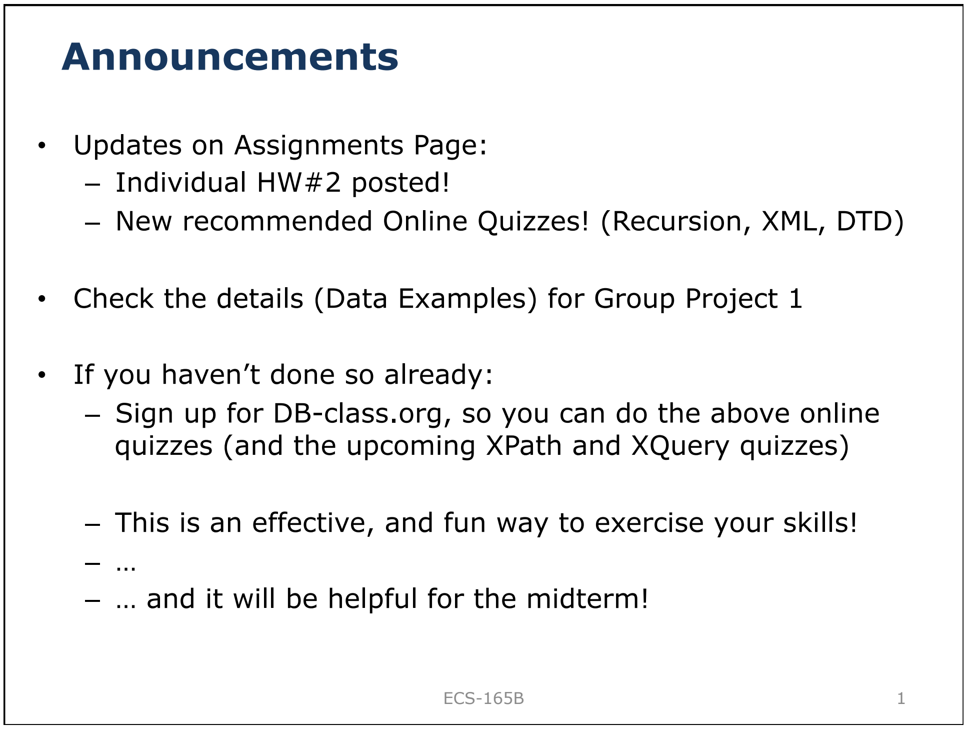
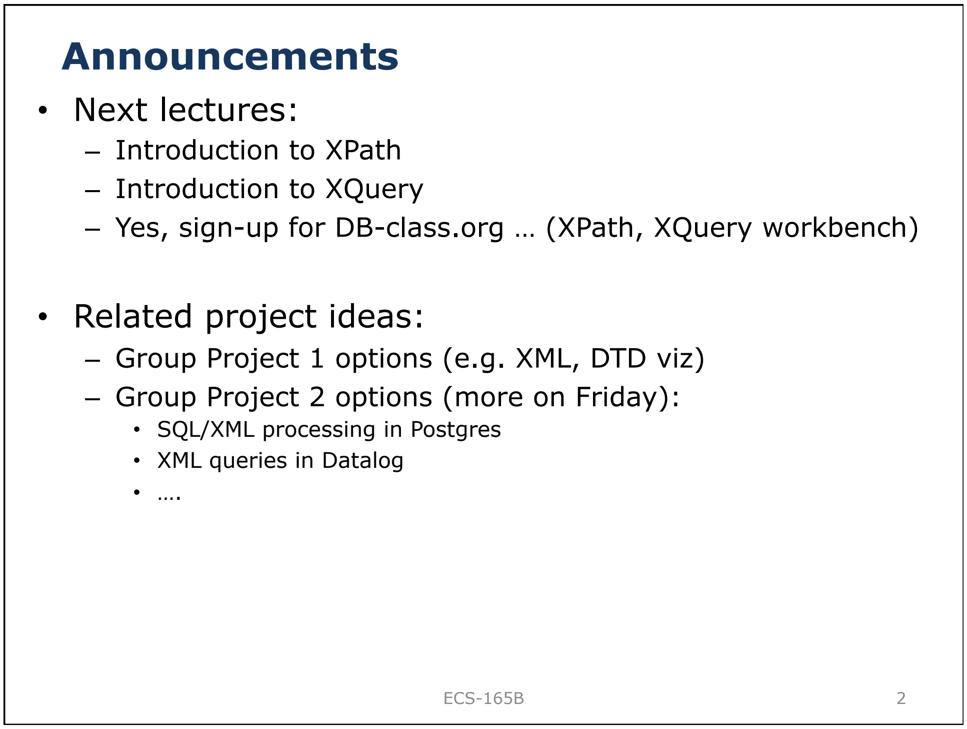


