Querying XML

XPath

Querying XML

Not as mature as SQL

- Relatively new
- No underlying algebra (as in relational algebra)

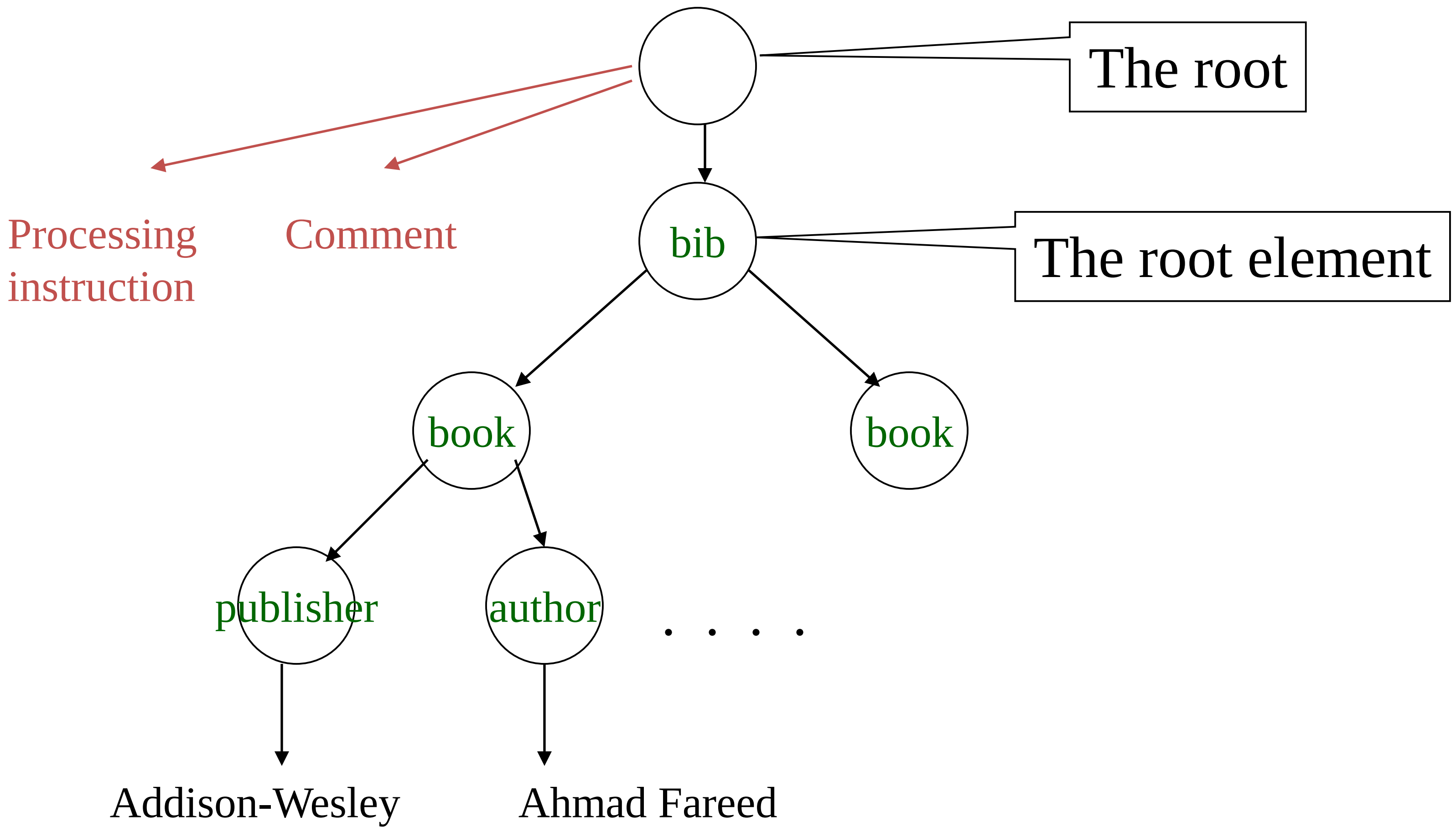
Related Topics (to be followed) development

