XQUERY
Prezentare

1. Despre XQuery

XQuery este o recomandare W3C si a fost creat pentru a permite interogarea documentelor XML si extragerea unor componente ale acestora (elemente sau atribute). Cea mai buna definire a XQuery este obtinuta prin analogia cu SQL. Astfel, XQuery este pentru XML ceea ce este SQL pentru bazele de date. Acest limbaj se bazeaza pe XPath, cele doua limbaje folosind acelasi model al datelor si suportand aceiasi operatori si aceleasi functii. XQuery este suportat de catre toate motoarele mari ce actioneaza pe baze de date (IBM, Oracle, Microsoft, etc.)

2. Sintaxa

Cateva notiuni de baza ale sintaxei:

- XQuery este un limbaj case-sensitive
- Elementele, atributele si variabilele XQuery trebuiesc sa fie nume XML valide
- Valorile XQuery trebuiesc puse fie intre ghilimele, fie intre apostrofuri
- O variabila XQuery se defineste folosind $NUME, unde NUME este numele variabilei respective (ex: $doc)
- Comentariile XQuery sunt delimitate de (: si de :) (ex: (: XQuery Comentariu :)

Expresiile XQuery au o structura de FLWOR, care este un acronim pentru „For, Let, Where, Order by, Return”. Aceasta structura extrage niste noduri din fisierul XML, aplica un predicat pentru a elimina din nodurile selectate, iar apoi construieste un rezultat.

Instructiunile FOR si LET genereaza o lista de noduri extrase din document, mentinand ordinea acestora. Aceasta instructiune este similara cu FROM dintr-un query SQL. Diferenta dintre cele doua instructiuni este faptul ca in cazul instructiunii FOR, variabila var se leaga pe rand la fiecare element din expresia expr, facand posibila iterarea, in timp ce in cazul instructiunii LET, variabila var se leaga la intreaga lista definita de expresia expr, returnand o singura valoare.
Observatie: Instructiunea FOR se foloseste in forma *FOR $x **in** expr*, in timp ce instructiunea LET se foloseste in forma *LET $x = expr*.

Instructiunea ORDER BY defineste ordinea in care sunt considerate elementele extrase de FOR sau LET in cazul in care nu se doreste ordinea in care au fost acestea in documentul XML.

Instructiunea WHERE aplica un predicat (una sau mai multe restrictii) eliminand unele din nodurile extrase de catre FOR sau LET. Aceasta instructiune este echivalentul expresiei WHERE dintr-un query SQL.

Instructiunea RETURN este aplicata asupra fiecarui nod care a supravietuit conditiei/conditiilor impuse in WHERE, generand o lista ordonata de noduri la iesire. Aceasta instructiune este echivalentul expresiei RETURN dintr-un query SQL.

3. Cazuri de utilizare

Pentru a simplifica intelegerea sintaxei XQuery, vom folosi din nou documentul XML dat ca exemplu in documentul de prezentare a XPath. Vom presupune ca numele acestui document este facultate.xml.

```
<?xml version=„1.0” encoding=„ISO-8859-1”?>
<grupa id=„324CB”>
    <studenti>
        <student id=„1212”>
            <nume>Popescu</nume>
            <prenume>Ion</prenume>
        </student>
        <student id=„1213”>
            <nume>Ionescu</nume>
            <prenume>Dan</prenume>
        </student>
    </studenti>
    <advisor id=„2323”>
        <nume>Georgescu</nume>
        <prenume>Alex</prenume>
    </advisor>
</grupa>
```

Pentru a putea deschide documentul asupra caruia sa se aplice expresiile XQuery se foloseste **doc("nume_document.xml")**. Aceasta instructiune permite mai departe referirea oricarui element din documentul respectiv prin expresii ce respecta sintaxa Xpath.

Exemple de utilizare a expresiilor XQuery:

- In cazul in care dorim sa referim valoarea campului nume al nodului student care are valoarea atributului id „1213”, putem sa scriem o expresie XPath de forma: doc(„facultate.xml”)//student[@id=„1213”]/nume. Folosind sintaxa FLWOR XQuery

  for $x **in doc**(„facultate.xml”)//student
  where $x/@id = 1213
  return $x/nume

  se obtine rezultatul: <nume>Ionescu</nume>.
  Pentru o mai buna intelegere, in continuare se va explica pe scurt cum lucreaza expresia de mai sus. Initial, prin instructiunea FOR (for $x in doc(„facultate.xml”)),

variabila x (definita prin $x) se leaga la **fiecare** din nodurile de tip student pe care le gaseste in document //student) si se obtin doua noduri studenti – cel cu id=„1212” si cel cu id=„1213”). Apoi, aceste noduri sunt filtrate cu ajutorul conditiei din WHERE, din cele doua noduri ramanand numai cel cu id=„1213”($x este leagata la cel de-al doilea nod student). Acest nod este trimis mai departe catre instructiunea RETURN pentru a se genera iesirea. Aici, se extrage nodul nume care este copilul nodului student care a supravietuit conditiei din WHERE si se afiseaza, astfel obtinandu-se rezultatul <nume>Ionescu</nume>.

- Rezultatele care se obtin in urma aplicarii unei expresii XQuery pot fi **sortate** in functie de unul din elemente:

  for $x **in doc**(„facultate.xml”)//student
  where $x/@id >1000
  order by $x/@id
  return $x/nume

  se obtine rezultatul: <nume>Popescu</nume><nume>Ionescu</nume>.

