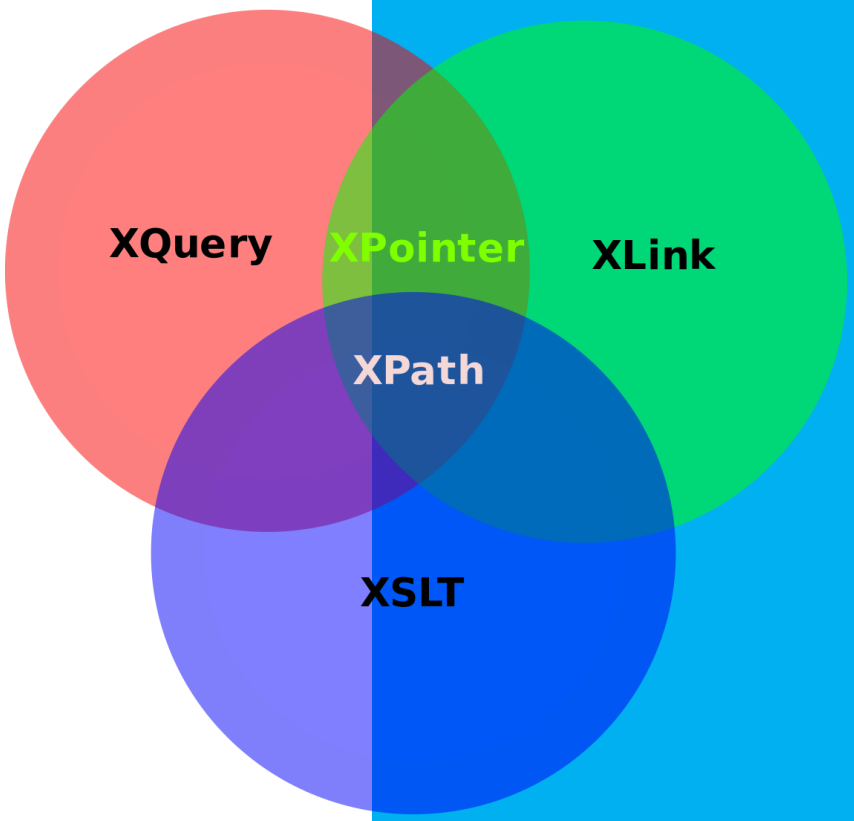
Actividad 8

Tratamiento De La Información Con XML



Rubén Beltrán Muñoz

Lenguaje De Marcas

1º DAM

Índice

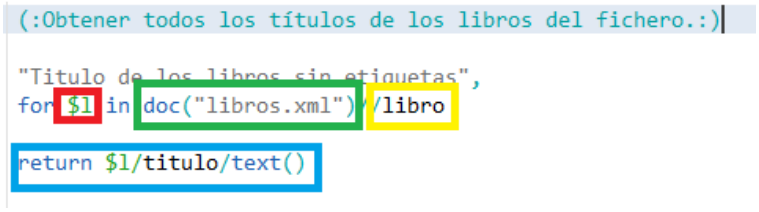
1. Ejemplo de uso de la cláusula FOR. Obtener todos los títulos de los libros del fichero.

```
(:Obtener todos los títulos de los libros del fichero.:)

"Titulo de los libros sin etiquetas",
for $l in doc("libros.xml")//libro

return $l/titulo/text()
```

Consulta XQuery

```
Titulo de los libros sin etiquetas
Learning XML
XML Imprescindible
XML Schema
XPath Essentials
Beginning XSLT 2.0: Form Novice to
Professional
XQuery
```

Resultado Consulta



El funcionamiento de nuestra consulta es el siguiente: realizamos un **bucle for**, en el cual la **variable $l** (color Rojo) tomara como valor cada uno de los **nodos de libro** (color amarillo) del documento **libros.xml** (color verde).
Cada uno de los nodos los retornaremos con la **variable $l** en cada una de las tuplas vinculadas a la misma.
Con la línea **$l/titulo/text()** declaramos que nos retorne dentro de la tupla guardada en la variable $l (libro) el título en formato texto **(text()).**
De no especificar ese último dato nos saldría el resultado con etiquetas como muestra la siguiente imagen:

```
(:Obtener todos los títulos de los libros del fichero.:)

"Titulo de los libros sin etiquetas",
for $l in doc("libros.xml")//libro

return $l/titulo
```

```
Titulo de los libros sin etiquetas
<titulo>Learning XML</titulo>
<titulo>XML Imprescindible</titulo>
<titulo>XML Schema</titulo>
<titulo>XPath Essentials</titulo>
<titulo>Beginning XSLT 2.0: Form Novice to
Professional</titulo>
<titulo>XQuery</titulo>
```