1. XPath
2. XSLT
3. XQuery

XPATH

“The Basic Building Block”

Data Model



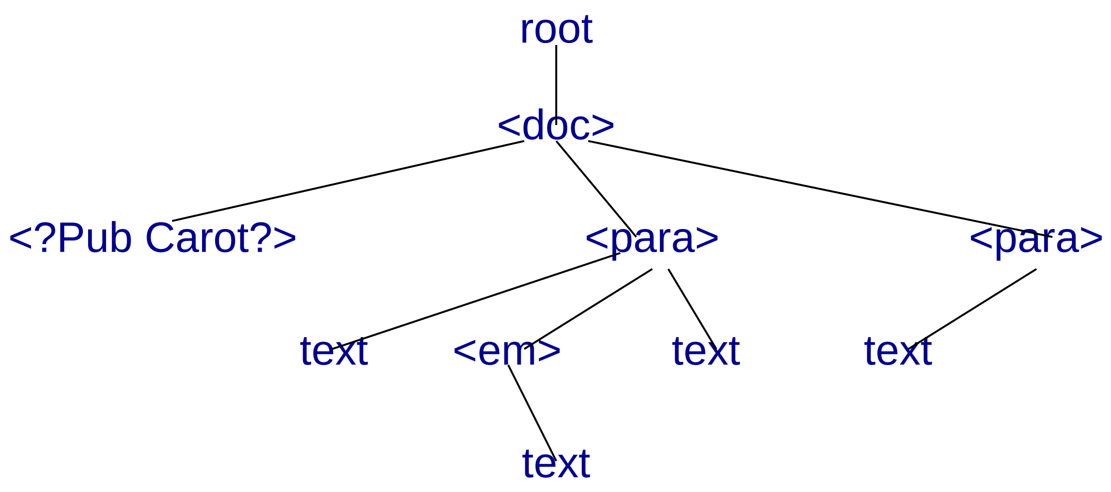
The root
Processing instruction
Comment
bib
The root element
book
book
publisher
author
. . . .
Addison-Wesley
Ahmad Fareed

Data Model Example 1

For this simple doc:

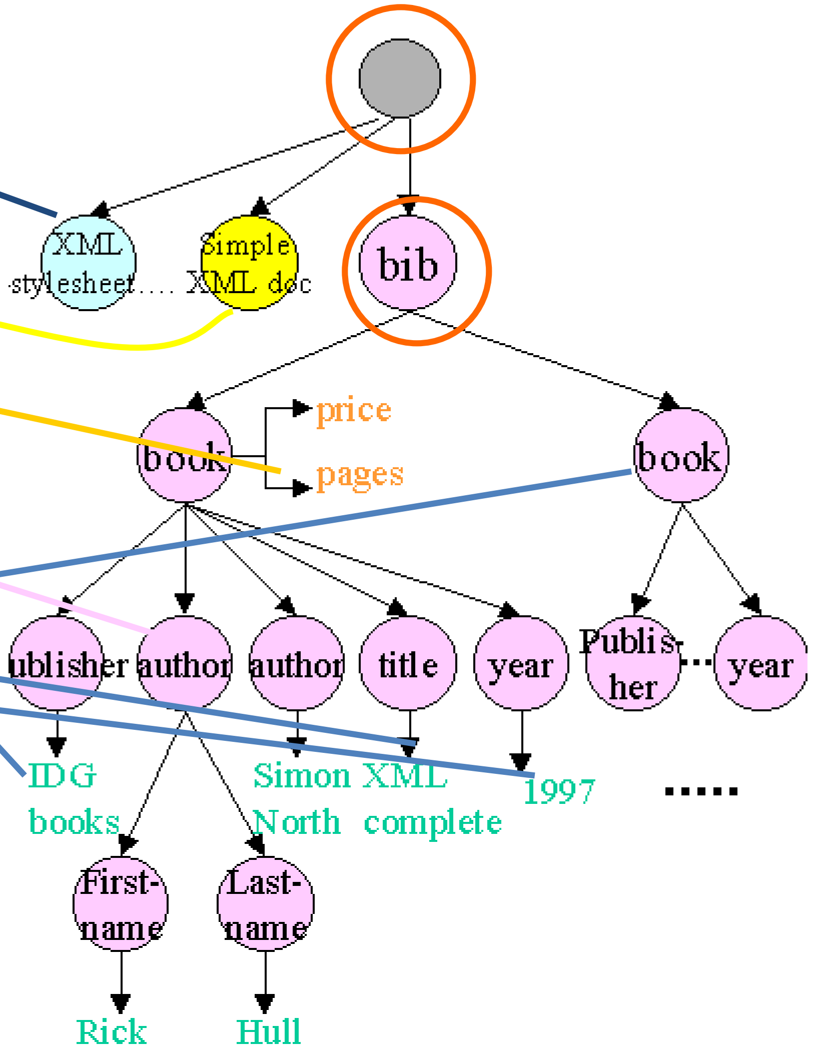
```
<doc>
<?Pub Carot?>
<para>Some <em>emphasis</em> here. </para>
<para>Some more stuff.</para>
</doc>
```