Announcements

- Updates on Assignments Page:
  - Individual HW#2 posted!
  - New recommended Online Quizzes! (Recursion, XML, DTD)
- Check the details (Data Examples) for Group Project 1
- If you haven’t done so already:
  - Sign up for DB-class.org, so you can do the above online quizzes (and the upcoming XPath and XQuery quizzes)
  - This is an effective, and fun way to exercise your skills!
  - …
  - … and it will be helpful for the midterm!



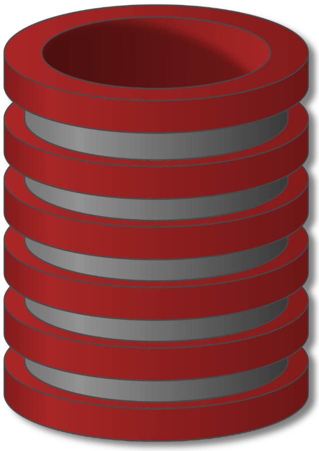
Announcements

- Next lectures:
  - Introduction to XPath
  - Introduction to XQuery
  - Yes, sign-up for DB-class.org … (XPath, XQuery workbench)
- Related project ideas:
  - Group Project 1 options (e.g. XML, DTD viz)
  - Group Project 2 options (more on Friday):
    - SQL/XML processing in Postgres
    - XML queries in Datalog
    - ….

Processing XML: SAX (**S**imple **A**PI for **X**ML)

- In SAX an XML tree is not viewed as a data structure, but as a **stream** of events generated by the parser. The kinds of events are:
  - the **start** and **end** of the document is encountered <a> </a>
  - the **start tag** or the **end tag** of an element is encountered
  - **character data** is encountered "string"
  - a processing instruction is encountered
- Scanning the XML file from start to end, each event invokes a corresponding callback method that the programmer writes.
- An XML tree can be built in response, but it is not required to construct a data structure.
- This is sometimes much more efficient:
  - the document can be **piped** through the application
  - the only real option for very large documents
  - good for local processing, not for random access

Q: Think about SAX/streaming "friendly" queries! (and those which aren't)