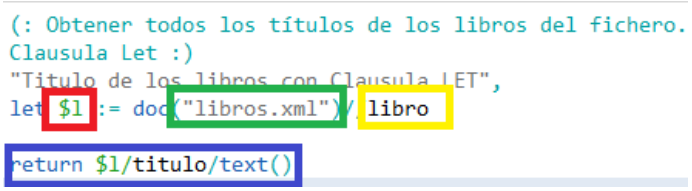
2. Ejemplo de uso de la cláusula LET. Obtener todos los títulos de los libros del fichero.

```
(: Obtener todos los títulos de los libros del fichero.
Clausula Let :)
"Titulo de los libros con Clausula LET",
let $l := doc("libros.xml")//libro

return $l/titulo/text()
```

Consulta XQuery

```
Titulo de los libros con Clausula LET
Learning XML
XML Imprescindible
XML Schema
XPath Essentials
Beginning XSLT 2.0: Form Novice to
Professional
XQuery
```

Resultado Consulta



El funcionamiento de nuestra consulta es el siguiente: con cláusula **LET** asignamos a la **variable $l** (color Rojo) el valor cada uno de los **nodos de libro** (color amarillo) del documento **libros.xml** (color verde).
Cada uno de los nodos los retornaremos con la **variable $l** que previamente hemos asignado con nuestra clausula **LET** en cada una de las tuplas vinculadas a la misma.
Con la línea **$l/titulo/text()** declaramos que nos retorne dentro de la tupla guardada en la variable $l (libro) el título en formato texto **(text()).**

De no especificar ese último dato nos saldría el resultado con etiquetas como muestra la siguiente imagen:

```
(: Obtener todos los títulos de los libros del fichero.
Clausula Let :)
"Titulo de los libros con Clausula LET",
let $l := doc("libros.xml")//libro

return $l/titulo
```

```
Titulo de los libros con Clausula LET
<titulo>Learning XML</titulo>
<titulo>XML Imprescindible</titulo>
<titulo>XML Schema</titulo>
<titulo>XPath Essentials</titulo>
<titulo>Beginning XSLT 2.0: Form Novice to
Professional</titulo>
<titulo>XQuery</titulo>
```

Se visualiza el resultado en vez con el **formato text()** con el **formato data()** siendo visible también:

```
(: Obtener todos los títulos de los libros del fichero.
Clausula LET:)
"Titulo de los libros con Clausula LET",
let $l := doc("libros.xml")//libro

return $l/titulo/data()
```

Consulta XQuery

```
Titulo de los libros con Clausula LET
Learning XML
XML Imprescindible
XML Schema
XPath Essentials
Beginning XSLT 2.0: Form Novice to
Professional
XQuery
```

Resultado Consulta

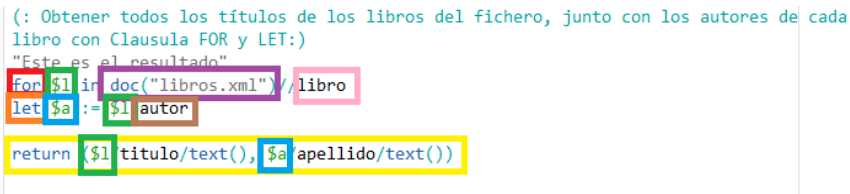
3. Ejemplo de uso de la cláusula FOR y LET juntas. Obtener todos los títulos de los libros del fichero, junto con los autores de cada libro.

```
(: Obtener todos los títulos de los libros del fichero, junto con los autores de cada
libro con Clausula FOR y LET:)
"Este es el resultado",
for $l in doc("libros.xml")//libro
let $a := $l/autor

return ($l/titulo/text(), $a/apellido/text())
```

```
Este es el resultado
Learning XML
Ray
XML Imprescindible
Harold
Means
XML Schema
van der Vlist
XPath Essentials
Watt
Beginning XSLT 2.0: Form Novice to
Professional
Tennison
XQuery
Walmsley
```

