

LMSGI05.2 XQuery



☰ Tratamiento y recuperación de datos en XQuery

☰ Expresiones

☰ Variables, condicionales y nuevos documentos

☰ Cláusulas FLWOR

☰ Operadores y funciones

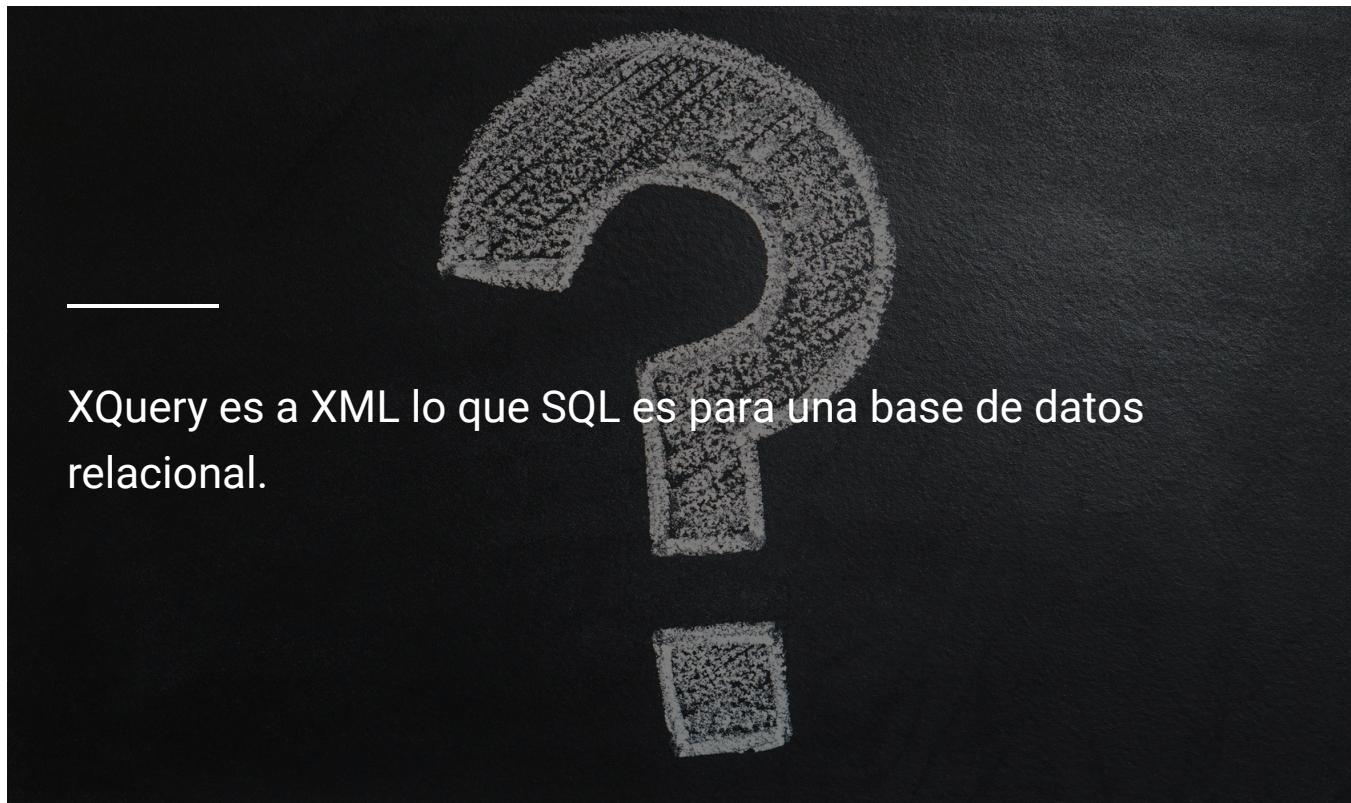


Autoevaluación

CONTENIDO PDF

☰ Contenido PDF

Tratamiento y recuperación de datos en XQuery



XQuery es a XML lo que SQL es para una base de datos relacional.