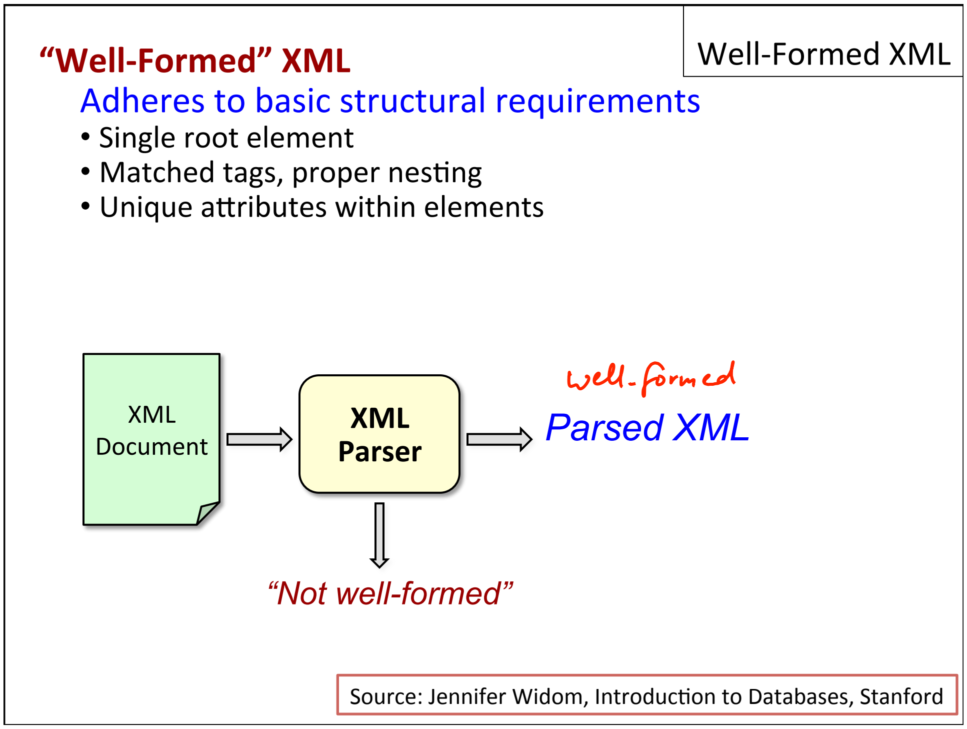


Introduction to Databases

- Well-formed XML ✓
- XML DTDs ✓
- XML Schema ✓

watch @db-class.org!!

Jennifer Widom
Stanford



Well-Formed XML

"Well-Formed" XML

Adheres to basic structural requirements

- Single root element
- Matched tags, proper nesting
- Unique attributes within elements

XML Document

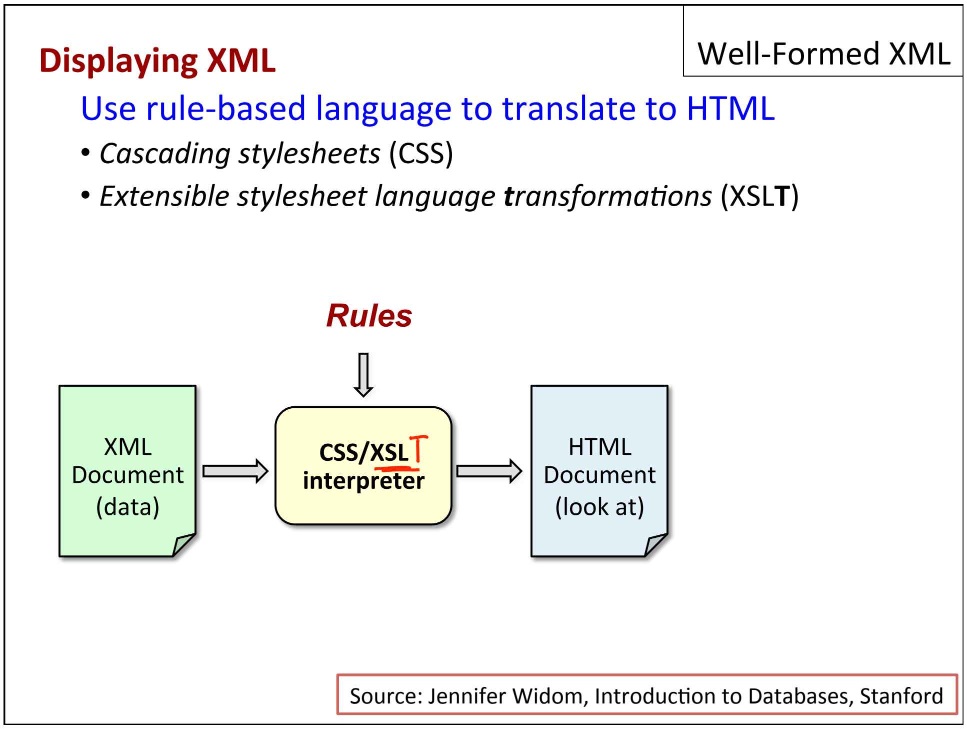
XML Parser

well-formed

Parsed XML

"Not well-formed"

Source: Jennifer Widom, Introduction to Databases, Stanford



Well-Formed XML

Displaying XML

Use rule-based language to translate to HTML

- *Cascading stylesheets* (CSS)
- *Extensible stylesheet language **t**ransformations* (XSL**T**)

Rules

XML Document (data)

CSS/XSLT interpreter

HTML Document (look at)

Source: Jennifer Widom, Introduction to Databases, Stanford

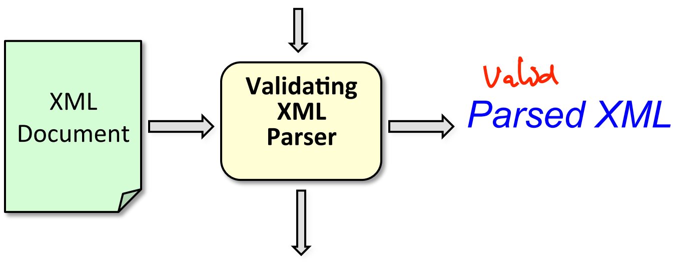
DTDs, IDs & IDREFs

"Valid" XML

Adheres to basic structural requirements

➢Also adheres to content-specific specification

DTD or XSD



"Not valid"