Consulta XQuery

Resultado Consulta



En esta consulta tenemos un **clausula for** la cual la asignamos en la **variable $l (libro)**el valor del nodo libro, del **archivo libros.xml**
Después tenemos una **cláusula Let** con un **variable $a** con la cual asignaremos en cada **bucle for** el valor de la tupla de la variable anterior **$l (libro) mas /autor**.

Después retornaremos por medio de la **variable $l (libro) /** titulo (el título de ese nodo de libros) seguido de la $a **(libro/autor) /**apellido el apellido de la misma tupla en la que nos encontramos. En formato texto en ambas sin etiquetas

En esta consulta seria mas visible los resultados obtenidos con las etiquetas de los mismos, con lo que eliminamos los formatos **Text()** de nuestra consulta:

```
(: Obtener todos los títulos de los libros del fichero, junto con los autores de cada libro:)
"Este es el resultado con etiquetas",
for $l in doc("libros.xml")//libro
let $a := $l/autor

return ($l/titulo, $a/apellido)
```

```
Este es el resultado con etiquetas
<titulo>Learning XML</titulo>
<apellido>Ray</apellido>
<titulo>XML Imprescindible</titulo>
<apellido>Harold</apellido>
<apellido>Means</apellido>
<titulo>XML Schema</titulo>
<apellido>van der Vlist</apellido>
<titulo>XPath Essentials</titulo>
<apellido>Watt</apellido>
<titulo>Beginning XSLT 2.0: Form Novice to
Professional</titulo>
<apellido>Tennison</apellido>
<titulo>XQuery</titulo>
<apellido>Walmsley</apellido>
```

Etiqueta duplicada

Consulta XQuery

Resultado Consulta

De esta manera podemos observar que por un mismo titulo nos ha sacado dos apellidos distintos debido a que ese libro tiene dos autores distintos. De la cual en la otra consulta no quedaba tan claro

4. Listar publicación y título de los libros que tienen más de un autor.

```
(: Listar publicacion y título de los libros que
tienen más de un autor:)
"Libros con mas de 1 Autor",

for $l in doc("libros.xml")//libro
where count($l/autor)>1

return ($l/@publicacion, $l/titulo/text(),
 $l/autor/apellido/text(), $l/autor/nombre/text())
```

```
Libros con mas de 1 Autor
publicacion="2003"
XML Imprescindible
Harold
Means
Elliot Rusty
W. Scott
```

Consulta XQuery

Resultado Consulta

```
for $l in doc("libros.xml")//libro
where count($l/autor)>1

return ($l/@publicacion, $l/titulo/text(),
 $l/autor/apellido/text(), $l/autor/nombre/text())
```

Esta consulta la realizamos por medio de un **clausula for** almacenándola en la **variable $l** del **archivo libros.xml** los **nodos de libro.** Con la **cláusula Where** declaramos que solo almacén las tuplas que cumplan la siguiente condición, la cual en este caso es que **cuente los autores** por tupla y solo **almacene los que sean mayores de 1 autor**.

En el mensaje de retorno podemos observar también que mostramos un **atributo** perteneciente a la **etiqueta libro ($l**), con el **símbolo @ seguido del nombre** del atributo. **Mostraremos también el título, apellido del autor y nombre del autor, todos ellos en formato texto**

