arrays**

MAPS EN ARRAYS IN XQUERY

- map { 'key': true(), 1984: (<a/>, <b/>) }
  map met 2 entry's
  NB: keys heterogeneous, values can be anything
- map { }
  empty map
- map:merge(
  for $i in 1 to 10
  return map { $i: 'value' || $i }
)
  Merge 10 maps of each 1 item to 1 map with10 items
- [(1,2) , 3] *same as* array { (1,2), 3 }
  is an array with 2 values

See BaseX docs → Map_Module and Array_Module

XQUERY IN DEPTH

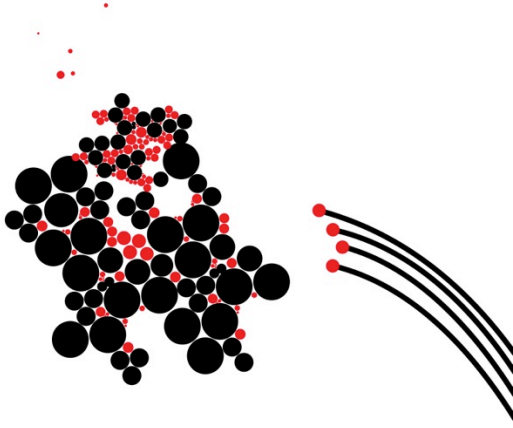
- This was a short, sketchy, incomplete intro to XQuery
- More in depth knowledge can be obtained by:
  - Looking for tutorials and fora on the web
  - Browse through the XML standard documents
    http://www.w3.org/standards/xml/query.html
    ... and you have lost all hope of ever understanding any of it, pick yourself up again and …
  - Study the examples in the W3C's Use Cases
  - Play with BaseX (of eXist, Saxon, etc.)
    BaseX documentation has many examples
  - **And of course the topic assignments**!

JSON AND XQUERY

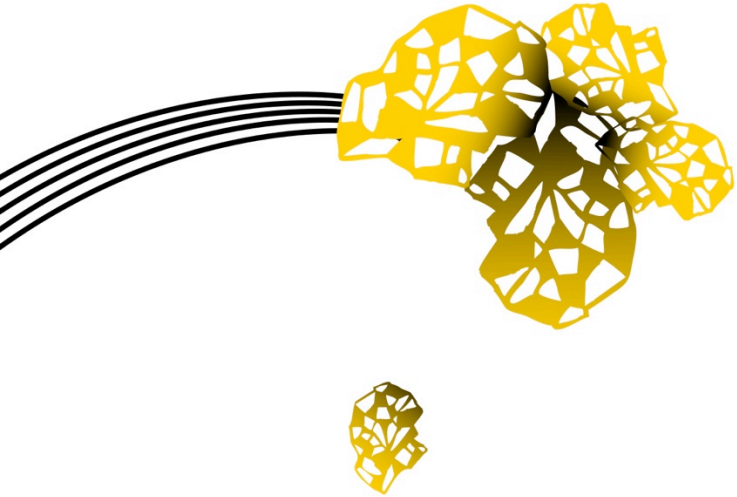
- The XML database BaseX supports a JSON module http://docs.basex.org/wiki/JSON_Module
  - *There are topic assignments on this*
- Some of the originators of XQuery developed a special query language for JSON based on lessons learned from XQuery: JSONiq http://www.jsoniq.org/

```
{
  for $sales in collection("sales")
  let $pname := $sales("product")
  group by $pname
  return
    $pname : sum(for $s in $sales return $s("quantity"))
}
```

UNIVERSITY OF TWENTE.



Semantic Web and Linked Open Data



DATA ON THE WEB
(SOURCE “W3C SEMANTIC WEB TUTORIAL”)

- There are more an more data on the Web
  - government data, health related data, general knowledge, company information, flight information, restaurants,…
- More and more applications rely on the availability of that data
- But… data are often in isolation, “silos”

The picture can't be displayed.

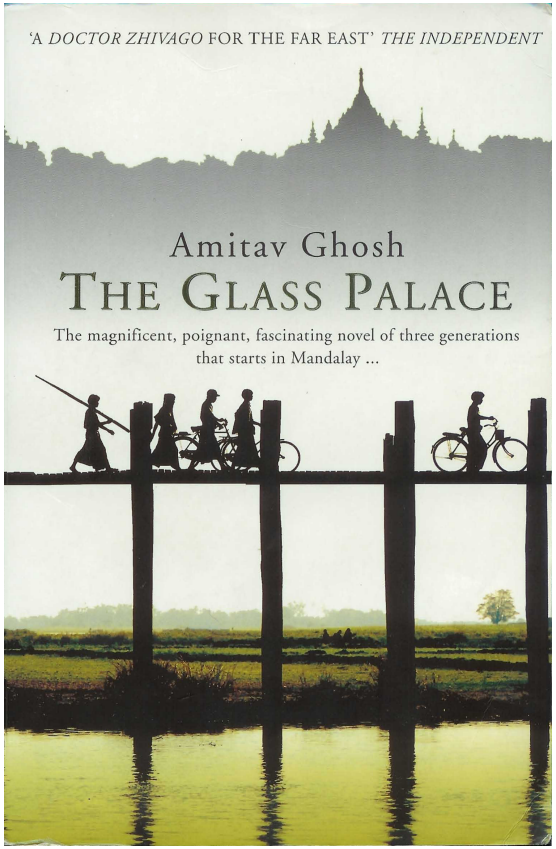
DATA ON THE WEB IS NOT ENOUGH …

- We need a proper infrastructure for a **real** *Web of Data*
  - data is available on the Web
  - accessible via standard Web technologies
  - data are interlinked over the Web
  - data can be **integrated** over the Web
- This is where Semantic Web technologies come in
- **Connect the silos!**

The picture can't be displayed.

THE ROUGH STRUCTURE OF DATA INTEGRATION

- Map the various data onto an abstract data representation
  - make the data independent of its internal representation…
- Merge the resulting representations
- Start making queries on the whole! (queries not possible on the individual data sets)
- We start with a book...



The picture can't be displayed.

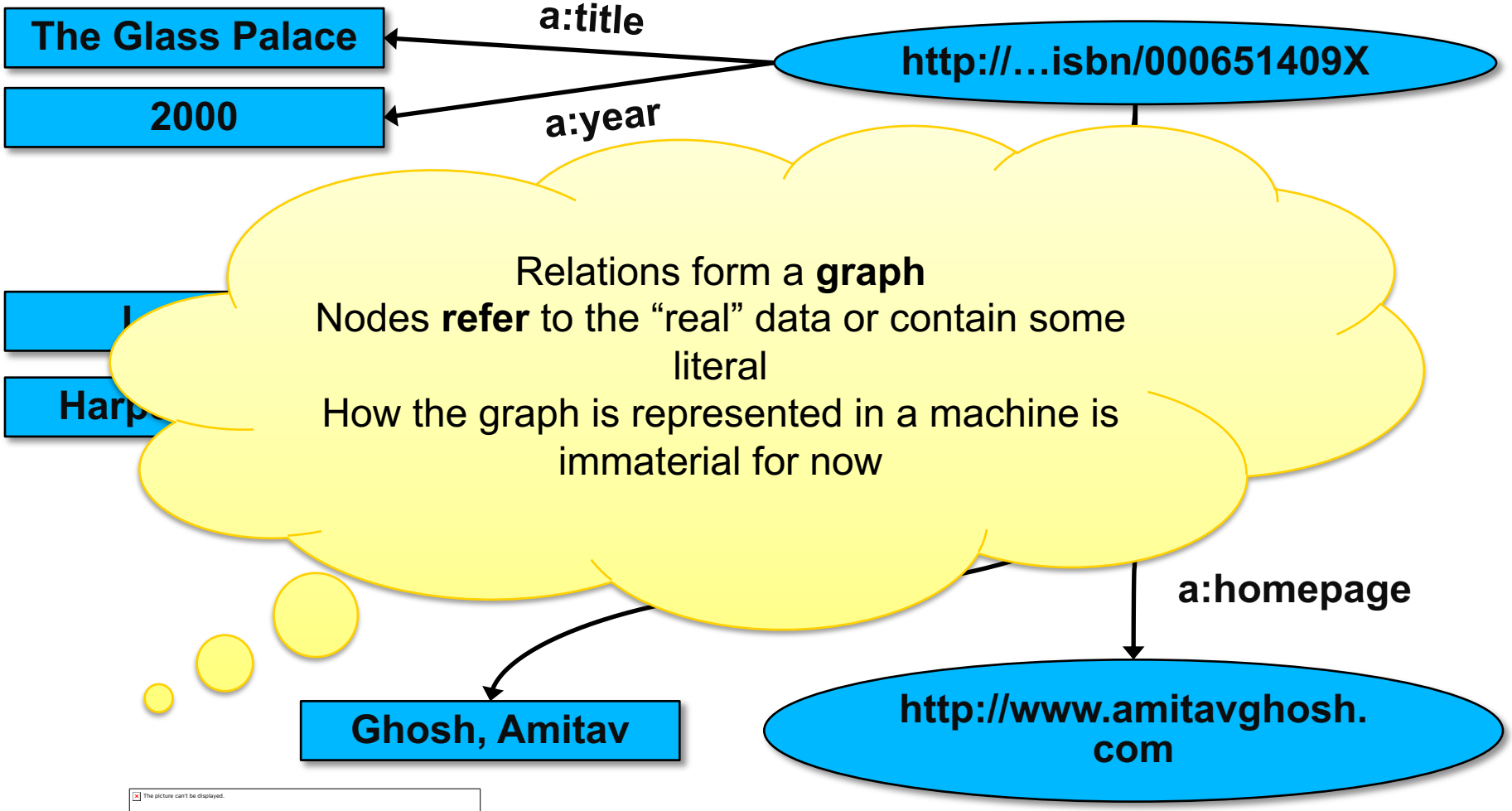
A SIMPLIFIED BOOKSTORE (DATA SET "A")
RELATIONAL DATABASE