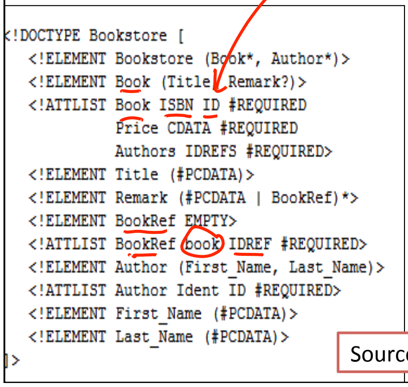
Source: Jennifer Widom, Introduction to Databases, Stanford

DTDs, IDs & IDREFs

Document Type Descriptor (DTD)

- Grammar-like language for specifying elements, attributes, nesting, ordering, #occurrences
- Also special attribute types ID and IDREF(S)

must be unique!
(scope?)

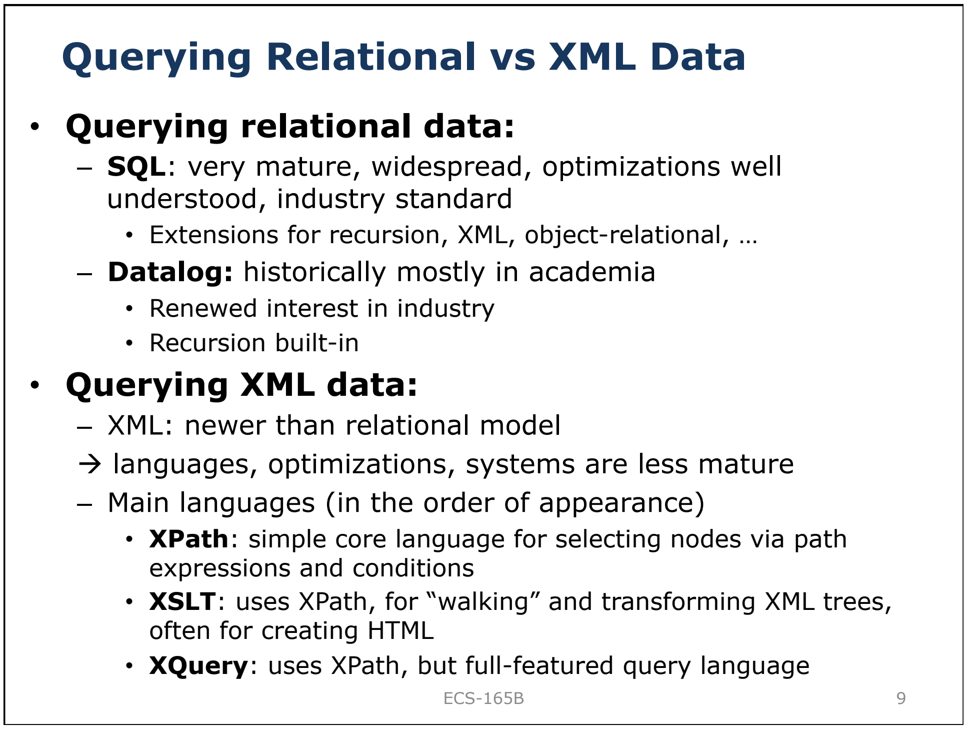


```
<!DOCTYPE Bookstore [
   <!ELEMENT Bookstore (Book*, Author*)>
   <!ELEMENT Book (Title, Remark?)>
   <!ATTLIST Book ISBN ID #REQUIRED
             Price CDATA #REQUIRED
             Authors IDREFS #REQUIRED>
   <!ELEMENT Title (#PCDATA)>
   <!ELEMENT Remark (#PCDATA | BookRef)*>
   <!ELEMENT BookRef EMPTY>
   <!ATTLIST BookRef book IDREF #REQUIRED>
   <!ELEMENT Author (First_Name, Last_Name)>
   <!ATTLIST Author Ident ID #REQUIRED>
   <!ELEMENT First_Name (#PCDATA)>
   <!ELEMENT Last_Name (#PCDATA)>
]>
```

"17 18"