Might be represented as:



Data Model Example 2

```
<?xml version="1.0">
<?xml-stylesheet type="text/xsl" href="bib.xsl" ?>
<! -- simple XML document -->
<bib>
  <book price="25.00" pages="400">
           <publisher> IDG books</publisher>
         <author> <first-name>Rick</first-name>
                  <last-name> Hull </last-
  name>
         </author>
         <author> Simon North</author>
         <title> XML complete </title>
         <year> 1997 </year>
  </book>
  <book>
         <publisher> Freeman </publisher>
         <author> Jeffrey D. Ullman </author>
         <title> Principles of Database </title>
         <year> 1998 </year>
  </book>
</bib>
```



Element Context

- Meaning of element can depend upon its context
  - **`<book><title>…</title></book>`** **`<person><title>…</title></person>`**
- Want to search for, e.g. title of book, not title of person
  - XPath exploits sequential and hierarchical context of XML to specify elements by their context (i.e. location in hierarchy)
    - **`title    book/title    person/title`**

Context

- All XPath expressions are evaluated in the context of a particular node (location) in the document. That node is called the context node.

- The context size is the number of children of the context node's parent. For example, if the context node is one of seven children of its parent, the context size is seven.

- The context position is the child number of the context node relative to its parent. For example, if the context node is the third of seven children of its parent, its context position is three.

XPath

- Goal = Permit access some nodes from document

- XPath main construct : Axis navigation
- Navigation step : axis + node-test + predicates
- Examples
  - descendant::node()
  - child::author
  - attribute::booktitle =“XML”

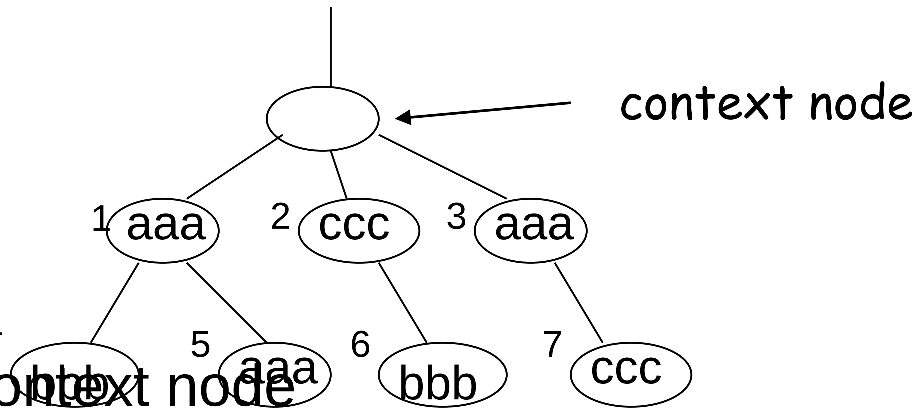
XPath

- XPath path consists of one or more navigation steps, separated by “/”
- Navigation step : axis + node-test + predicates
- Examples
  - /descendant::node() /child::author
  - /descendant::node() /child::author [parent /attribute::booktitle =“XML”][2]