ISBN	Author	Title	Publisher	Year
0006511409X	id_xyz	The Glass Palace	id_qpr	2000

ID	Name	Homepage
id_xyz	Ghosh, Amitav	http://www.amitavghosh.com

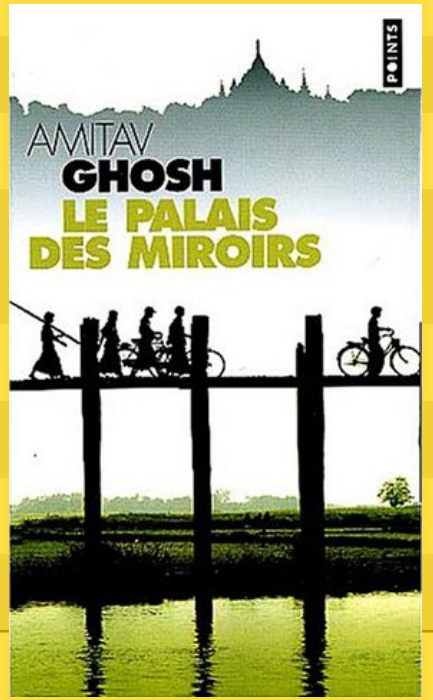
ID	Publisher's name	City
id_qpr	Harper Collins	London

1. EXPORT YOUR DATA AS A SET OF RELATIONS



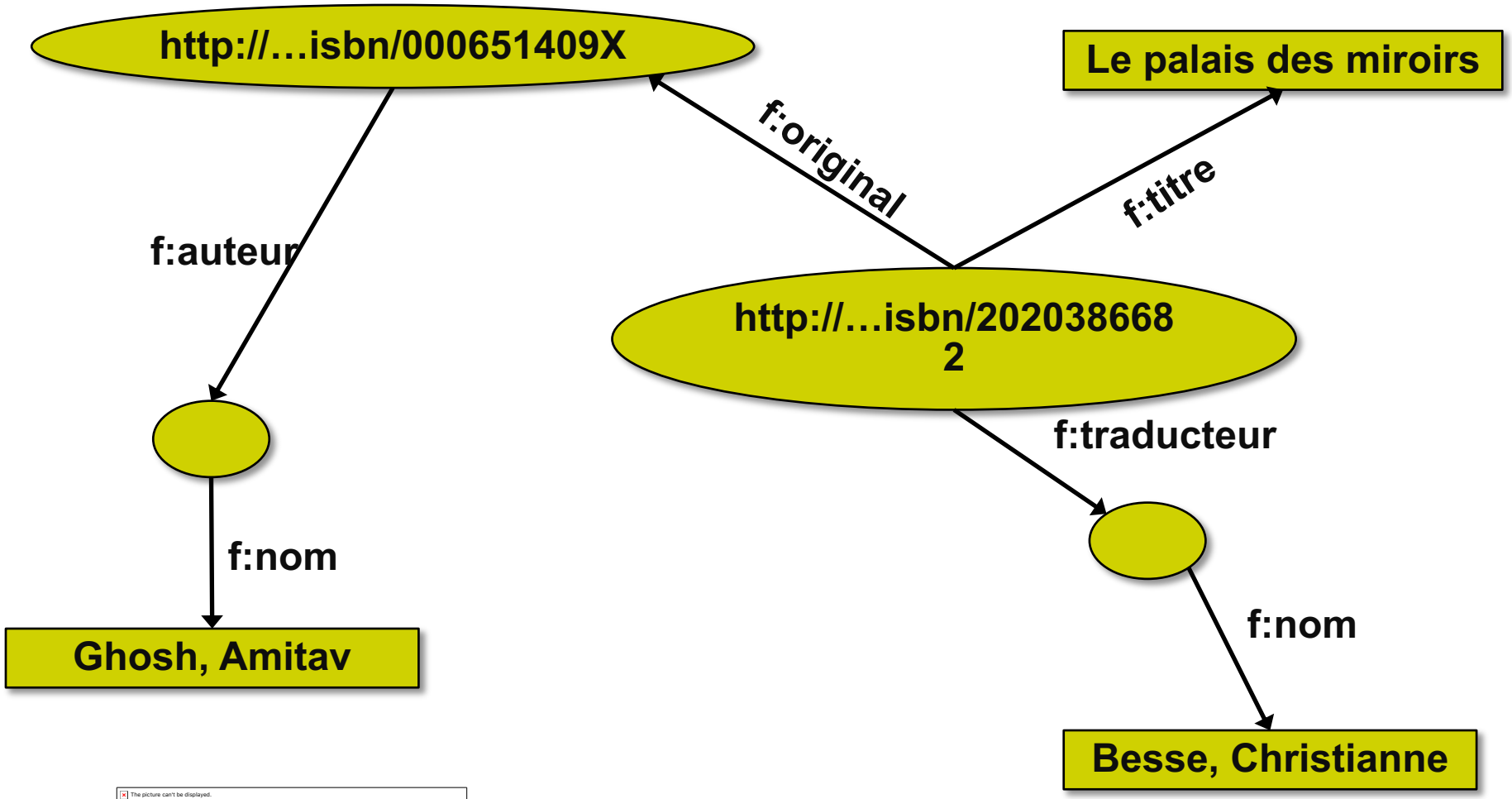
ANOTHER BOOKSTORE (DATA SET "F")
SPREADSHEET CONTAINING SAME BOOK IN FRENCH

	A	B	C	D
1	**ID**	**Titre**	**Traducteur**	**Original**
2	ISBN 2020286682	Le Palais des Miroirs	$A12$	ISBN 0-00-6511409-X
5				
6	**ID**	**Auteur**		
7	ISBN 0-00-6511409-X	$A11$		
9				
10	**Nom**			
11	Ghosh, Amitav			
12	Besse, Christianne			