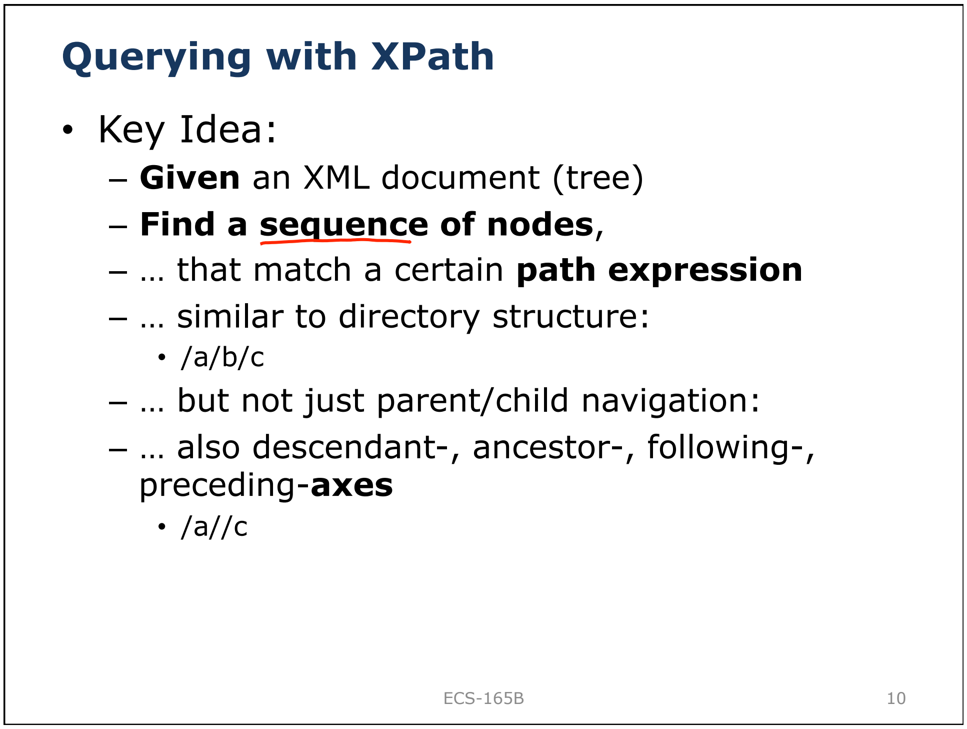
```
<Book ISBN=.. Price="$20" Authors=" ">
  <title> ... </title>
  <Remark> Great Book! </Remark>
</Book>
<Author Ident="17">
  ...
</Author>
...
```