- XPath offers shortcuts :
  - no axis means child
  - // ≡ /descendant-or-self::node()/

XPath- Child Axis Navigation

- author is shorthand for child::author.
- Examples:
  - aaa -- all the children nodes labeled aaa
  - aaa/bbb -- all the bbb grandchildren of aaa children
  - */bbb all the bbb grandchildren of any child



- Notes:
  - . -- the context node
  - / -- the root node

XPath- Child Axis Navigation

- /doc -- all doc children of the root
- ./aaa -- all aaa children of the context node (equivalent to aaa)
- text() -- all text children of context node
- node() -- all children of the context node (includes text and attribute nodes)
- .. -- parent of the context node
- .// -- the context node and all its descendants
- // -- the root node and all its descendants
- //text() -- all the text nodes in the document

Predicates

- [2] -- the second child node of the context node
- chapter[5] -- the fifth chapter child of context node
- [last()] -- the last child node of the context node
- chapter[title="introduction"] -- the chapter children of the context node that have one or more title children whose string-value is "introduction" (string-value is concatenation of all text on descendant text nodes)
- person[.//firstname = "joe"] -- the person children of the context node that have in their descendants a firstname element with string-value "Joe"

Axis navigation

- So far, our expressions have moved us *down* by moving to children nodes.

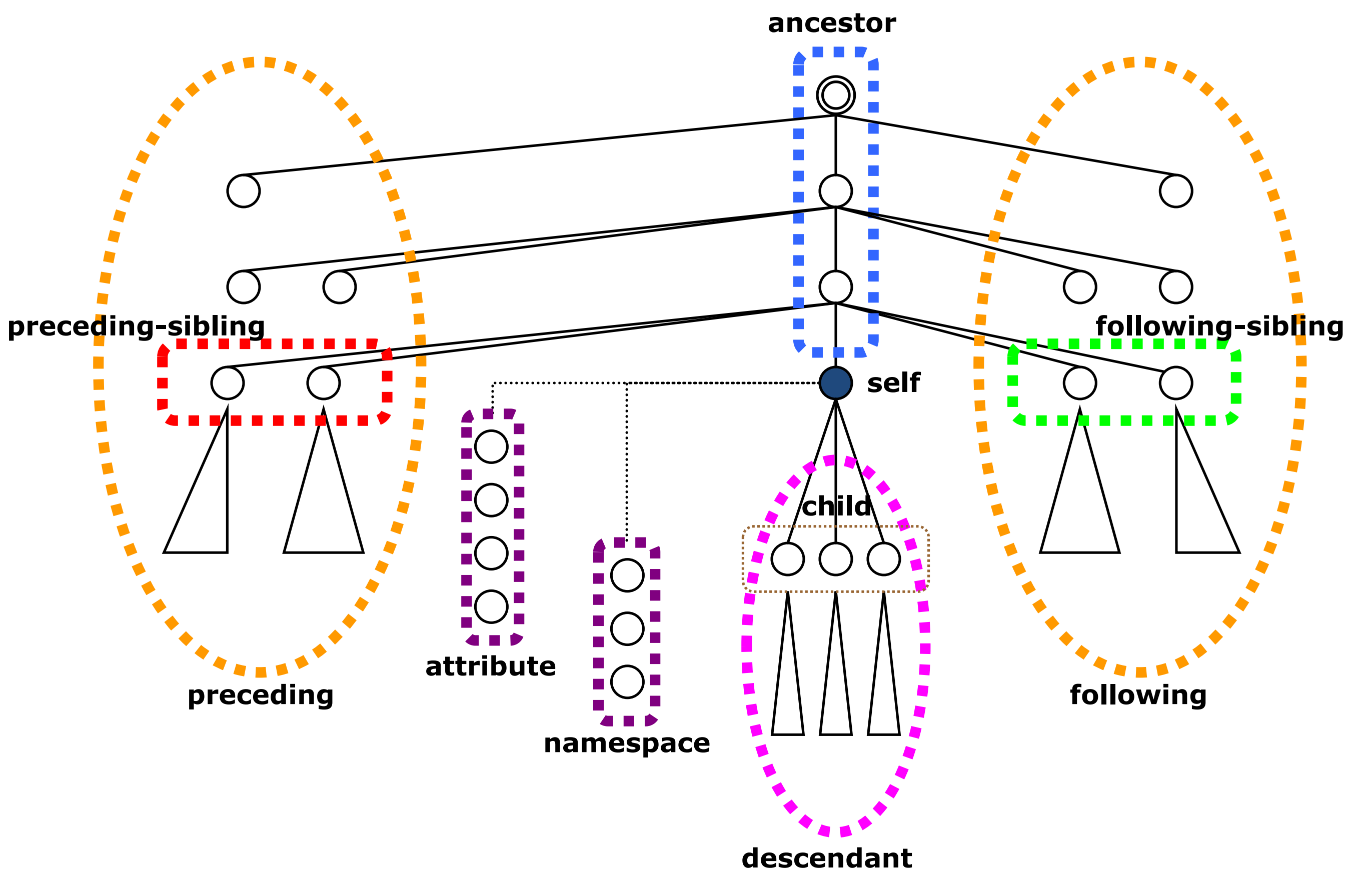
- Exceptions are :
  - .   stay where you are
  - /   go to the root
  - //   all descendants of the root
  - .//   all descendants of the context node