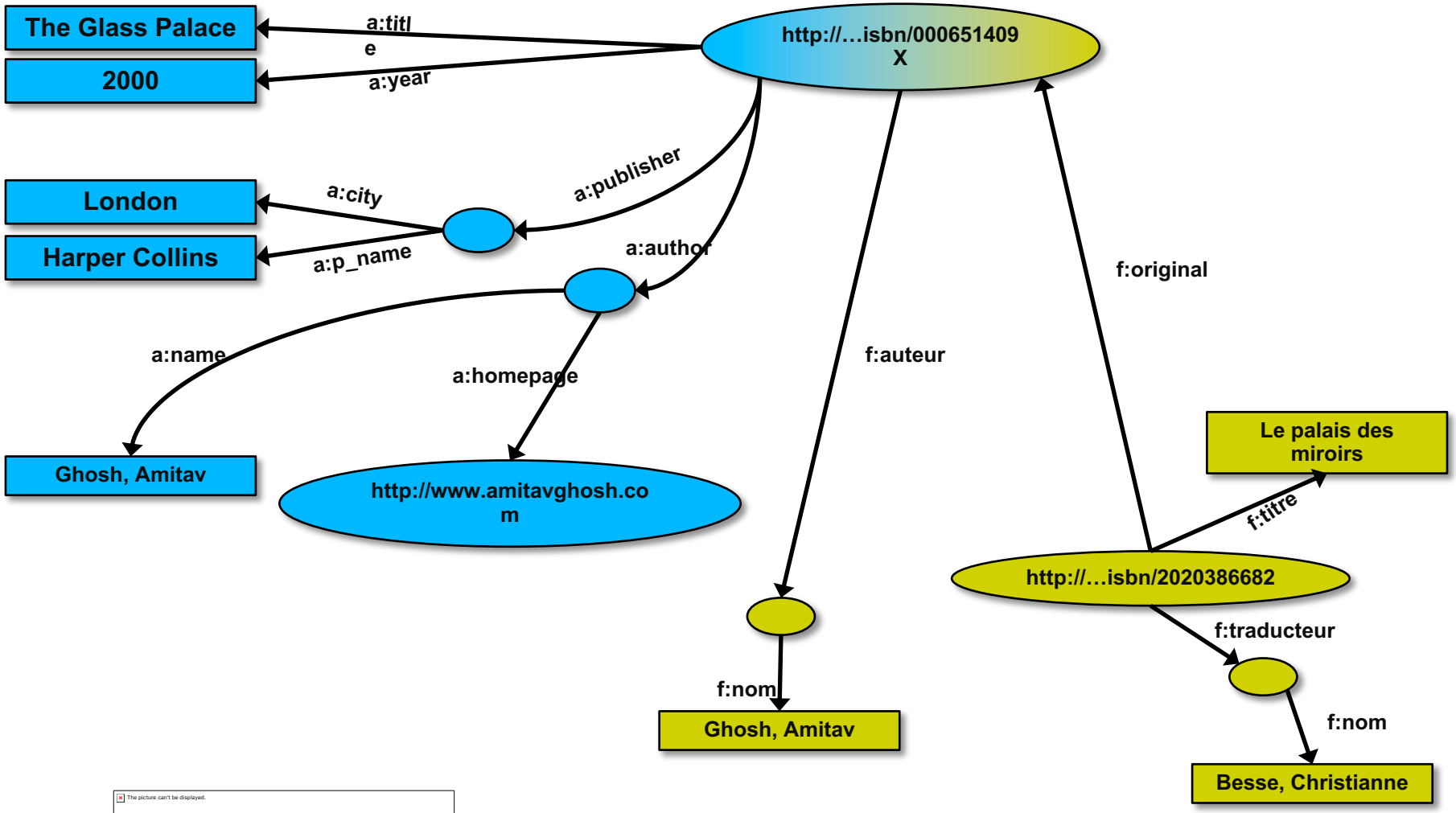
The picture can't be displayed.

2. EXPORT YOUR SECOND DATA SET



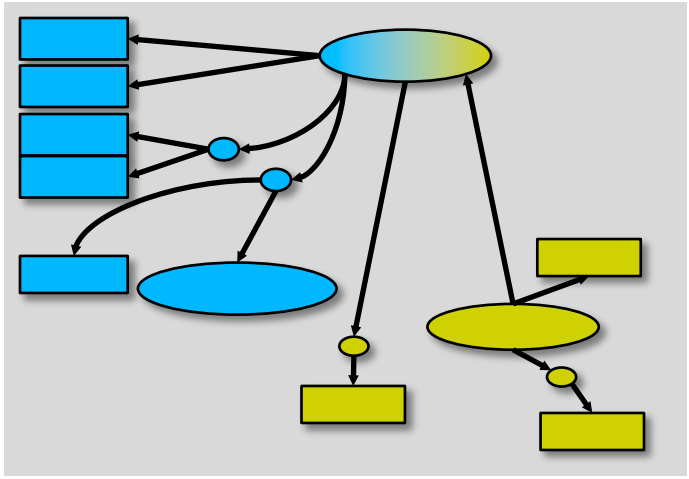
The picture can't be displayed.

3. START MERGING YOUR DATA



START MAKING QUERIES

- User of data “F” can now ask queries like:
  - “give me the title of the original”
  - well, … « donnes-moi le titre de l’original »
- This information is not in the dataset “F”…
- but can be retrieved by merging with dataset “A”!

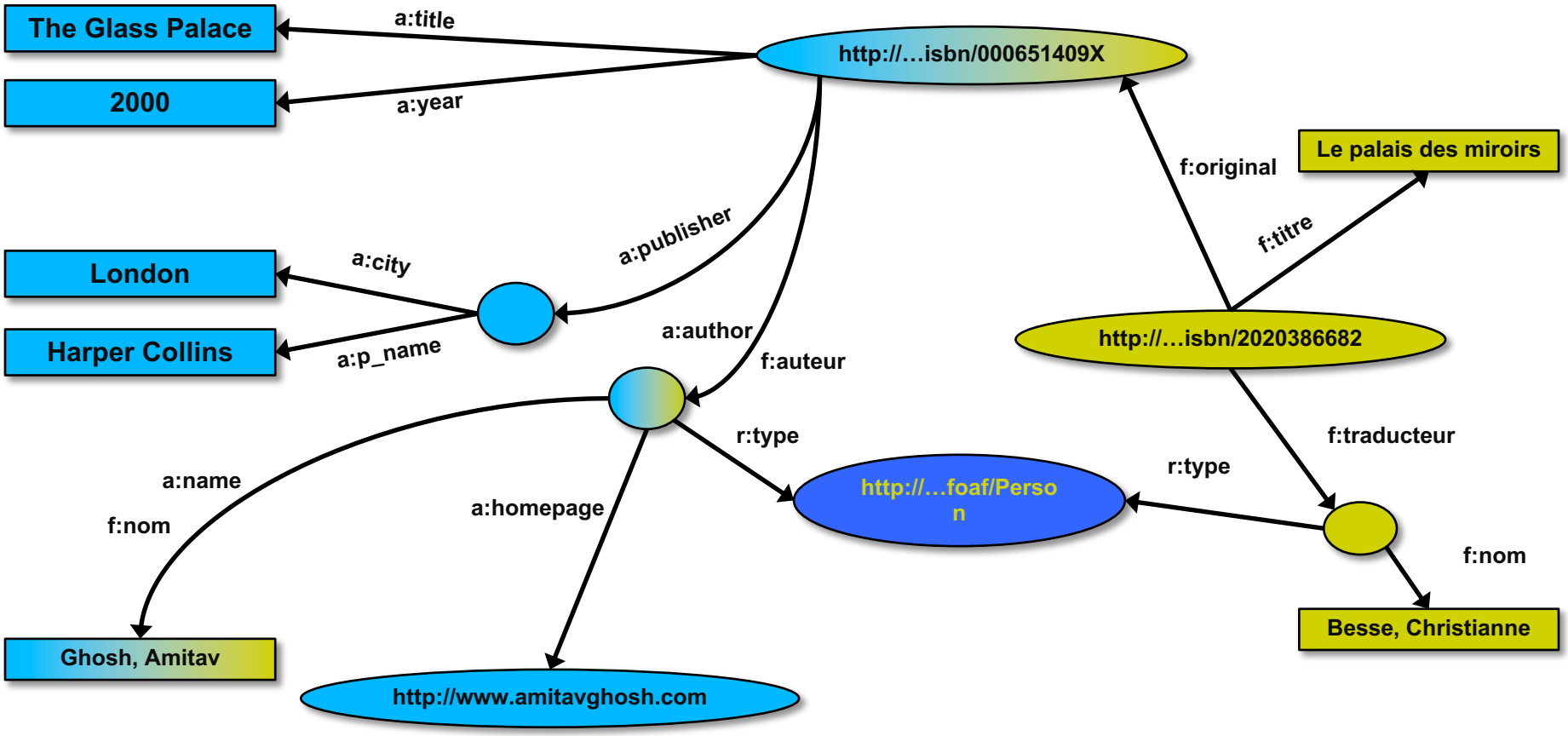


The picture can't be displayed.