Source: Jennifer Widom, Introduction to Databases, Stanford



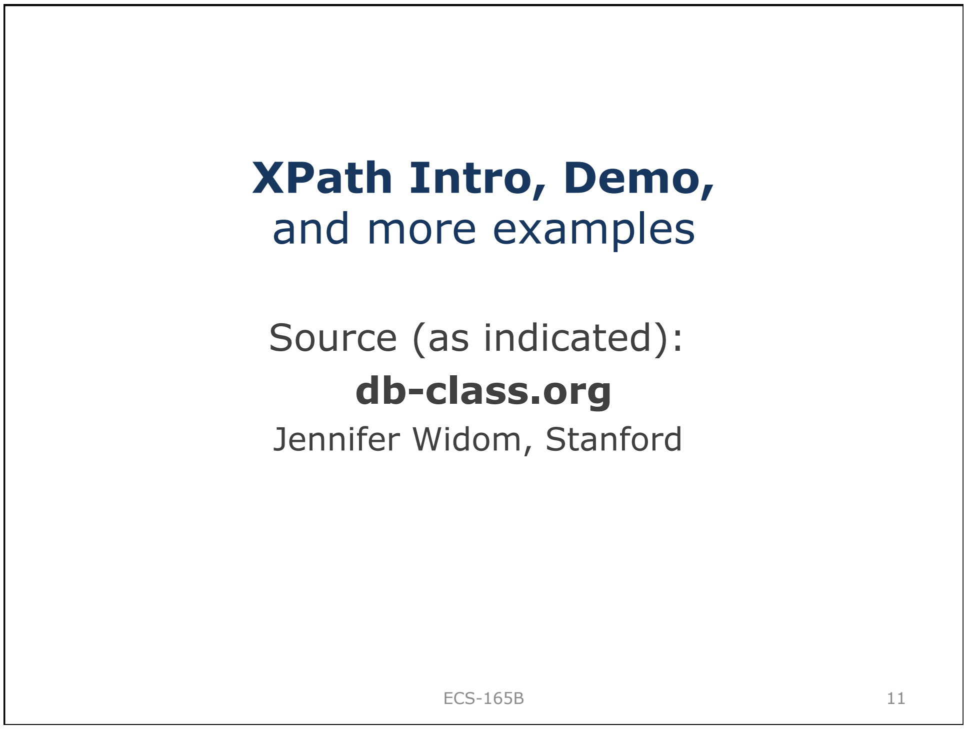
Querying Relational vs XML Data

- **Querying relational data:**
  - **SQL**: very mature, widespread, optimizations well understood, industry standard
    - Extensions for recursion, XML, object-relational, …
  - **Datalog:** historically mostly in academia
    - Renewed interest in industry
    - Recursion built-in
- **Querying XML data:**
  - XML: newer than relational model
  - → languages, optimizations, systems are less mature
  - Main languages (in the order of appearance)
    - **XPath**: simple core language for selecting nodes via path expressions and conditions
    - **XSLT**: uses XPath, for "walking" and transforming XML trees, often for creating HTML
    - **XQuery**: uses XPath, but full-featured query language



Querying with XPath

- Key Idea:
  - **Given** an XML document (tree)
  - **Find a sequence of nodes**,
  - … that match a certain **path expression**
  - … similar to directory structure:
    - /a/b/c
  - … but not just parent/child navigation:
  - … also descendant-, ancestor-, following-, preceding-**axes**
    - /a//c

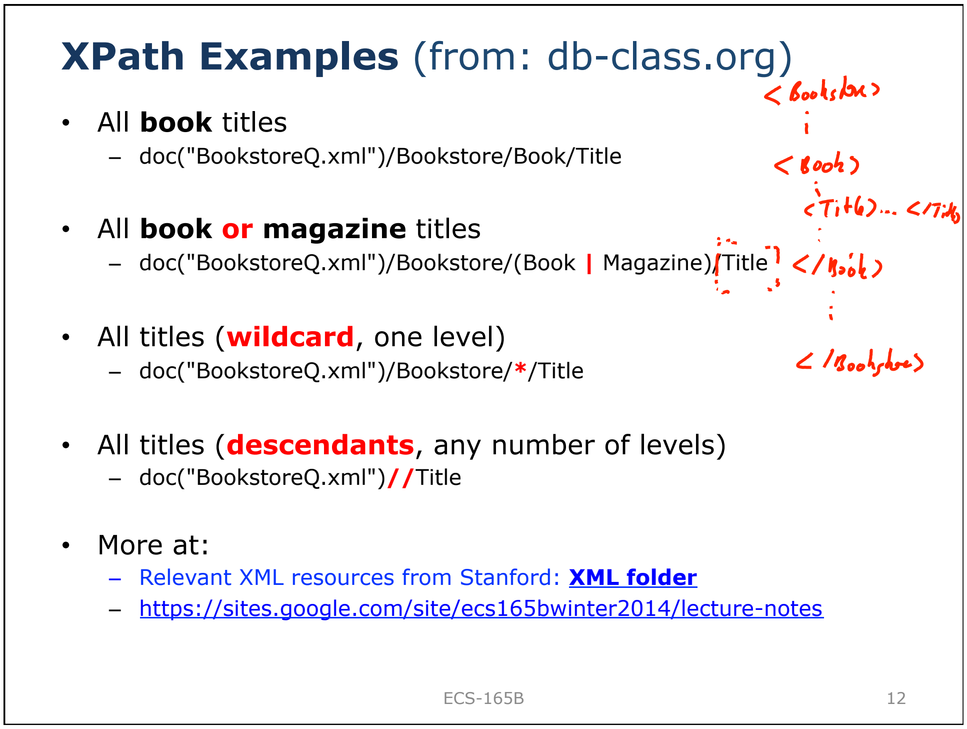


XPath Intro, Demo, and more examples

Source (as indicated):