- In cazul in care nu se doreste ca expresia XQuery sa intoarca un nod, ci se doreste sa se intoarca un text, aceasta se poate realiza folosind instructiunea **text()**:

  for $x **in doc**(„facultate.xml”)//student
  where $x/@id = 1213
  return $x/nume/text()

  Rezultatul acestui query va fi: Ionescu

- Daca se doreste ca rezultatul obtinut sa fie un **document valid XML sau HTML**, aceasta se poate realiza incadrand continutul expresiei XQuery in elementul care se doreste, punand continutul acesteia intre acolade ({}):

  ```
  <?xml version=„1.0” encoding=„ISO-8859-1”?>
  <studenti>
  {
      for $x in doc(„facultate.xml”)//student
      where $x/@id >1000
      order by $x/@id
      return $x/nume
  }
  <studenti>
  ```

  Scriptul de mai sus va avea ca efect crearea unui document XML valid care va arata in felul urmator:

  ```
  <?xml version=„1.0” encoding=„ISO-8859-1”?>
  <studenti>
      <nume>Popescu</nume><nume>Ionescu</nume>
  <studenti>
  ```

- Daca se doreste modificarea numelor sau tipurilor tagurilor, se poate folosi o instructiune similara cu:

```
<html>
<body>
<h1> Studenti </h1>
{
    for $x in doc(„facultate.xml”)//student
    where $x/@id >1000
    order by $x/@id
    return <p ID = „{$x/@id }”>{$x/nume/text()}</p>
}
</body>
</html>
```

Rezultatul acestui query va fi:

```
<html>
<body>
<h1> Studenti </h1>
<p ID =”1212”>Popescu</p>
<p ID =”1213”>Ionescu</p>
</body>
</html>
```

Observati faptul ca **elementele ce se doresc a fi procesate sunt incluse intre acolade** ({$x/@id }, {$x/nume/text()}).

- Pentru a intelege **diferenta esentiala dintre FOR si LET**, urmariti urmatoarele doua query-uri si rezultatele acestora

Expresie XQuery	**Rezultat**
for $x **in doc**(„facultate.xml”)//student **where** $x/@id > 1000 **return** <out>$x/nume</out>	<out><nume>Popescu</nume></out> <out><nume>Ionescu</nume></out>
let $x **= doc**(„facultate.xml”)//student **where** $x/@id > 1000 **return** <out>$x/nume</out>	<out> <nume>Popescu</nume> <nume>Ionescu</nume> </out>

- In cadrul expresiilor XQuery se pot folosi **expresii conditionale** de tipul **„if-then-else”.**

```
for $x in doc(„facultate.xml”)//student
where $x/@id >1000
return if ($x/@id >1212)
then <ionescu>{$x/prenume/text()}</ionescu>
else < popescu >{$x/prenume/text()}</popescu >
```

Rezultatul acestui query este: <ionescu>Dan</ionescu> < popescu >Ion</popescu >.

OBSERVATIE: Expresia continuta in IF trebuie pusa intre paranteze rotunde (). Else trebuie sa fie prezenta, dar poate sa fie vida (else ()).

- **Comparatiile in XQuery pot fi de doua feluri:**
  - **Generale:** =, !=, <, <=, >, >= → expresia intoarce adevarat relatia este respectata de catre cel putin un element
  - **Comparari de valori:** eq, ne, lt, le, gt, ge → expresia intoarce adevarat daca este un singur element care este comparat cu o valoare si daca relatia este respectata. Daca sunt mai mule elemente care se doresc a fi comparate, se intoarce eroare

Expresie XQuery	**Rezultat**
doc(„facultate.xml")//student/@id> 1000	**Adevarat** (sunt doi studenti ai caror atribute id au valoare mai mare ca 1000)
doc(„facultate.xml")//student/@id> 1212	**Fals** (sunt tot doi studenti, dar unul din cei doi are vlaoarea atributului egala cu 1212, deci conditia nu este respectata)
doc(„facultate.xml")//student/@id gt 1000	**EROARE** (sunt doi studenti si datorita acestui lucru se intoarce eroare)

4. Tooluri

Exista diferite instrumente care permit folosirea sintaxei XQuery. Printre acestea amintim Altova XMLSpy si StylusStudio (care au versiuni de trial) dar si QUIP (care are un GUI usor de utilizat dar care nu a mai fost updatat din 2002 – are nevoie de java 1.3 pentru a rula GUI). La adresa http://www.w3.org/XML/Query/#products gasiti o lista mai cuprinzatoare a produselor ce permit utilizarea XQuery.

5. Concluzii

XQuery este un limbaj ce poate fi utilizat pentru extragerea de informatii din documentele XML, generarea de rapoarte, transformarea documentelor XML in XHTML si cautarea in documentele Web a unor informatii relevante. XQuery este compatibil cu diverse standarde W3C, cum ar fi XML, Namespaces, XSLT, XPath, si XML Schema.

6. Bibliografie

- http://www.w3.org/TR/xquery/
- http://www.w3.org/TR/query-semantics/
- http://www.w3.org/TR/xmlquery-use-cases
- http://xml.apache.org/xalan-j/xpath_apis.html