HOWEVER, MORE CAN BE ACHIEVED

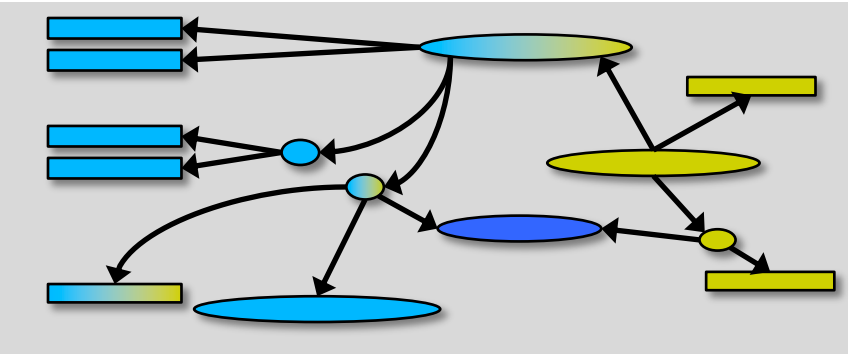
- We “feel” that *a:author* and *f:auteur* should be the same
- But an automatic merge does not know that!
- Let us add some extra information to the merged data:
  - *a:author* same as *f:auteur*
  - both identify a “Person”
  - a term that a community may have already defined:
    - a “Person” is uniquely identified by his/her name and, say, homepage
    - it can be used as a “category” for certain type of resources

3 REVISITED: USE THE EXTRA KNOWLEDGE



START MAKING RICHER QUERIES!

- User of dataset “F” can now query:
  - “donnes-moi la page d’accueil de l’auteur de l’original”
  - well… “give me the home page of the original’s ‘auteur’”
- The information is not in datasets “F” or “A”…
- …but was made available by:
  - merging datasets “A” and “F”
  - adding 3 simple extra statements as an extra “glue”



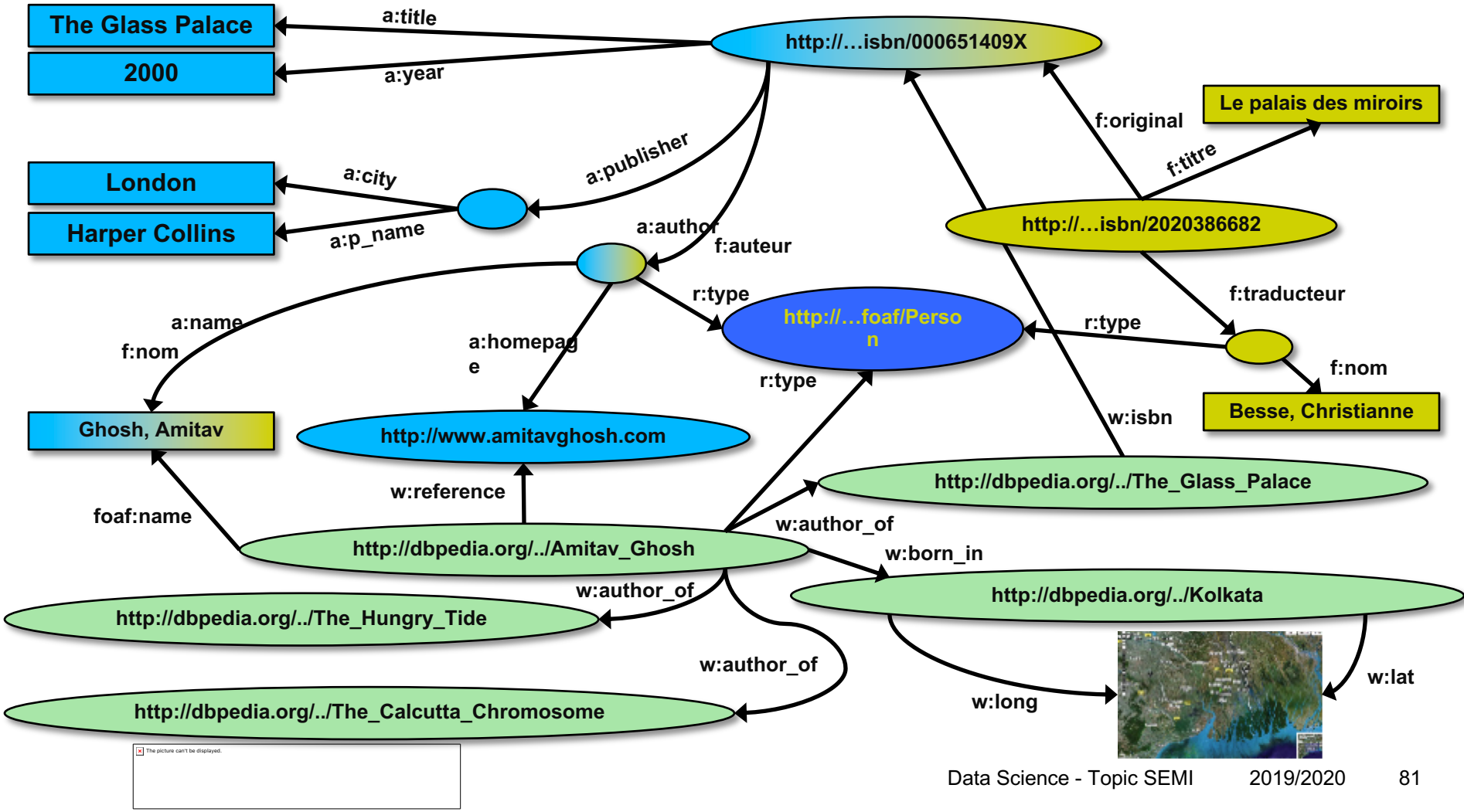
The picture can't be displayed.

CONTINUE MERGING WITH OTHER DATA SETS

- Using, e.g., the **entity** “Person”, the dataset can be combined with other sources
- For example, data in *Wikipedia* can be extracted using dedicated tools
  - e.g., the “**dbpedia**” project can extract the “infobox” information from Wikipedia already

