XQuery

Plan

1. Introduction

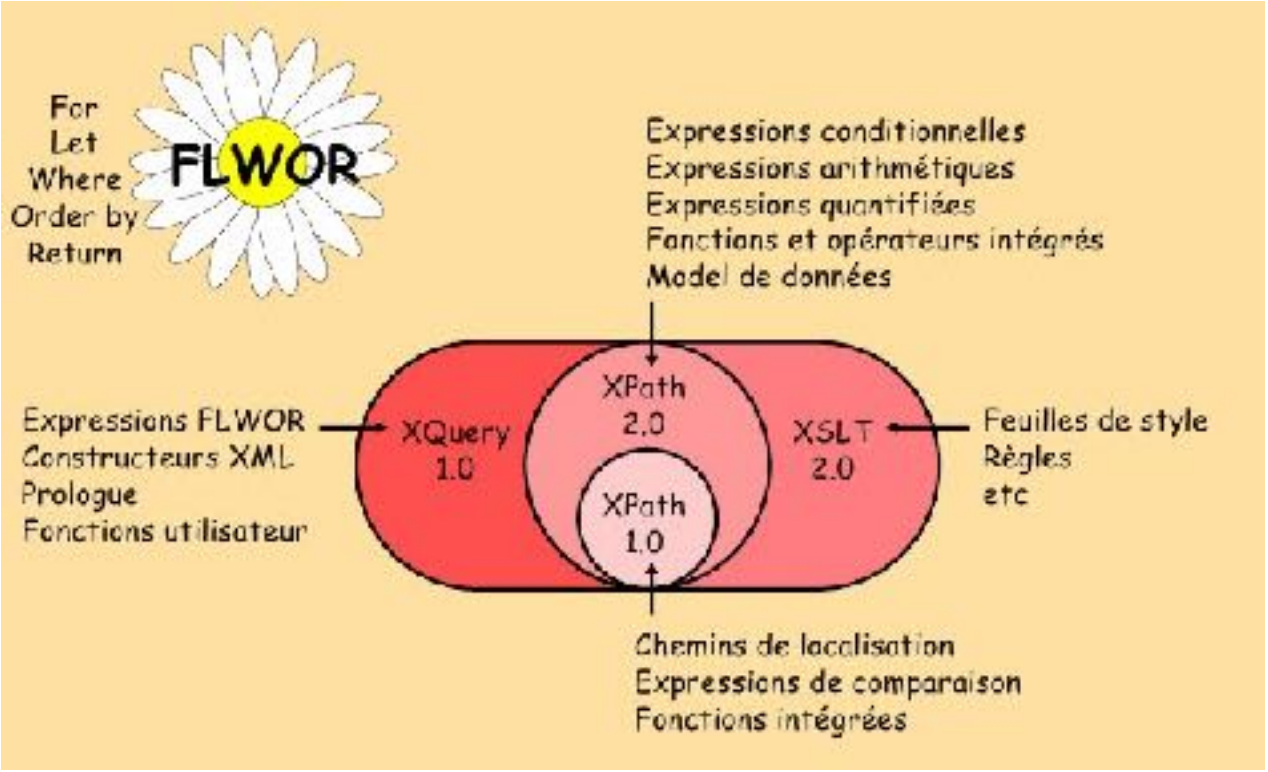
2. Modèle de données

3. Expressions

Introduction

- Langage d'interrogation de documents XML
- Recommandation du W3C du 23 janvier 2007 (http://www.w3.org/TR/xquery/)

Dernière recommandation : 23 octobre 2013 (version 3, http://www.w3.org/TR/xquery-30/)



Modèle de données

- Le modèle de données est semblable à celui de XPath

Il y a 7 sortes de nœuds (les éléments d'un documents XML):
Document Node, Element Node, Attribute Node,Text Node
Comment Node, Processing Instruction Node, Namespace Node

- Valeur atomique: Instance de type (String, Integer, date,...)
- Séquence : collection ordonnée d'Items
- Valeur typée: Séquence de valeurs atomiques
- Un **item** est un nœud ou une valeur atomique

Modèles de données: valeurs/séquence

·Pas de distinction entre un item et une séquence de longueur 1 :

·47 = (47)

·Une séquence peut contenir des valeurs hétérogènes :

(1, “toto”, <toto/>)

·Pas de séquences imbriqués :

(1, (2, 6), "toto", <toto/>) = (1, 2, 6, “toto”, <toto/>)

·Une séquence peut être vide : ()

Règles générales de XQuery

- XQuery est un langage sensible à la casse
  - Les mots clés sont en minuscules
- Chaque expression a une valeur
- Les expressions sont composables
- Les expressions peuvent générer des erreurs
- Les commentaires sont possibles

  (: un commentaire :)

Qu'est ce qu'une requête XQuery?

- Une requête est une expression qui
  - Lit une séquence de fragments XML ou de valeurs atomiques
  - Retourne une séquence de fragments XML ou de valeurs atomiques

- Les formes principales que peuvent prendre une expression XQuery sont :
  - Expressions de chemins
  - Constructeurs
  - Expression FLWOR
  - Expressions de listes
  - Conditions
  - Expressions quantifiées
  - Expressions de types de données
  - Fonctions

Types d'expressions

- Une requête XQuery est une composition d'expressions analogue à une expression Xpath.
- Une expression de chemin XPath est une requête XQuery
- Un chemin retourne un ensemble ordonné de nœuds d'un document ou une erreur

