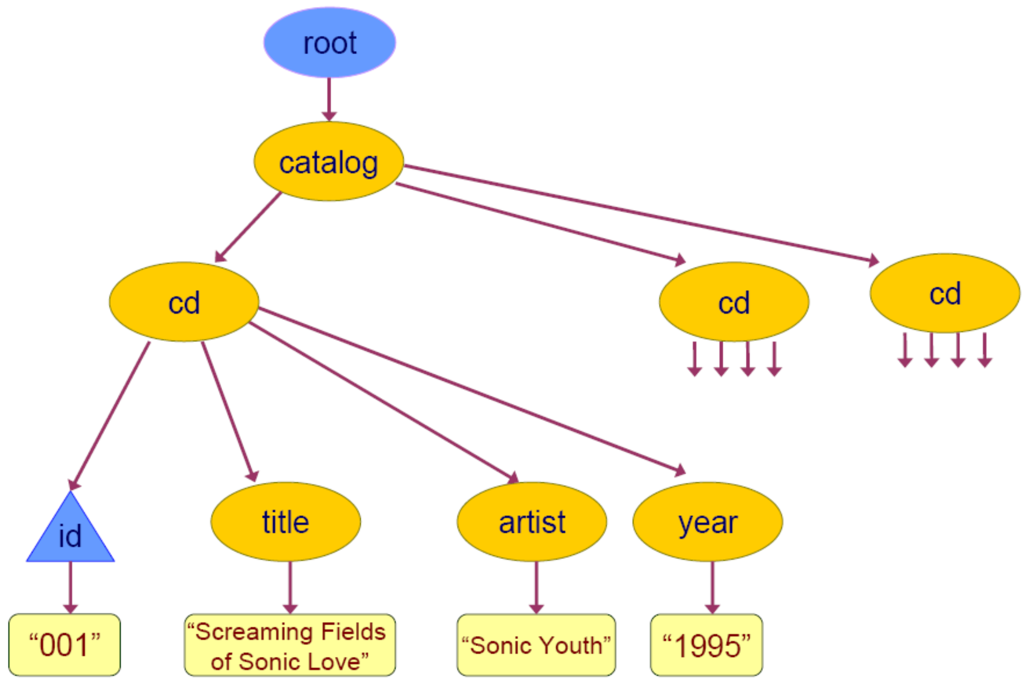
Xpath Examples

– In Xpath queries you can specify different kinds of nodes, including elements, attributes, text, processing instructions and comments. Some examples follow:

node() - any node	*text()* - a text node
element::foo - an element named foo	*foo* - an element named foo
attribute::foo - an attribute named foo	*@foo* - an attribute named foo
*@** - any attribute	* - any element
. - this element	.. - the parent element
/ - the root node	/* - the root element
//foo - an element foo at any level	

Xpath Examples

```
<?xml version="1.0" encoding="UTF-8"?>
<catalog>
<cd id="0001">
   <title>Screaming Fields of Sonic Love</title>
   <artist>Sonic Youth</artist>
   <year>1995</year>
</cd>
<cd id="0002">
    <title>Uh Huh Her</title>
   <artist>PJ Harvey</artist>
   <year>2004</year>
</cd>
<cd id="0003">
   <title>The Mirror Conspiracy</title>
   <artist>Thievery Corporation</artist>
   <year>2000</year>
</cd>
</catalog>
```



Expressão	**Acção**
cd	Selects all "cd" nodes (children of current node)
/	Selects the document root node
//cd	Selects all "cd" nodes in the document (independent of their location)
.	Selects current node
..	Selects parent of current node
@id	Selects current node "id" attribute

Xpath Examples

– Operators and functions that can be used in XPATH statements:

  +, -, *, div, mod, =, !=, <, <=, >, >=, or, and

  number(arg), abs(arg), floor(arg), round(arg), …, string(arg), compare(s1,s2), concat(s1,s2,...), contains(s1,s2), substring(s,start,len), normalize-space(), starts-with(s1,s2), …, name(), count(item1,item2,...), first(), last(), position(), …, dateTime(date,time), ...