http://en.wikipedia.org/wiki/The_Glass_Palace

MERGE WITH DBPEDIA DATA

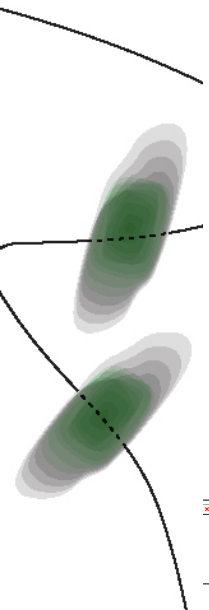


A BIT OF REFLECTION

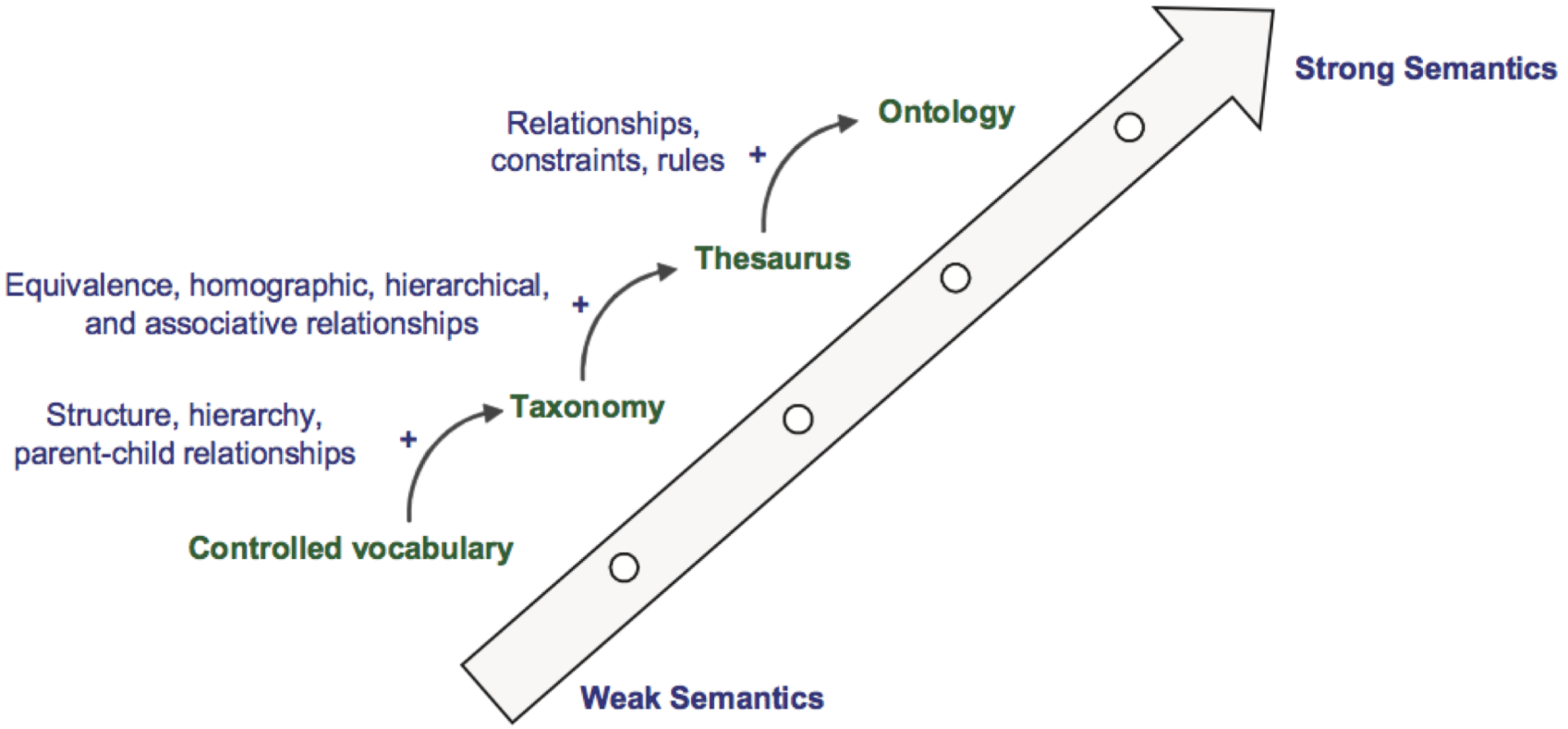
- What we did via automatic means is done every day by Web users!
- The difference: a bit of extra rigor so that machines could do this, too
- We could add extra knowledge to the merged datasets
  - e.g., a full classification of various types of library data
  - geographical information
  - etc.
- This is where ontologies, extra rules, etc, come in
  - ontologies/rule sets can be relatively simple and small, or huge, or anything in between…
- Even more powerful queries can be asked as a result

WHAT IS THE SEMANTIC WEB

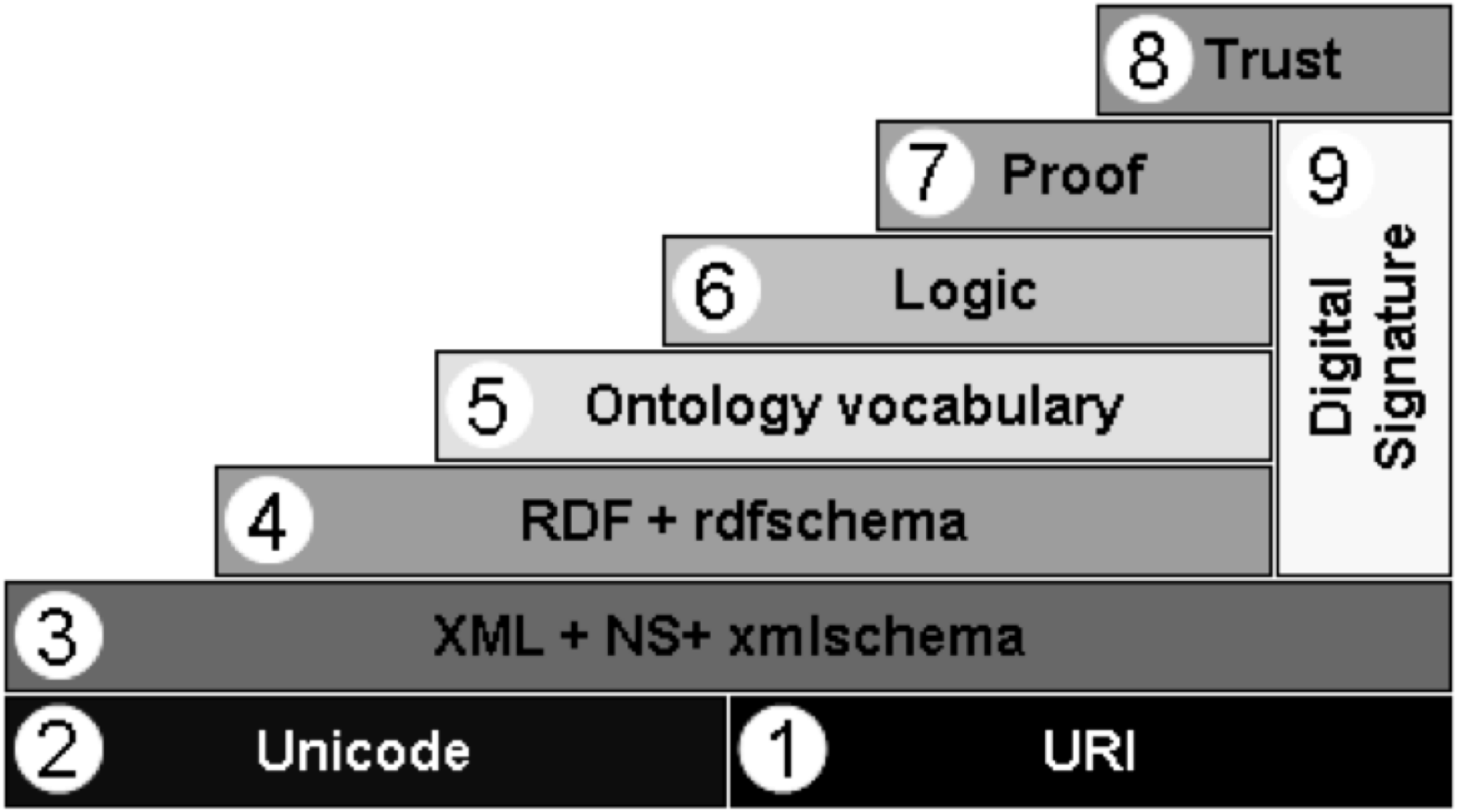
- Set of evolving standards, languages and technologies
  - languages to describe information on the web together with its semantics / meaning
  - usable by computers not only to read and transform, but also to reason with
  - to allow interoperability and integration



WHAT IS THE SEMANTIC WEB
LEVELS OF SEMANTICS



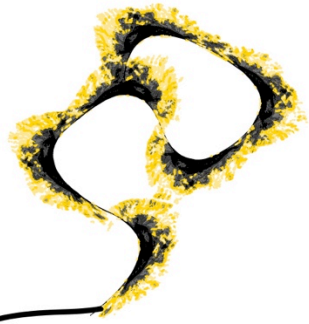
WHAT IS THE SEMANTIC WEB
SEMANTIC WEB ARCHITECTURE