Axis navigation

- XPath has several axes: ancestor, ancestor-or-self, attribute, child, descendant, descendant-or-self, following, following-sibling, namespace, parent, preceding, preceding-sibling, self

- Some of these describe single nodes:
  - self, parent
- Some describe sequences of nodes:
  - All others

XPath Navigation Axes



XPath Abbreviated Syntax

(nothing)	child::
@	attribute::
//	/descendant-or-self::node()
.	self::node()
.//	descendant-or-self::node
..	parent::node()

(document root)

So Far

Differences between SQL and XPATH?

- What are similar query capabilities?
- What features does SQL have, but not XPATH?
- What features does XPATH support, but not SQL?
- Is XPath a full-fledged query language?

XPath examples I (1)

The following XML document example is the one used on the introduction to XML.

```
<?xml version="1.0" encoding="iso-8859-1"?> <pets>
<pet type="dog" color="brown">Max</pet>
<pet type="cat" color="white">Toula</pet> </pets>
```

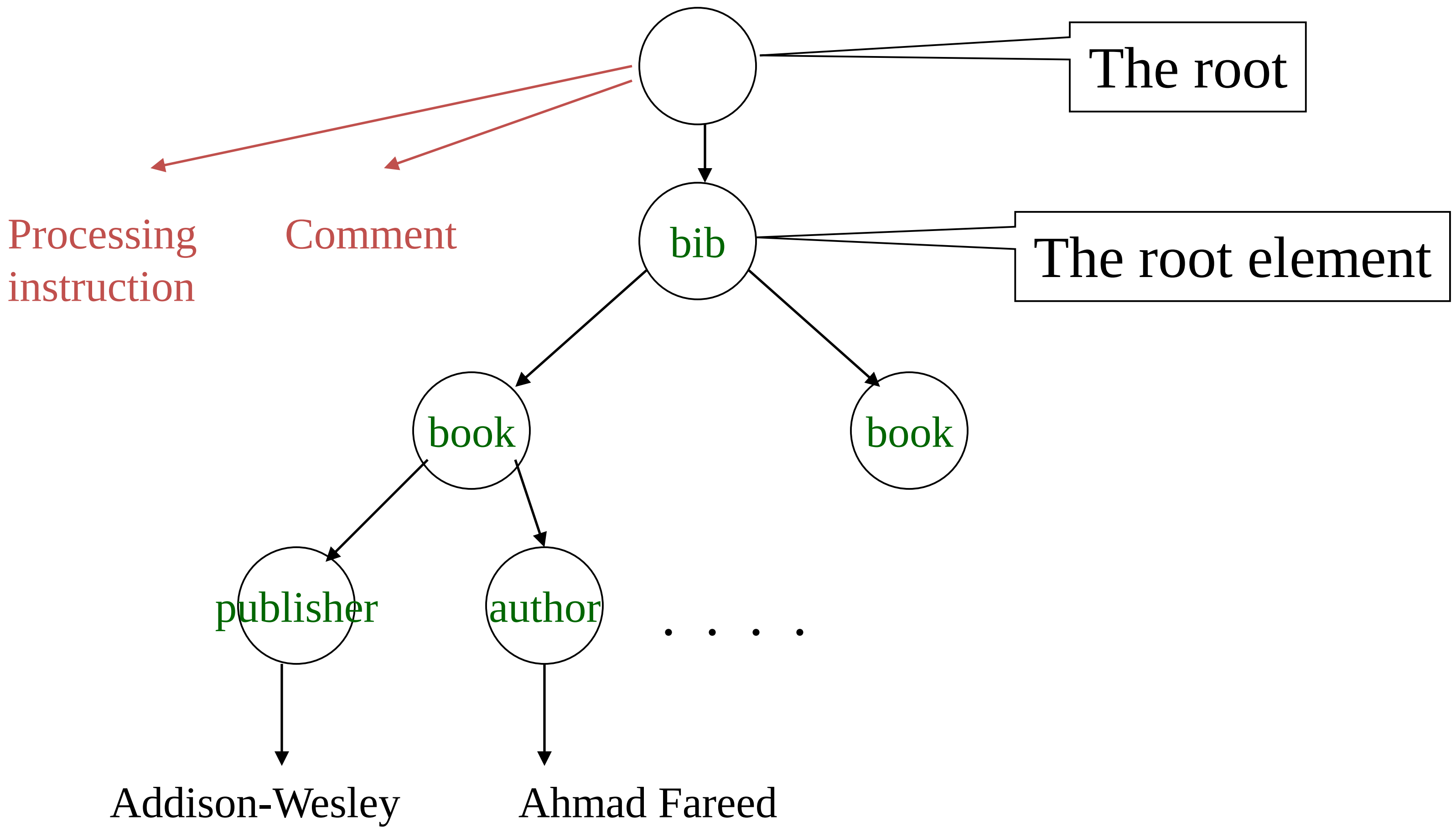
We will now demonstrate some very simple XPath examples using that document.

XPATH Tool

XPath examples I (2)

- ***Select all pet elements***
- **//pet or alternatively //pets/pet or //pets/child::***
- ***Select the first pet***
- **//pets/pet[1]**
- ***Select all pets of type dog***
- **//pet[@type ="dog"]**
- ***Select all pets of white color***
- **//pet[@color="white"]**
- ***Select the color of all dogs***
- **//pet[@type ="dog"]/@color**
- ***Get the types of pets with the name Max***
- **/pets/pet[text()="Max"]/@type**