- For processing of nodes with unknown name ...

*	matchs any element node
@*	matchs any attribute node
node()	matchs any node

– Some Xpath statement examples:

/catalog/cd[artist='Sonic Youth']	Selects all catalog cds with "Sonic Youth" artist
/catalog/cd[year]	Selects all catalog cds that have year data assigned
//cd[count(artist) = 1]	Returns all cds with only one artist
*/catalog/**	All catalog childreen elements
*//**	All catalog elements
//[@]*	All elements having at least one attribute

Xpath Examples

Some examples of location paths using the abbreviated syntax:

- *para* selects the para element children of the context node
- * selects all element children of the context node
- *text()* selects all text node children of the context node
- *@name* selects the name attribute of the context node
- *@** selects all the attributes of the context node
- *para[1]* selects the first para child of the context node
- *para[last()]* selects the last para child of the context node
- **/para* selects all para grandchildren of the context node
- */doc/chapter[5]/section[2]* selects the second section of the fifth chapter of the doc
- *chapter//para* selects para element descendants of the chapter element children of the context node
- *//para* selects all the para descendants of the document root and thus selects all para elements in the same document as the context node
- *//olist/item* selects all item elements in the same document as the context node that have an olist parent

Xpath Examples

Some examples of location paths using the <u>abbreviated syntax</u>:

- . selects the context node
- *.//para* selects the para element descendants of the context node
- .. selects the parent of the context node
- *../@lang* selects the lang attribute of the parent of the context node
- *para[@type="warning"]* selects all para children of the context node that have a type attribute with value warning
- *para[@type="warning"][5]* selects the fifth para child of the context node that has a type attribute with value warning
- *para[5][@type="warning"]* selects the fifth para child of the context node if that child has a type attribute with value warning
- *chapter[title="Introduction"]* selects the chapter children of the context node that have one or more title children with string-value equal to Introduction
- *chapter[title]* selects the chapter children of the context node that have one or more title children
- *employee[@secretary and @assistant]* selects all the employee children of the context node that have both a secretary attribute and an assistant attribute

Xpath Examples

XPATH Functions:

- *Node Set Functions*
  - *number* **last**() - returns a number equal to the context size
  - *number* **position**() - function returns a number equal to the context position
  - *number* **count**(*node-set*) - function returns the number of nodes in the argument node-set
  - …
- *String Functions*
  - *string* **string**(*object*?) - converts object into a string. A *node-set* is converted to a string-value of the 1st node in the node-set. A *number* is converted to a string, *booleans* into false/true strings.
  - *string* **concat**(*string*, *string*, *string**) - returns the concatenation of its arguments.
  - *boolean* **contains**(*string*, *string*) - returns true if the 1st argument contains the 2nd argument.
  - *string* **substring**(*string*, *number*, *number*?) - returns the substring 1st argument starting at the position specified in the 2nd argument with length specified in the 3rd argument.
  - *number* **string-length(*string*?)**
  - …

Xpath Examples

XPATH Functions:

- *Boolean Functions*
  - *boolean* **boolean**(*object*) - returns true iff it is neither positive or negative (number), non-empty (node-set), non-zero length (string)
  - *boolean* **not**(*boolean*) - returns true if its argument is false, and false otherwise.
  - *boolean* **lang**(*string*) - returns true if the language of the context node as specified by xml:lang attributes is the same as or is a sublanguage of the language specified by the argument string.
  - …
- *Number Functions*
  - *number* **number**(*object*?) - string is converted to number, boolean true is converted to 1, false to 0, node-set is converted to string and from string to number.
  - *number* **sum**(*node-set*) - returns the sum, for each node in the argument node-set, of the result of converting the string-values of the node to a number.
  - *number* **round**(*number*) - returns the number that is closest to the argument and that is an integer. If there are two such numbers, then the one that is closest to positive infinity.
  - …