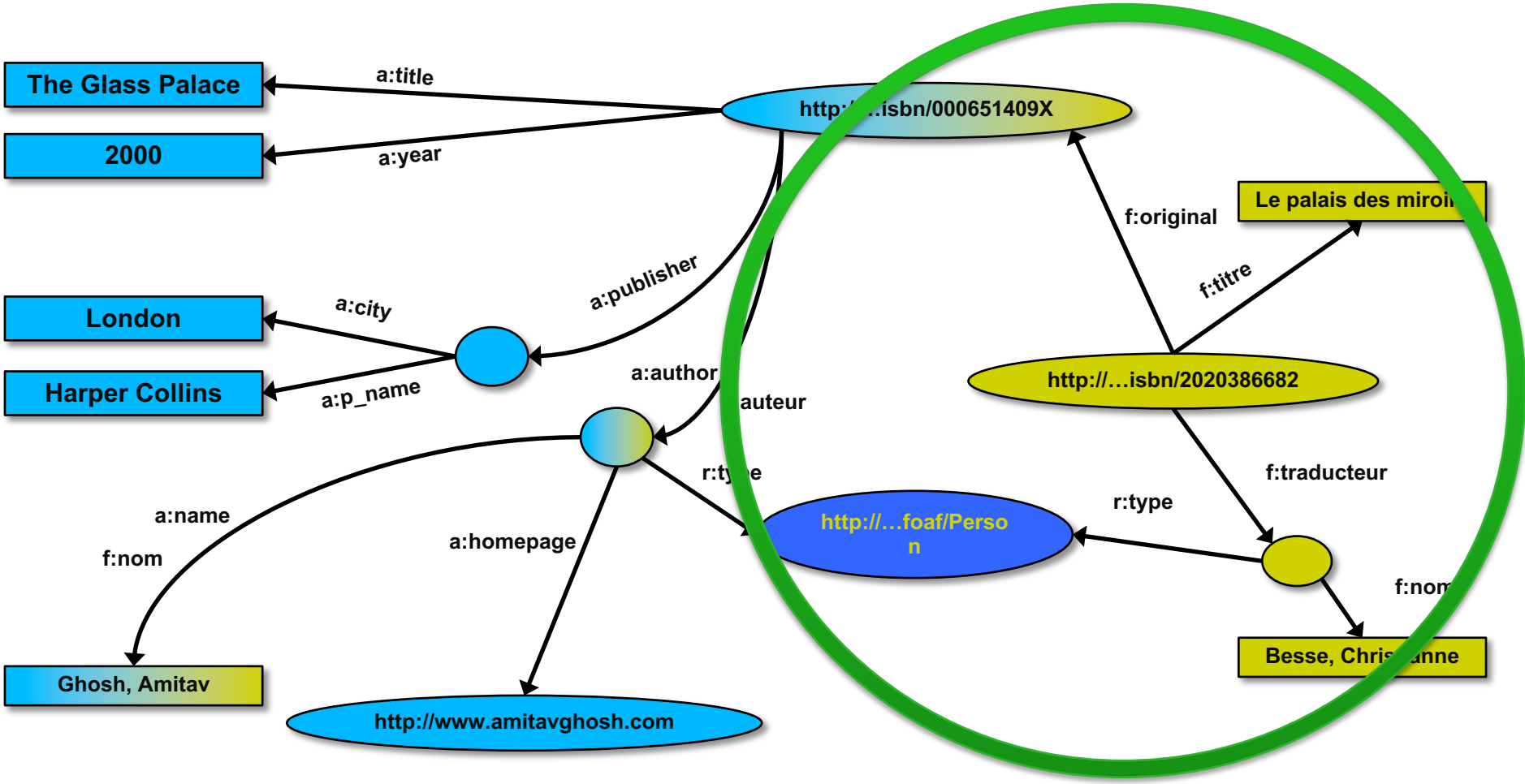
The picture can't be displayed.

UNIVERSITY OF TWENTE.

RDF and SPARQL

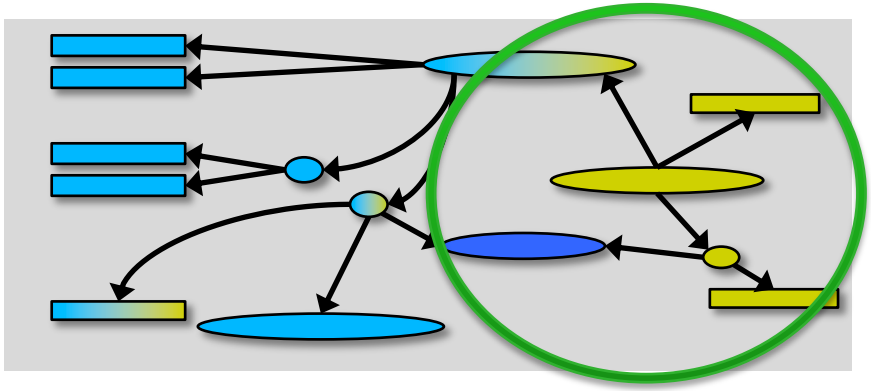


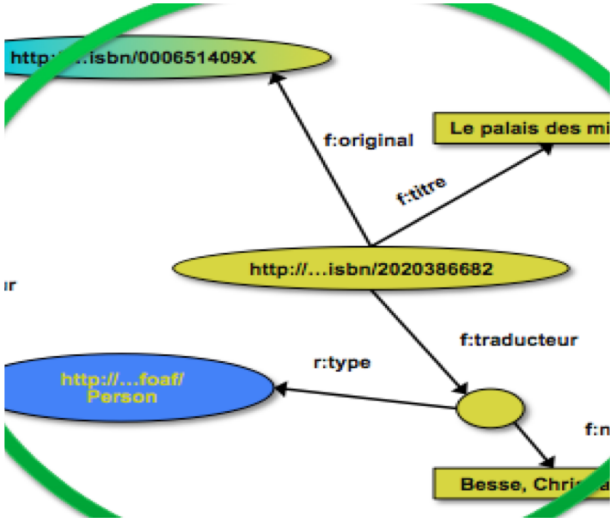
REMEMBER THIS GRAPH?
IT IS AN **RDF** GRAPH



AN RDF GRAPH IS A SET OF TRIPLES

EACH EDGE/ARROW IS A TRIPLE





@prefix foaf: http://xmlns.com/foaf/0.1/

@prefix rdf: http://www.w3.org/1999/02/22-rdf-syntax-ns#

@prefix f: http://…

(http://…isbn/2020386682, f:orginal, http://…isbn/000651409X)

(http://…isbn/000651409X, f:titre, “Le palais des miroirs)

(http://…isbn/000651409X, f:traducteur, _:a)

(_:a, f:nom, “Besse, Christianne”)

(_:a, rdf:type, foaf:Person)

WHAT IS THE SEMANTIC WEB
RDF / RDFS / OWL EXAMPLE

```
<owl:Class
   rdf:about="http://wwwhome.cs.utwente.nl/~keulen/ontologies/2011/Test
   .owl#OWLClass_012962256265304840000">
    <rdfs:label>Pizza</rdfs:label>
    <rdfs:subClassOf
   rdf:resource="http://wwwhome.cs.utwente.nl/~keulen/ontologies/2011/T
   est.owl#OWLClass_012962256265255500000"/>
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty
   rdf:resource="http://wwwhome.cs.utwente.nl/~keulen/ontologies/2011/T
   est.owl#OWLObjectProperty_012962266046868080000"/>
          <owl:someValuesFrom
   rdf:resource="http://wwwhome.cs.utwente.nl/~keulen/ontologies/2011/T
   est.owl#OWLClass_012962259664293110000"/>
        </owl:Restriction>
    </rdfs:subClassOf>
   </owl:Class>
```

Description logic definition
A pizza is an independent entity that, among other things, has a tomato topping

$$Pizza \subseteq Independent\_entity \wedge \exists hasTopping.Tomato\_topping$$

OPEN DATA SET “DBPEDIA”

