Ecole d'ingénieurs et d'architectes de Fribourg
Hochschule für Technik und Architektur Freiburg
University of Applied Sciences of Fribourg

Travail pratique
XQuery - 2

Houda CHABBI DRISSI

1 Introduction

Les données structurées au format XML sont très souvent utilisées de différentes manières dans les applications actuelles. On en retrouve très fréquemment dans les fichiers de configurations ou alors dans les bases de données afin de stocker directement l'information de manière structurée. Néanmoins, la structuration des données ne serait pas si importante sans un outil performant pour la recherche et l'interrogation de ces données. Le monde relationnel avait le SQL. Le monde XML possède lui différentes alternatives dont XQuery.

2 Objectifs

Les objectifs pour cette deuxième partie du travail pratique sur XQuery sont les suivants :

- Comprendre les similitudes et les différences entre SQL et XQuery.
- Comprendre les similitudes et les différences entre XSLT et XQuery.
- Comprendre l'utilisation des namespace avec XQuery.

3 Travail à réaliser

3.1 SQL vs XQuery

Comme mentionné dans l'introduction, le langage XQuery est en quelque sorte l'équivalent du SQL dans le monde du XML. Il n'est donc pas étonnant de retrouver de nombreuses similitudes entre ces deux langages. Par exemple, on retrouve plus ou moins les mêmes mots-clefs dans la structure des requêtes :

SQL	XQuery
SELECT	RETURN
FROM	FOR…IN…
WHERE	WHERE
ORDER BY	ORDER BY
SET	LET
…	…

En plus de ces mots-clefs, XQuery implémentes de nombreuse fonctions se trouvant déjà dans SQL. Par exemple, XQuery implémente des fonctions d'agrégation comme *min()*, *max()*, *avg()*… ou encore, des fonction telles que *distinct-values()*, l'équivalent du « SELECT DISTINCT » en SQL, permettant d'éliminer les doublons dans un résultat.
Cet exercice va donc vous permettre de réécrire l'équivalent d'une requête SQL existante en XQuery.
Pour cela, nous allons utiliser le cas de la base de données ci-dessous permettant de gérer *des clients* et *leurs commandes* :

Table *Customers*

customerID	Name	address	zipCode
0001	John Doe	126, Baker St.	04563-300
0002	Tom Jones	33, Plaza Av.	14345-650
0003	Jack Smith	450, Ocean Drive	01232-200

Table *Products*

productID	description	price
BIKE-12	12 speed race bike	145.00
BIKE-18	18 speed race bike	176.00
MONO	Monocycle	65.00
CROSS	Cross-country bike	130.00

Table *Invoices*

invoiceNumber	Date	customerID
2145	2014-11-01	0002
2146	2014-11-02	0001
2147	2014-11-02	0003

Table *InvoiceItems*

invoiceNumber	productID	qty	salePrice	discount
2145	MONO	1	65.00	0.00
2145	BIKE-12	2	290.00	0.00
2146	BIKE-18	1	170.00	6.00
2147	MONO	3	180.00	15.00

Cette base de données peut également être exprimée sous forme XML. Un exemple de fichier XML équivalent à cette base se trouve dans le répertoire « Galax » de ce TP (invoice.xml).

La requête ci-dessous permet d'afficher le nom des clients ayant commandés le produit « MONO » ainsi que la date à laquelle cette commande a été effectuée :

```
SELECT Invoices.date, Customers.name
      FROM Customers, Invoices, InvoiceItems
      WHERE Customers.customerID = Invoices.customerID AND
            Invoices.invoiceNumber = InvoiceItems.invoiceNumber AND
            InvoiceItems.productID = 'MONO'
```

En voici son résultat :

date	Name
2014-11-01	Tom Jones
2014-11-02	Jack Smith

Ecrivez maintenant l'équivalent de la requête SQL ci-dessus à l'aide d'une expression FLWOR XQuery. Cette requête portera sur le fichier « invoice.xml ».