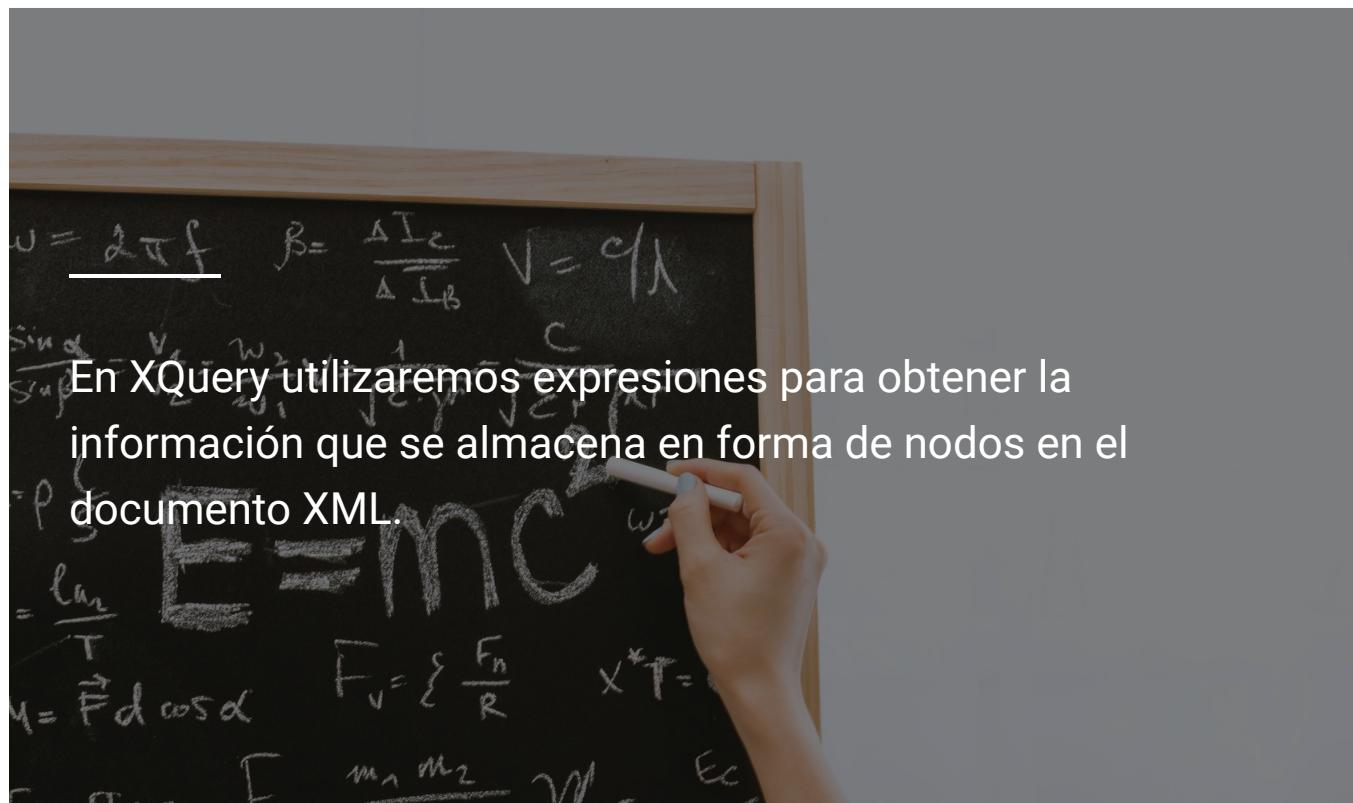
Características

XQuery es un lenguaje de consulta diseñado para extraer y manipular datos de documentos XML. Fue desarrollado por el W3C y se basa en XPath (XML Path Language) y en la teoría de las bases de datos relacionales.

Las principales características y usos de XQuery incluyen:

- **Consultas sobre documentos XML:** XQuery permite realizar consultas complejas sobre documentos XML, incluyendo la selección de elementos, la filtración de datos, la combinación de documentos y la transformación de estructuras XML.
- **Manipulación de datos XML:** Además de realizar consultas, XQuery también puede utilizarse para insertar, actualizar y eliminar datos dentro de documentos XML, lo que permite la modificación dinámica de contenido XML.
- **Integración con otros estándares:** XQuery se integra bien con otros estándares XML, como XPath, XSLT (Extensible Stylesheet Language Transformations) y XML Schema, lo que facilita la combinación de estas tecnologías para procesamiento de documentos XML complejos.
- **Soporte en sistemas de bases de datos:** Muchos sistemas de bases de datos XML y sistemas de gestión de contenido ofrecen soporte para XQuery, lo que permite a los desarrolladores aprovechar las capacidades de consulta y manipulación de datos proporcionadas por XQuery en entornos empresariales.