db-class.org

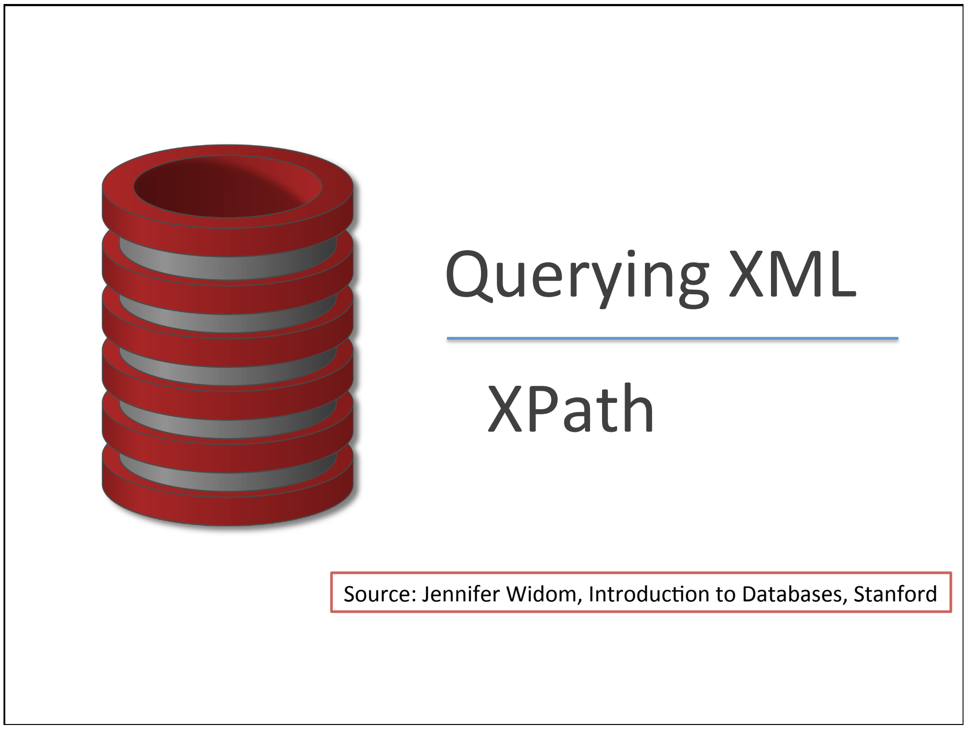
Jennifer Widom, Stanford



XPath Examples (from: db-class.org)

- All **book** titles
  - doc("BookstoreQ.xml")/Bookstore/Book/Title
- All **book or magazine** titles
  - doc("BookstoreQ.xml")/Bookstore/(Book | Magazine)/Title
- All titles (**wildcard**, one level)
  - doc("BookstoreQ.xml")/Bookstore/*/Title
- All titles (**descendants**, any number of levels)
  - doc("BookstoreQ.xml")//Title
- More at:
  - Relevant XML resources from Stanford: **XML folder**
  - https://sites.google.com/site/ecs165bwinter2014/lecture-notes

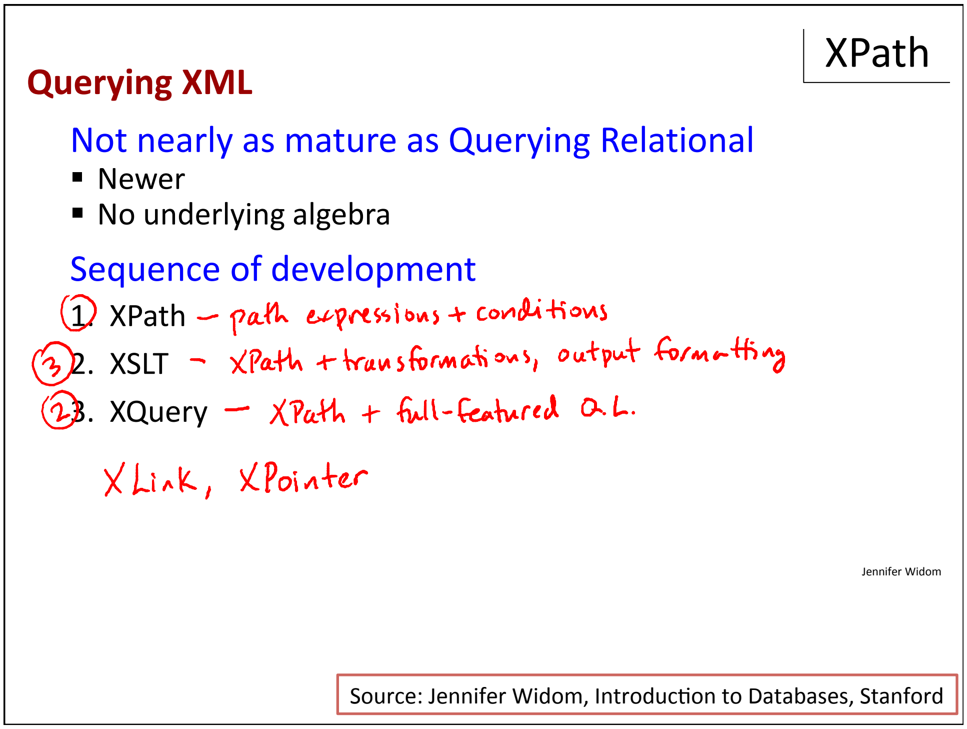
<Bookstore>
:
<Book>
<Title>... </Title>
</Book>
:
</Bookstore>