Data Model

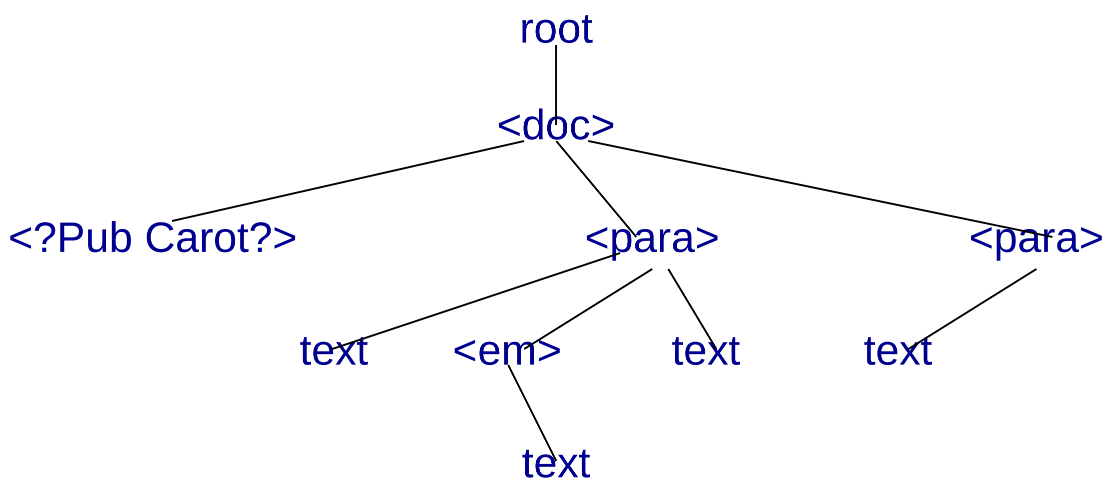

The root
Processing instruction
Comment
bib
The root element
book
book
publisher
author
. . . .
Addison-Wesley
Ahmad Fareed

Data Model Example 1

For this simple doc:

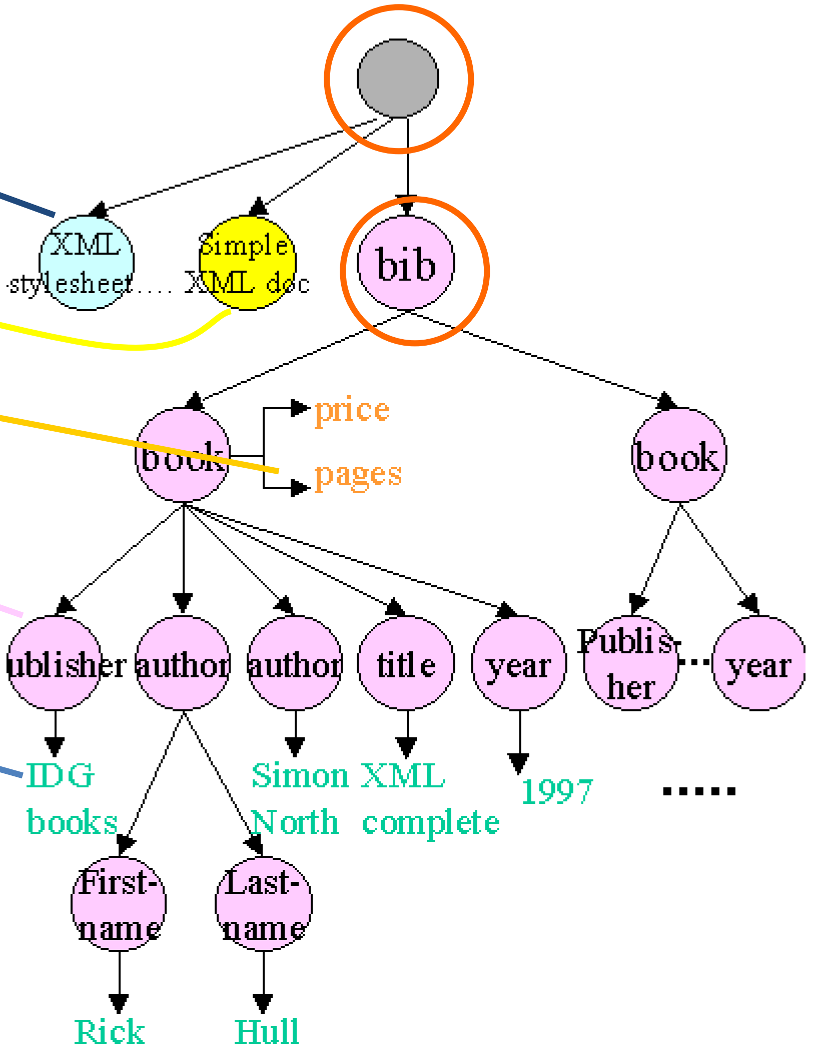
```
<doc>
<?Pub Carot?>
<para>Some <em>emphasis</em> here. </para>
<para>Some more stuff.</para>
</doc>
```