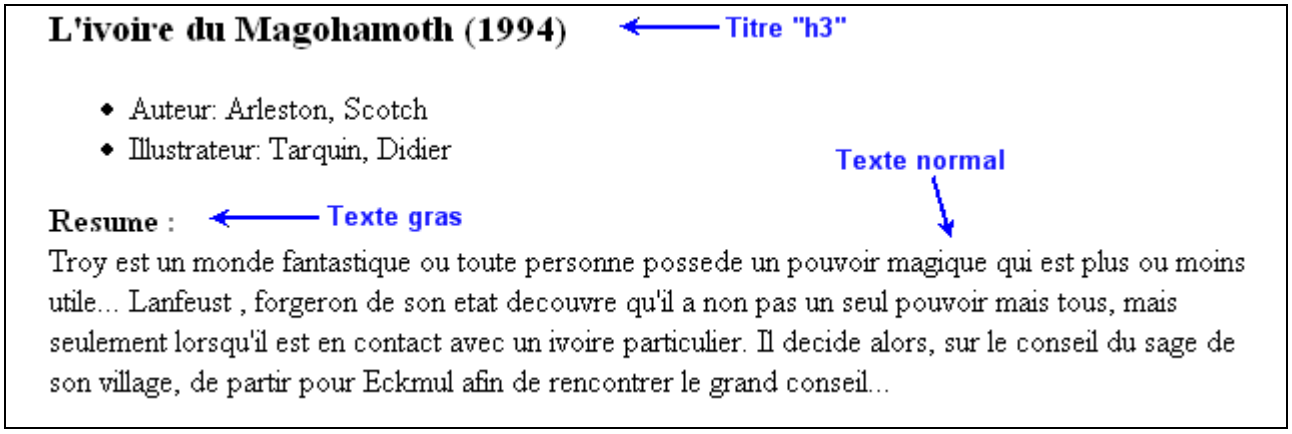
3.2 XSLT vs XQuery

Comme nous avons pu le voir durant les exercices précédents, XQuery permet d'interroger une source de données au format XML et d'en présenter les résultats en

utilisant un format personnalisé. XQuery présente donc de grandes similitudes avec le langage de transformation XML : XSLT !

Afin de démontrer cette similitude, cet exercice va nous permettre d'écrire une requête XQuery dont le résultat sera affiché dans un fichier HTML.

Veuillez écrire une requête XQuery permettant d'afficher la liste des DB (contenue dans le fichier « bd.xml ») sur une page HTML avec leurs auteurs et illustrateurs respectifs ainsi que leur résumé. Le croquis ci-dessous illustre la manière avec laquelle chaque DB devra être présentée dans la liste.



4 XQuery et les namespaces

Soit le document suivant décrivant une commande :

```
<o:shiporder xmlns:o="http://www.eif.ch/order">
      <o:orderperson>John Smith</o:orderperson>
      <o:shipto>
            <o:name>Ola Nordmann</o:name>
            <o:address>Langgt 23</o:address>
            <o:city>4000 Stavanger</o:city>
            <o:country>Norway</o:country>
      </o:shipto>
      <o:item>
            <o:title>Empire Burlesque</o:title>
            <o:note>Special Edition</o:note>
            <o:quantity>1</o:quantity>
            <o:price>10.90</o:price>
      </o:item>
      <o:item>
            <o:title>Hide your heart</o:title>
            <o:quantity>1</o:quantity>
            <o:price>9.90</o:price>
      </o:item>
</o:shiporder>
```

Tous les éléments de ce document XML sont définis dans le namespace « http://www.eif.ch/order ». Ce namespace est référencé à l'aide du préfix « o: » dans le document.
Quelqu'un a essayé d'écrire la requête XQuery suivante pour retrouver tous les « items » de la commande. Voici la requête qu'il a écrite :

```
for
      $i in document("namespace.xml")//o:item
return
      $i/o:title
```

Malheureusement, c'est requête ne fonctionne pas. Déterminez donc la cause du problème de cette requête et corrigez-la.

5 Réféfences

[1] Spécifications XQuery du W3C
http://www.w3.org/XML/Query/