Exemple: document bib.xml

```
<bib>
<book year="2000" title="Comprendre XSLT">
        <author><la>Amann</la><fi>B.</fi></author>
        <author><la>Rigaux</la><fi>P.</fi></author>
        <publisher>O'Reilly</publisher>
        <price>28.95</price>
        </book>
<book year="2001" title="Spatial Databases">
        <author><la>Rigaux</la><fi>P.</fi></author>
        <author><la>Scholl</la><fi>M.</fi></author>
        <author><la>Voisard</la><fi>A.</fi></author>
        <publisher>Morgan Kaufmann Publishers</publisher>
        <price>35.00</price>
</book>
<book year="2000" title="Data on the Web">
        <author><la>Abiteboul</la><fi>S.</fi></author>
        <author><la>Buneman</la><fi>P.</fi></author>
        <author><la>Suciu</la><fi>D.</fi></author>
        <publisher>Morgan Kaufmann Publishers</publisher>
        <price>39.95</price>
</book></bib>
```

Exemple 1

- Requête XQuery définie dans le fichier ex1.xq

Afficher les auteurs des livres

doc("bib.xml")//book/author

Résultat
$ **fire-xquery --query=ex1.xq**

```
<author><la>Amann</la><fi>B.</fi></author>
<author><la>Rigaux</la><fi>P.</fi></author>
<author><la>Rigaux</la><fi>P.</fi></author>
<author><la>Scholl</la><fi>M.</fi></author>
<author><la>Voisard</la><fi>A.</fi></author>
<author><la>Abiteboul</la><fi>S.</fi></author>
<author><la>Buneman</la><fi>P.</fi></author>
<author><la>Suciu</la><fi>D.</fi></author>
```

- Expression de chemin avec un prédicat

doc("bib.xml")//book[@year="2001"]

```
<book year="2001" title="Spatial Databases">
<author><la>Rigaux</la><fi>P.</fi></author>
<author><la>Scholl</la><fi>M.</fi></author>
<author><la>Voisard</la><fi>A.</fi></author>
<publisher>Morgan Kaufmann Publishers</publisher>
<price>35.00</price>
</book>
```

Constructeurs

- Une expression XQuery peut construire un nouvel élément XML

```
<employee empid="12345">
  <name>John Doe</name>
    <job>XML specialist</job>    <deptno>187</deptno>
<salary>125000</salary>
</employee>
```

- Le résultat de cette requête est la requête elle même.

Exemple

Requête ex2_crs_3_s.xq

```
element employee{
    attribute empid {12345},
    element name {'John Doe'},
    element job {"XML specialist"},
    element deptno {187},
    element salary {125000}
}
```

```
fire-xquery --query=ex2_crs_3_s.xq
<?xml version="1.0" encoding="UTF-8"?>
<employee empid="12345">
   <name>John Doe</name>
   <job>XML specialist</job>
   <deptno>187</deptno>
   <salary>125000</salary>
</employee>
```

Constructeurs

- D'une manière générale, si on doit calculer la valeur d'un élément ou d'un attribut, on l'imbrique entre{…}.
- Dans les crochets, on met n'importe quelle expression XQuery qui retourne une valeur
- Si on doit calculer et le nom de la balise et son contenu, on utilise la notation suivante

```
element { name-expr } { content-expr }
attribute { name-expr } { content-expr }
```

FLWOR

FLWOR: For Let Where Order by Return

```
for $x in doc('bib.xml')//book

 where $x/price>30.0

 order by $x/price

 return element titre {$x/@title}
```

```
fire-xquery --query=ex4_crs.xq
<?xml version="1.0" encoding="UTF-8"?>
<titre title="Spatial Databases"/>
<titre title="Data on the Web"/>
```

- La clause *for* lie la variable $x avec chaque livre
- La clause *where* filtre
- La clause *order* trie
- La clause *return* renvoie le résultat

Utilisation de let

Les auteurs de livres parus en 2001

```
for $x in doc("bib.xml")//book
let $aux := $x/author
 where $x/@year="2001"
 return <ens nbAut="{count($aux)}">
{$aux/la }
 </ens>
```

Permet de générer un seul lien

```
<ens nbAut="3">
<la>Rigaux</la>
<la>Scholl</la>
<la>Voisard</la>
</ens>
```

if-then-else

```
<livres>
{ for $b in doc("bib.xml")//book
where $b/author/la = "Rigaux" return
if ($b/@year < 2001)
then <livre recent="false"> {$b/@title} </livre>
else <livre> {$b/@title} </livre> }
</livres>
```

```
<livres>
<livre recent="false" title="Comprendre XSLT"/>
<livre title="Spatial Databases"/>
</livres>
```

Fichier addr.xml

```
<addresses>
<person>
        <name>Amann</name>
        <country>France</country>
        <institution>CNAM</institution>
</person>
<person>
        <name>Scholl</name>
        <country>France</country>
        <institution>CNAM</institution>
</person>
<person>
         <name>Voisard</name>
         <country>Germany</country>
         <institution>FU Berlin</institution>
</person>
</addresses>
```

Jointures

```
for $b in doc("bib.xml")//book
      return element livre {
                attribute titre {$b/@title},
for $a in $b/author
     return element auteur {attribute nom {$a/la},
for $p in doc("addr.xml")//person
       where $a/la = $p/name
return attribute institut {$p/institution}
}
}
```

Création d'une balise livre

Création d'un attribut

titre

Jointure sur le nom

Résultat

```
<livre titre="Comprendre XSLT">
        <auteur nom="Amann" institut="CNAM"/>
        <auteur nom="Rigaux"/></livre>
<livre titre="Spatial Databases">
        <auteur nom="Rigaux"/>
        <auteur nom="Scholl" institut="CNAM"/>
        <auteur nom="Voisard" institut="FU Berlin"/></livre>
<livre titre="Data on the Web »>
        <auteur nom="Abiteboul"/>
        <auteur nom="Buneman"/>
        <auteur nom="Suciu"/></livre>
```