Might be represented as:



Data Model Example 2

```
<?xml version="1.0">
<?xml-stylesheet type="text/xsl" href="bib.xsl" ?>
<! -- simple XML document -->
<bib>
  <book price="25.00" pages="400">
            <publisher> IDG books</publisher>
          <author> <first-name>Rick</first-name>
                   <last-name> Hull </last-
  name>
          </author>
          <author> Simon North</author>
          <title> XML complete </title>
          <year> 1997 </year>
  </book>
  <book>
          <publisher> Freeman </publisher>
          <author> Jeffrey D. Ullman </author>
          <title> Principles of Database </title>
          <year> 1998 </year>
  </book>
</bib>
```



Element Context

- Meaning of element can depend upon its context
  - **`<book><title>…</title></book>`** **`<person><title>…</title></person>`**
- Want to search for, e.g. title of book, not title of person
  - XPath exploits sequential and hierarchical context of XML to specify elements by their context (i.e. location in hierarchy)
    - **`title      book/title      person/title`**

XML PATH EXAMPLES

```
<topic id="abc">
  <title>Using XPATH</title>
  <body>
    <p>Using XPATH is easy.</p>
    <fig>
      <image href="images/xpath.png"/>
    </fig>
    <section>
      <title>Examples</title>
      <p audience="novice">A simple example.</p
      <p audience="expert">An advanced example
      <p audience="expert">Another advanced exa
      <fig>
        <image href="images/xpath-axes.png">
          <alt>This screenshot shows the XPATH a
        </image>
      </fig>
    </section>
    <p>The End.</p>
  </body>
</topic>
```

/topic	Returns the <topic> root element.
//title	Returns any <title> element.
//section/title	Returns only the <title> element that is a child of a <section> element.
//p	Returns any <p> element.
//p[@audience='expert']	Returns any <p> element where the @audience attribute is set to expert.
//p[not(@audience)]	Returns any <p> element where the @audience attribute is missing.
//p[not(@audience='admin')]	Returns any <p> element where the @audience attribute is not of value admin OR is missing.
//p[text()='To start this process']	Returns any <p> elements that start with the text string To start this process.
//p[contains(.,'button')]	Returns any <p> element that contain the text string button somewhere in the text.
//image[not(alt)]	

```
<?xml version="1.0" encoding="utf-8" ?>
<Catalog>
   <Album artist="The Last Shadow Puppets" title="The
Age Of The Understatement">
      <Track rating="4" length="P3M7S">The Age Of The
Understatement</Track>
      <Track rating="3" length="P2M18S">Standing Next
To Me</Track>
      <Track rating="5" length="P2M26S">Calm Like
You</Track>
      <Track rating="3" length="P3M38S">Separate and
Ever Deadly</Track>
      <Track rating="2" length="P2M37S">The
Chamber</Track>
      <Track rating="3" length="P2M44S">Only The
Truth</Track>
   </Album>
   <Album artist="Kings Of Leon" title="Because Of The
Times">
      <Track rating="4" length="P7M10S">Knocked
Up</Track>
      <Track rating="2"
length="P2M57S">Charmer</Track>
      <Track rating="3" length="P3M21S">On Call</Track>
      <Track rating="4"
length="P3M09S">McFearless</Track>
      <Track rating="1" length="P3M59S">Black
Thumbnail</Track>
   </Album>
</Catalog>
```

/Catalog/Album

/Catalog/Album/@artist

/Catalog/Album/Track

**/Catalog/Album[@artist="Kings
Leon"]**

**/Catalog/Album[@artist="Kings
Leon"]/Track**

/Catalog/Album[2]

/Catalog/Album/Track[@rating>2

/Catalog/Album/Track/text()