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

dbpedia:Amsterdam
  dbterm:officialName "Amsterdam" ;
  dbterm:longd "4" ;
  dbterm:longm "53" ;
  dbterm:longs "32" ;
  dbterm:website <http://www.amsterdam.nl> ;
  dbterm:populationUrban "1364422" ;
  dbterm:areaTotalKm "219" ;
  ...
dbpedia:ABN_AMRO
  dbterm:location dbpedia:Amsterdam ;
  ...
```

Amsterdam

— Municipality / City —

Coordinates: 52°22′23″N 4°53′32″E

Country	Netherlands
Province	North Holland
COROP	Amsterdam
Boroughs	Boroughs
Government	
- Mayor	Eberhard van der Laan (PvdA)
- Aldermen	Carolien Gijkiels Hans Gerson Maarten van Poelgeest Freek Ossel Marijke Vos
- Secretary	Henk de Jong
Area[1][2]	
- Municipality / City	219 km² (84.6 sq mi)
- Land	166 km² (64.1 sq mi)
- Water	53 km² (20.5 sq mi)
- Urban	1,003 km² (387.3 sq mi)
- Metro	1,815 km² (700.8 sq mi)
Elevation[3]	2 m (7 ft)
Population (June 2009)[4][5]	
- Municipality / City	762,057
- Density	4,459/km² (11,548.8/sq mi)
- Urban	1,364,422

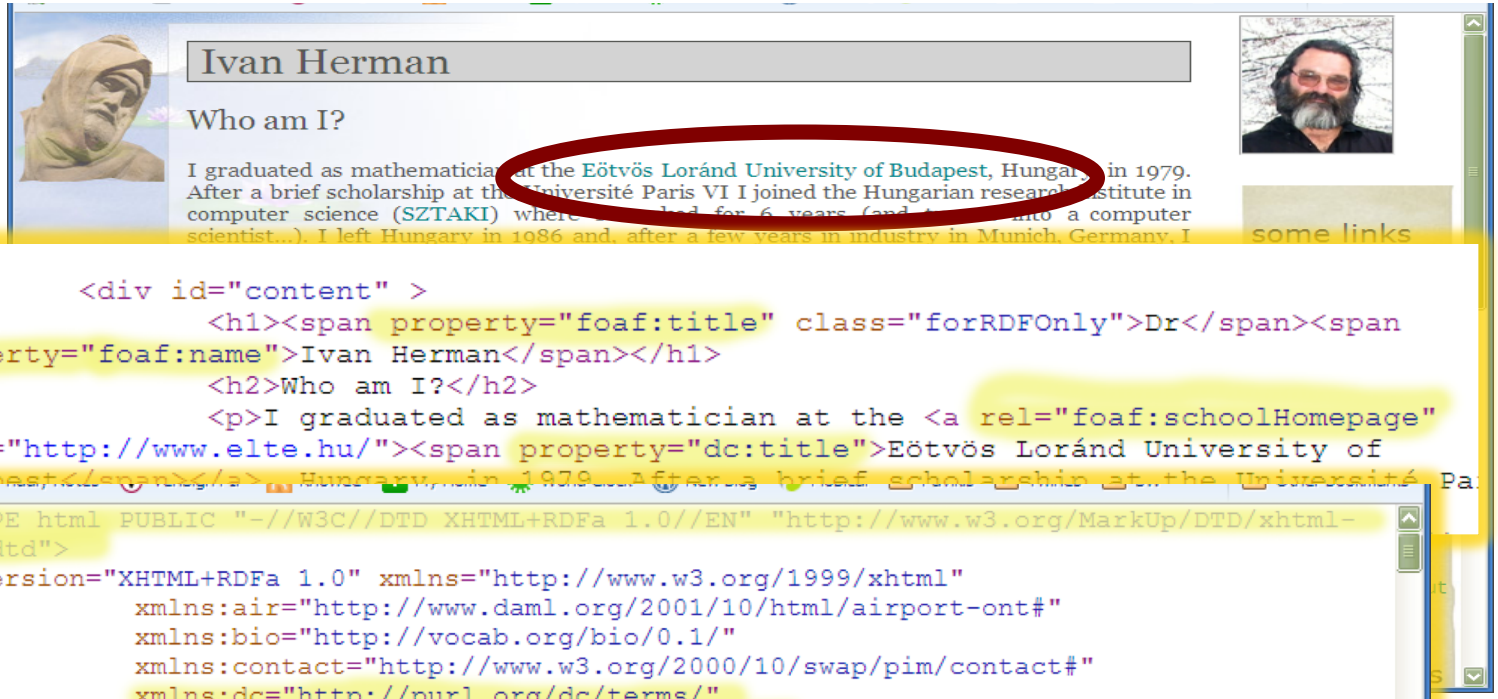
LINKING AMONG OPEN DATASETS

```
<http://dbpedia.org/resource/Amsterdam>
  owl:sameAs <http://rdf.freebase.com/ns/...> ;
  owl:sameAs <http://sws.geonames.org/2759793> ;
  ...
```