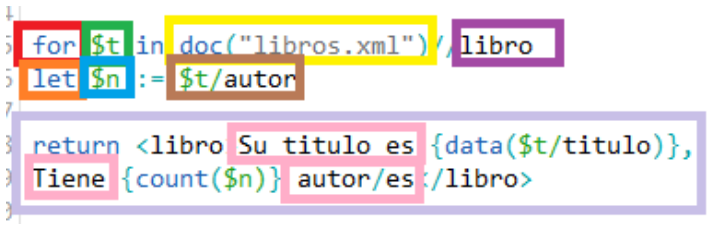
5. Por cada libro, obtener su título y el número de autores, agrupados en un elemento

```
(:Por cada libro, obtener su título y
 el número de autores, agrupados en un elemento:)
"Resultado de la consulta",

for $t in doc("libros.xml")//libro
let $n := $t/autor

return <libro>Su titulo es {data($t/titulo)},
Tiene {count($n)} autor/es</libro>
```

```
<libro>Su titulo es Learning XML,
Tiene 1 autor/es</libro>
<libro>Su titulo es XML Imprescindible,
Tiene 2 autor/es</libro>
<libro>Su titulo es XML Schema,
Tiene 1 autor/es</libro>
<libro>Su titulo es XPath Essentials,
Tiene 1 autor/es</libro>
<libro>Su titulo es Beginning XSLT 2.0: Form Novice to
Professional,
Tiene 1 autor/es</libro>
<libro>Su titulo es XQuery,
Tiene 1 autor/es</libro>
```

Consulta XQuery

Resultado Consulta



Esta consulta realizamos por medio de la **clausula For** un bucle el cual almacenamos en la **variable $t** de los **nodos libro** correspondiente al **archivo libros.xml**
Después por medio de la **cláusula Let** asignamos en la **variable $n** los datos correspondientes a la tupla de la **variable $t (libro) / autor**

Esta forma de retornar la consulta es mas visible ya **incluyendo un texto** podemos realizar una descripción de lo que es cada dato que nos va a devolver la consulta. En este caso nos devolver **el titulo y el numero de autores por libro.** Como observamos el resultado es más fácil de interpretar, en consultas anteriores para saber a que pertenece los datos incluíamos la etiqueta de la misma eliminando el formato Text() de la misma.

6. Ejemplo de uso de las cláusulas WHERE y ORDER BY en una consulta con dos ficheros. Obtener los títulos de los libros prestados con sus autores y la fecha de inicio y devolución del préstamo, ordenados por la fecha de inicio del préstamo.

```
(:Ejemplo de uso de las clausulas WHERE y ORDER BY
en una consulta con dos ficheros. Obtener los
títulos de los libros prestados con sus autores
y la fecha de inicio y devolución del préstamo,
ordenados por la fecha de inicio del préstamo.:)

for $l in doc("libros.xml")//libro,
    $p in doc("prestamos.xml")//entrada
where $l/titulo= $p/titulo
order by $p/prestamo/inicio

return <libro>{
 <titulo>{data($l/titulo)}</titulo>,
 <autor>{data($l/autor)}</autor>,
 <fecha_i>{data($p/prestamo/inicio)}</fecha_i>,
 <fecha_d>{data($p/prestamo/devolucion)}</fecha_d>
 }
 </libro>
```

```
<libro>
  <titulo>XML Imprescindible</titulo>
  <autor>HaroldElliot Rusty MeansW. Scott</autor>
  <fecha_i>2011-02-12</fecha_i>
  <fecha_d>2011-02-16</fecha_d>
</libro>
<libro>
  <titulo>XPath Essentials</titulo>
  <autor>WattAdrew</autor>
  <fecha_i>2011-02-23</fecha_i>
  <fecha_d>2011-03-10</fecha_d>
</libro>
<libro>
  <titulo>XML Imprescindible</titulo>
  <autor>HaroldElliot Rusty MeansW. Scott</autor>
  <fecha_i>2011-05-02</fecha_i>
  <fecha_d/>
</libro>
```

En esta consulta por medio de una **clausula For** vamos a recorrer dos archivos .xml y vamos a incluir en dos variables distintas dos nodos distintos. Después por medio de la **cláusula Where** vamos a realizar una comparación de dos datos de los nodos incluidos en nuestras variables. Una vez realizado la comparación las vamos a ordenar por medio de la cláusula **Order By** de un valor que vamos a establecer nosotros.
En vamos a retornar una serie de datos, correspondientes a nuestras variables y las vamos a encapsular en unas etiquetas para que la consulta se mas reconocible.

```
for $l in doc("libros.xml")//libro,
    $p in doc("prestamos.xml")//entrada
where $l/titulo= $p/titulo
order by $p/prestamo/inicio
```