Querying XML

XPath

Source: Jennifer Widom, Introduction to Databases, Stanford



XPath

Querying XML

Not nearly as mature as Querying Relational

- Newer
- No underlying algebra

Sequence of development

① 1. XPath – path expressions + conditions

③ 2. XSLT – XPath + transformations, output formatting

② 3. XQuery – XPath + full-featured Q.L.

XLink, XPointer

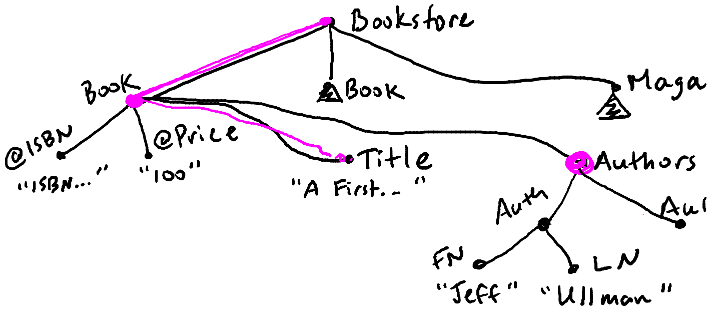
Jennifer Widom

Source: Jennifer Widom, Introduction to Databases, Stanford

XPath

XPath = *Path expressions* + *Conditions*

Think of XML as a tree



Basic Constructs

[N]

[3]

Jennifer Widom

Source: Jennifer Widom, Introduction to Databases, Stanford

XPath

XPath = *Path expressions* + *Conditions*

Built-in functions (lots of them)

Contains(s1, s2)
name()

Navigation "axes" (13 of them)

Parent::

following-sibling::

descendents:: //

self::

Jennifer Widom

Source: Jennifer Widom, Introduction to Databases, Stanford

XPath

More Details

XPath queries operate on & return *sequence of elements*

- XML document
- XML stream

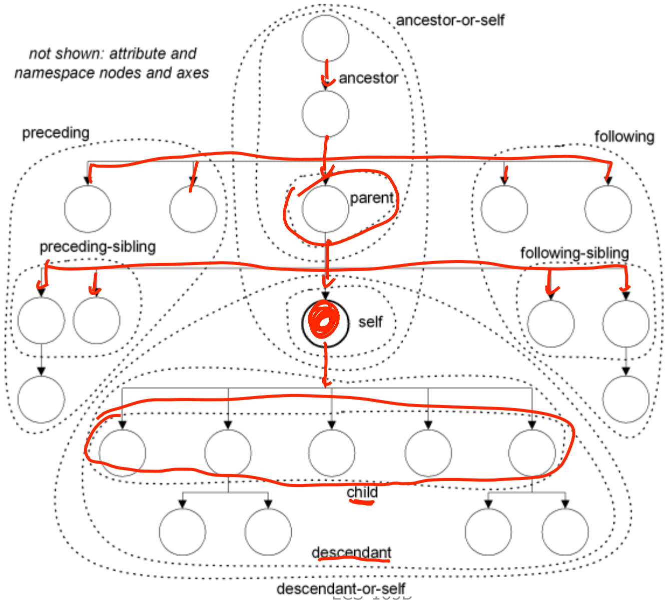
Sometimes result can be expressed as XML, not always

Demo: XPath examples over bookstore data

Source: Jennifer Widom, Introduction to Databases, Stanford

Navigating the XML Tree with XPath:

It's all about The Family! (parents, children, siblings, …)



ECS 165B 18