```
<http://sws.geonames.org/2759793>
  owl:sameAs <http://dbpedia.org/resource/Amsterdam>
  wgs84_pos:lat "52.3666667" ;
  wgs84_pos:long "4.8833333";
  geo:inCountry <http://www.geonames.org/countries/#NL> ;
 ...
```

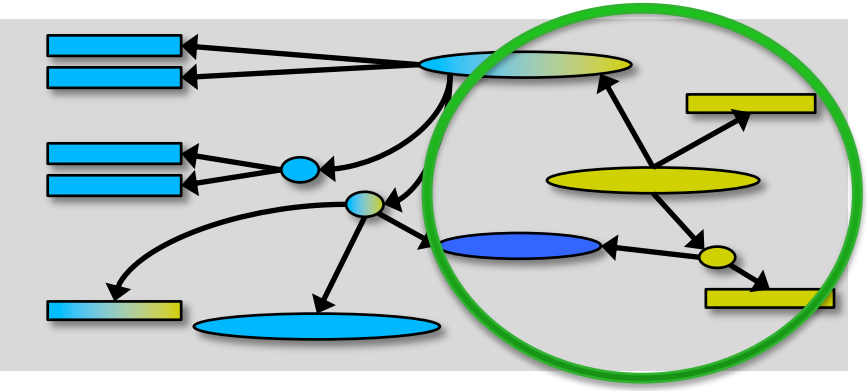
RDFa: ONE WAY OF ADDING KNOWLEDGE TO YOUR DATA

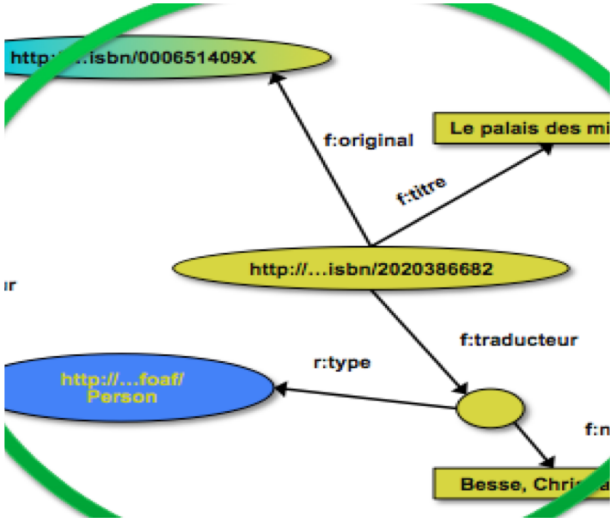
- RDFa uses XHTML attrs for “meta” information



AN RDF GRAPH IS A SET OF TRIPLES

EACH EDGE/ARROW IS A TRIPLE





@prefix foaf: http://xmlns.com/foaf/0.1/

@prefix rdf: http://www.w3.org/1999/02/22-rdf-syntax-ns#

@prefix f: http://…

(http://…isbn/2020386682, f:orginal, http://…isbn/000651409X)

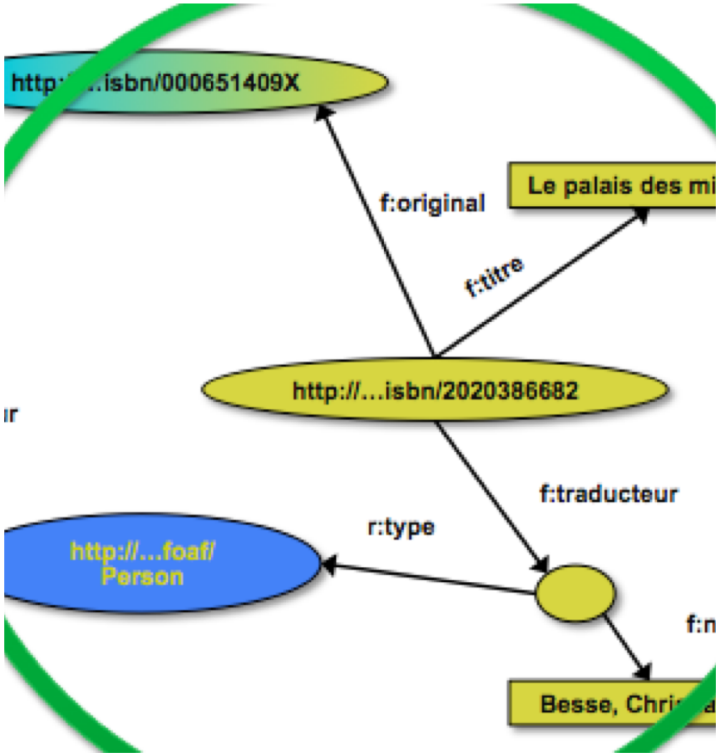
(http://…isbn/000651409X, f:titre, “Le palais des miroirs)

(http://…isbn/000651409X, f:traducteur, _:a)

(_:a, f:nom, “Besse, Christianne”)

(_:a, rdf:type, foaf:Person)

SPARQL – EXAMPLE QUERY

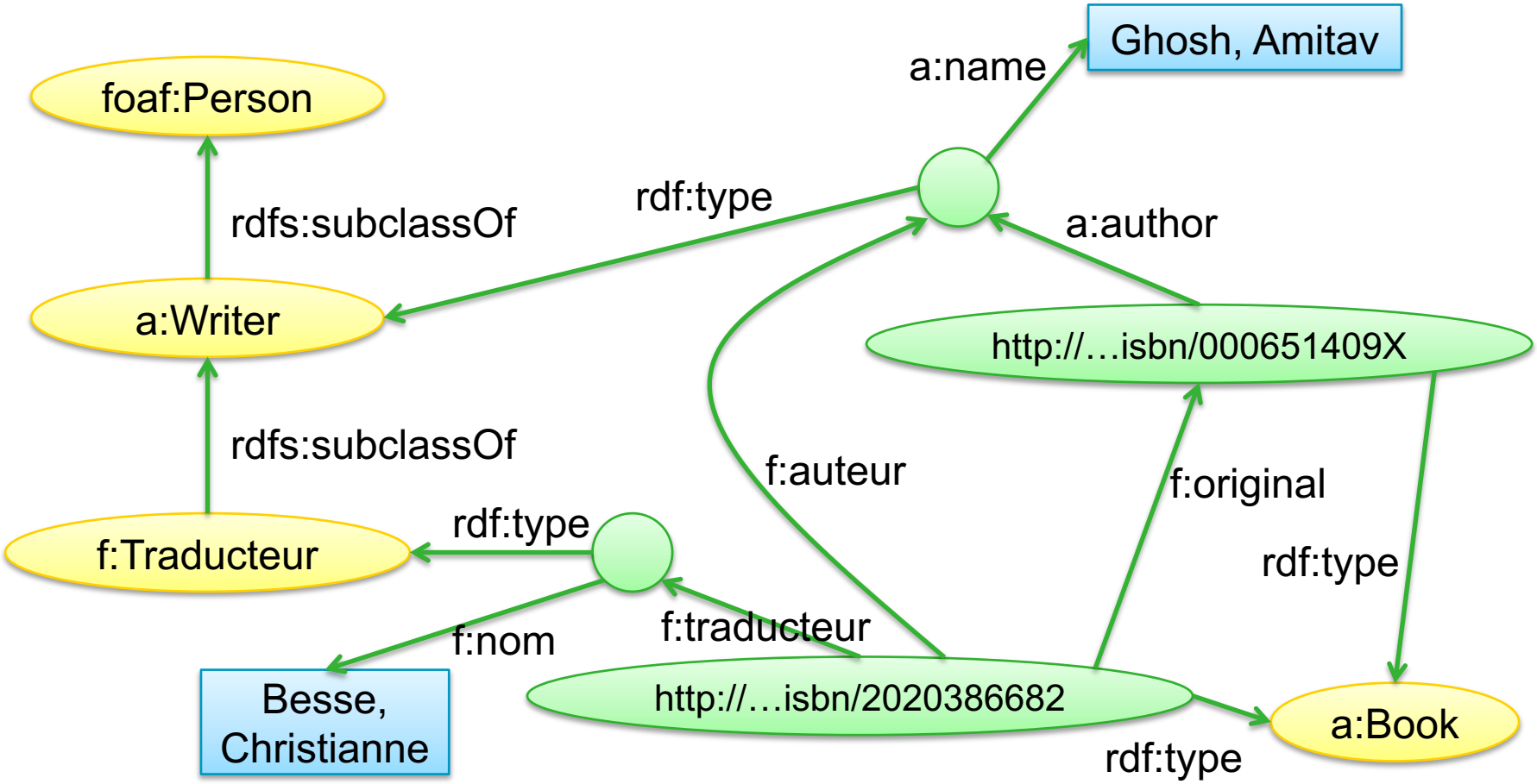



```
PREFIX f: <http://…>

SELECT ?n
FROM NAMED <bookstore.rdf>
WHERE {
  ?livre f:titre ”Le palais
des miroirs" .
  ?livre f:traducteur ?trad .
  ?trad f:nom ?n
}
```

LET'S WRITE A QUERY OF OUR OWN

GIVE ME THE NAMES OF ALL WRITERS ASSOCIATED WITH THE FRENCH BOOK



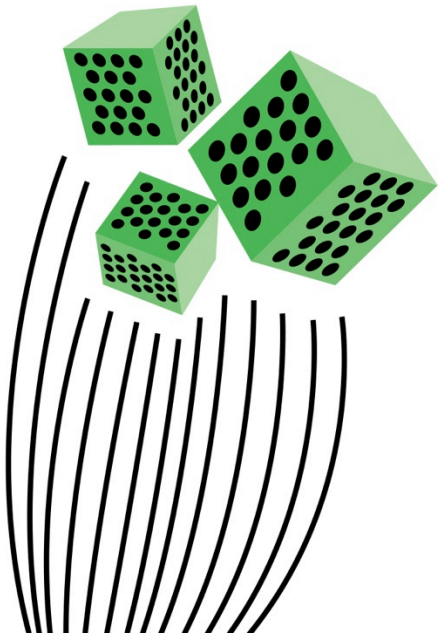
SPARQL FEATURES

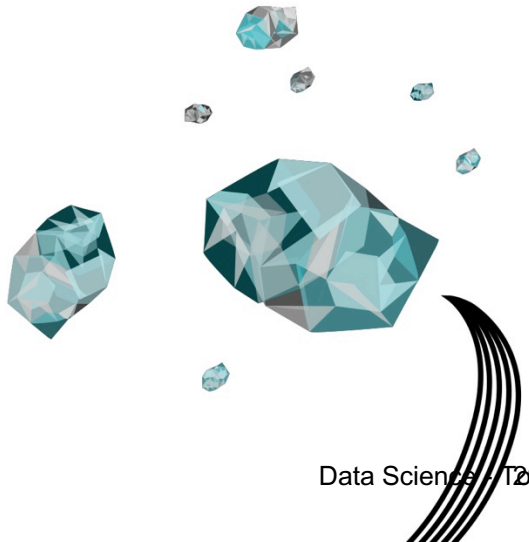
- Subgraph matching
- Any position in triple can be a variable
- Optional, filter, predicates, etc. etc.
- Querying multiple RDF graphs
- Constructing new RDF graphs
- Several RDF stores (“triple stores”) supporting SPARQL querying

BUT

- Only RDF, no RDFS, so no subclassing!

CONCLUSIONS





SUMMARY

- History of web standards
- XML / JSON
  - Relational to XML with SQL/XML
  - XML Databases (XPath / XQuery / JSONiq)

- What is (the purpose) the Semantic Web
  - Integrating data on the web
  - Linked Open Data (LOD)
  - Data : RDF; and querying : SPARQL