**XQuery**

Contenu

•Définitions

•Structure d’un document XQuery

•Les expression FLWOR

•Construction de fragment

•Syntaxe

•Fonctions et modules

Définitions

XQuery est langage de requête pour interroger les base de données XML

XQuery pour XML est comme SQL pour les bases de données relationnelles

•XQuery est basé sur les expressions Xpath

•XQuery est une recommandation W3C

Xquery= Xpath2,0 + autre

Exemples

**XQuery**peut être utilisé pour:

•Extraire des informations pour être utiliser dans un Web service.

•Générer des rapports de synthèse.

•Rechercher des documents Web pour obtenir des informations pertinentes.

**Les expression FLWOR**

•**FLWOR**est l’acronyme des cinq clauses **for**,**let**,**where**,**order by**et**return**.

•Elle est équivalente à l’idée de **select**-**from**-**where**-...-**order**-**by**de**SQL**.

**For**

•Itèresuruneséquenced'entréeetcalculeunevaleurpourchaqueélémentdecetteséquence

Exemple

**For** $t **in**//thème/titre

Retrun $t

**Let**

Permet d'assigner une valeur à une variable.

**Exemple**:

Let $titre := document("catalogue.xml")//book/title

**Where**

Permet de définir une condition de sélection pour faire des filtres sur le résultat.

Exemple

for $acteur in //acteur

where $video/genre = $genre

and$acteur/@id = $acteurRefs

return concat($genre, ": ", $acteur)

**Orderby**

Permet de trier les résultats

Exemple

for $acteur in //acteur

ODRER BY $acteur

return $acteur

**Return**

Construit l'expression à retourner à chaque itération.

Exemple

for $acteur in //acteurreturn$acteur

**Construction de fragments XML**

Le résultat d'une requête devrait normalement se présenter sous une forme appropriée pour être réutilisée.

**Version un seul fragment**

<resultat>

{ **for**$t **in**//topic/ titre/text()

**return**<title>{$t}</ title > }

</resultat>

<resultat>

< title > Catalogue </ title >

< title > Introduction </ title >

< title > developement</ title >

< title> analyse </title>

</resultat>

**Syntaxe**

**Règles de syntaxe de base XQuery**

Quelques règles de syntaxe de base:

•XQueryest sensible à la casse

•Une valeur de chaîne de caractères XQuerypeut être entre guillemets simples ou doubles

•Une variable XQueryest définie avec un $ suivi d'un nom

•Les commentaires XQuerysont délimités par (: et :)

**Syntaxe**

**Expressions conditionnelles**

XQueryprend en charge l'instruction**if-then-else**conditionnelle

**Exemple:**

**for**$x **in** doc("books.xml")/bookstore/book

**return if**($x/@category="CHILDREN")

**then**<child>{data($x/title)}</child>

**else**<adult>{data($x/title)}</adult>

•Les parenthèses autour de l'expression **if**sont obligatoires.

•**Else**est obligatoire, mais il peut être juste **else()**.

***INTRODUCTION***

XQueryUpdateFacility1.0estunepetiteextensionrelativeàXQuery.

PermetdemodifierdesdonnéesXMLnatives.

Introduitunenouvellecatégoried'expressionsappeléesexpressionsdemise-à-jour,permettantdemodifierl'étatd'unnœudXMLexistant.

***Syntaxede XQueryUpdate Facility***

XQueryUpdateFacilityestuneextensiondeXQuery.

UnlangagederequêteXMLstandardiséparleW3C.

CetteextensionutilisedesexpressionsXquerypourretrouverlesnoeudsXML,etd'autresexpressionspourmodifiercesnoeuds.

PossibilitédemettreàjourlesfichiersXMLeux-mêmes(DanslecasoùlabasededonnéesXMLestunecollectiondedocumentsXML,commelecasduserveurBaseXquienregistrelesdonnéesXMLsousformededocuments).

***Suppression –delete:***

Cettefonctiondemise-à-jourpermetdesupprimertouslessous-arbesXML(partiesdudocumentoriginal)enracinéspardesnoeudsXML“**product**”ayantunattribut"**id**"aveclavaleur"p1".

**deletenodedoc("products.xml")/products/product[@id = 'p1']**

***Insertion –insert***

**Insertnode**

**<productid="p7">**

**<name>papa</name>**

**<price>2100</price>**

**<stock>4</stock>**

**<country>China</country>**

**</product>**

**beforedoc("products.xml")/products/product[1]**

**<!--before/afterou "as first/last into" -->**

L'expressionXQueryci-dessuspermetd'insérerunnouvelélémentXMLdetype**product**avec**id="p7"**commeétantlepremierélémentdetouslesproduitsexistants.Notez,commeleditlecommentaire,aulieude«before»,nouspouvonsutiliser«after»ou«asfirst»ou«lastinto».

***Replacement (noeudouvaleur)-replace***

**replace value of node doc ("products.xml")/products/product[2]/name with "Romeo"**

**replace node doc("products.xml")/products/product[3]/name with <NAME>test</NAME>**

**<!--Notez que nous pouvons remplacer la valeur ou le nœud, Notez également que nous avons besoin d'un "," entre les deux mises à jour -->**

Ci-dessusnousavonsdeuxmisesàjourmontrant1)commentremplacerunevaleurd'unnœudXMLet2)commentremplacertoutunnœudXMLparunautre.

***Renommage-rename***

**Rename node doc("products.xml")/products/product[1] as "PRODUCT"**

**<!—renommer plusieurs nœuds -->**

**for $x in doc("products.xml")//(\*|@\*)**

**return**

**rename node $x**

**as upper-case (name($x))**

Lapremièrelignerenommelepremierélémentduproduitenmajuscule,etladeuxièmepourrenommertouslesnomsd'élémentsettouslesnomsd'attributsenmajuscules.

***l'équivalenceentre certainesopérations***

Le XQuery suivantpeutêtreremplacépar:

**replace node doc("products.xml")/products/product[3]/name with <NAME>test</NAME>**

1-une opérationdelete:

**delete node doc("products.xml")/products/product[3]/name**

2-une opérationinsert:

**insert node**

**<NAME >test</NAME>**

**After doc("products.xml")/products/product[2]**

Le XQuery suivantpeutêtreremplacépar:

**Rename node doc ("products.xml")/products/product[3]/name as**

**"NAME"**

1-une opérationdelete:

**deletenodedoc("products.xml")/products/product[3]/name**

2-une opérationinsert:

**insert node**

**<NAME >test</NAME>**

**After doc("products.xml")/products/product[2]**

***Fonctionsde mise-à-jour***

Permettentdeprendreenentréequelquesparamètres(desvariablesquireprésententdesnoeudsXMLparexemple)eteffectuerdesmises-à-joursurcesnoeudsXML,présentésparlesargumentsdelafonction,ousurleurdescendants/ancêtres.

Lafonctiondemise-à-journedoitpasavoirunevaleurderetouretl'argumentpasséàlafonctionnepeutpasêtreuneautrerequêtedemiseàjour.

***Fonctionsde mise-à-jour***

**Declare updating function**

**local:renameNode($elemas element(),**

**$repas xs:string)**

**{ renamenode$elemas $rep**

**};**

**local:renameNode(doc("dbxml:/con.dbxml/mydoc.xml")/a/b1,"aab1")**

L’appel dans le document XML

renamenode$elemas $rep

**<a> <b1>first child</b1>**

**<b2> second child</b2>**

**<b3>thirdchild</b3>**

**</a>**

Alors ce document devient:

**<a>**

**<aab1>first child</aab1>**

**<b2>second child</b2>**

**<b3>thirdchild</b3>**

**</a>**