Realizamos unas **clausula For**, por la cual vamos a recorrer dos archivos. En el **bucle For** que realizamos en el archivo **libros.xml** declaramos una **variable $l** en la cual vamos a incluir los **nodos libro**. En el otro bucle que estamos realizamos en el archivo **prestamos.xml** declaramos una **variable $p** en la que incluimos los **nodos correspondientes a entrada**

Tras realizar el paso anterior tenemos un clausula **Where** que vamos a guardar solo los nodos en los cuales tengan el mismo titulo tanto la tupla perteneciente **a libro ($l)** como la perteneciente a **entrada ($p).**
Una vez realizado el filtro anterior las vamos a ordenar por medio de la cláusula **Order By** y con los datos de **$p (entrada) / préstamo /inicio**

```
for $l in doc("libros.xml")//libro,
    $p in doc("prestamos.xml")//entrada
where $l/titulo= $p/titulo
order by $p/prestamo/inicio
```

```
return <libro>{
 <titulo>{data($l/titulo)}</titulo>,
 <autor>{data($l/autor)}</autor>,
 <fecha_i>{data($p/prestamo/inicio)}</fecha_i>,
 <fecha_d>{data($p/prestamo/devolucion)}</fecha_d>
 }
 </libro>
```

Los datos que nos va a retornar los tenemos entre etiquetas para una mayor claridad de los mismo. Como podemos observar solicitarnos datos pertenecientes a la **variable libro ($l)[archivo libros.xml]** y **variable entrada ($p)[archivo prestamos.xml]**

7. Ejemplo, hacer una consulta que devuelva los títulos de los libros almacenados en el fichero y su primer autor. En caso de que haya más de un autor para un libro se añade un segundo autor “cia”.

```
(:Consulta que devuelva los títulos de los libros
 almacenados en el fichero y su primer autor.
  En caso de que haya más de un autor para un libro
   se añade un segundo autor “cia”.:)

for $l in doc("libros.xml")//libro
let $b := $l/autor
return($l/titulo,$l/autor[1],

if (count($b)>1) then
 <cia>el segundo autor es {data ($b[2]) } </cia>
else  ()
)
```

```
<titulo>Learning XML</titulo>
<autor>
  <apellido>Ray</apellido>
  <nombre>Erik T.</nombre>
</autor>
<titulo>XML Imprescindible</titulo>
<autor>
  <apellido>Harold</apellido>
  <nombre>Elliot Rusty</nombre>
</autor>
<cia>el segundo autor es MeansW. Scott</cia>
<titulo>XML Schema</titulo>
<autor>
  <apellido>van der Vlist</apellido>
  <nombre>Eric</nombre>
</autor>
<titulo>XPath Essentials</titulo>
<autor>
  <apellido>Watt</apellido>
  <nombre>Adrew</nombre>
</autor>
<titulo>Beginning XSLT 2.0: Form Novice to
```

En esta consulta al igual que en el punto 3, en esta consulta tenemos un **clausula For** la cual la asignamos en la **variable $l (libro)**el valor del nodo libro, del **archivo libros.xml**
Después tenemos una **cláusula Let** con un **variable $b** con la cual asignaremos en cada **bucle For** el valor de la tupla de la variable anterior **$l (libro) mas /autor**.

Después retornaremos por medio de la **variable $l (libro) /** titulo (el título de ese nodo de libros) seguido de la **$b (libro/autor)** de la misma tupla en la que nos encontramos. Tenemos al igual que una Array [1] para especificar que utilicemos este formato en el **autor [1]**

```
for $l in doc("libros.xml")//libro
let $b := $l/autor
return($l/titulo,$l/autor[1],
```

Por medio de una **cláusula if** y de una comparativa de **tuplas de autor mayor a 1,** en el caso de que **encuentre más de 1 autor mostrara también los siguientes autores con la etiqueta <cia></cia>**

```
if (count($b)>1) then
 <cia>el segundo autor es {data ($b[2]) } </cia>
else  ()
)
```

Observación del alumno: Una vez realizada la actividad de consulta por medio del Software se observa que es muy similar a la realización de consultas en un base de datos por medio de una aplicación destinado a ello como SqlDeveloper, con la peculiaridad de que trabajamos con las etiquetas y las podemos mostrar para su fácil interpretación