Expresiones

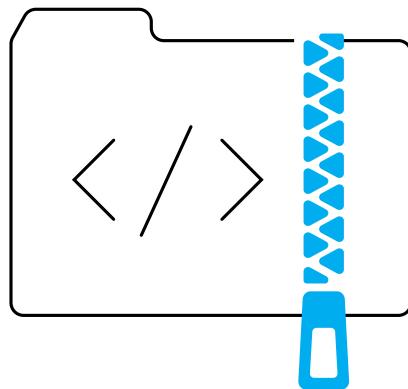


Expresiones básicas

Una consulta en XQuery es una expresión que lee una secuencia de datos en XML y devuelve como resultado otra secuencia de datos en XML.

La mayoría de las expresiones están compuestas por la combinación de expresiones más simples unidas mediante operadores y palabras reservadas.

Por ejemplo, si tenemos el siguiente documento XML, "clase.xml":



Custom code isn't available in PDF format

Para seleccionar nodos en este documento se utilizan funciones. Por ejemplo, con la función `doc()` abriremos y veremos el documento XML.

```
doc("clase.xml")
```

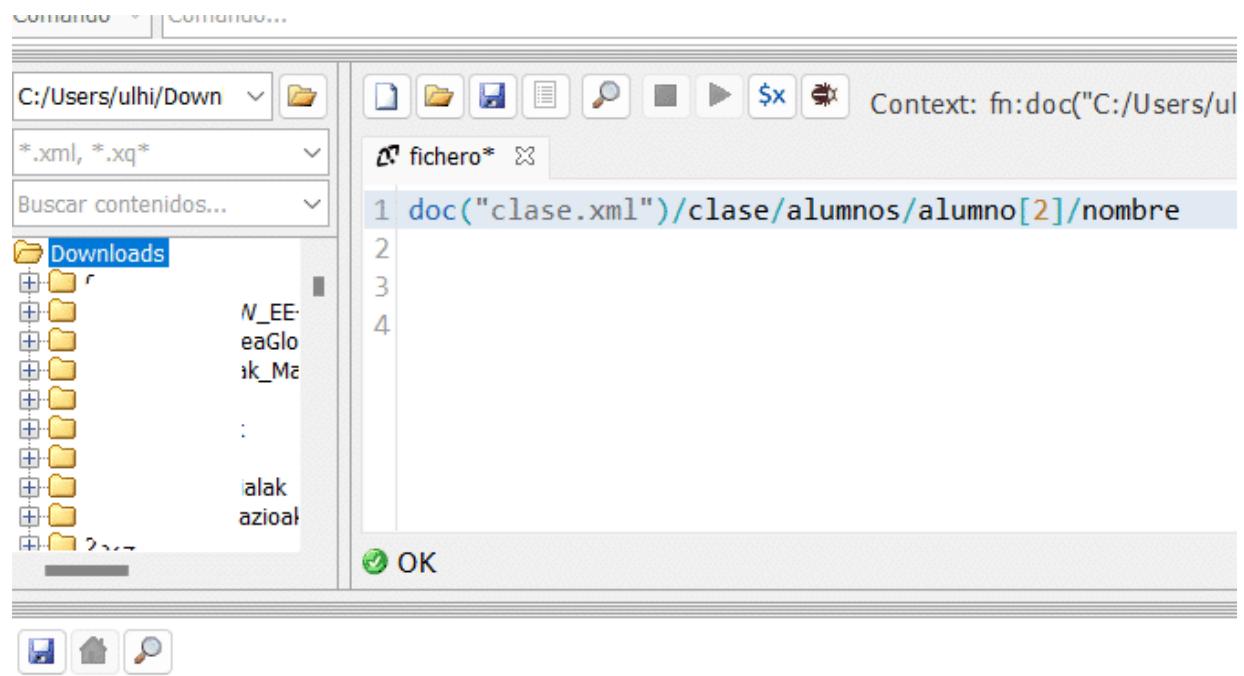
A partir de aquí, se pueden aplicar expresiones XPath y otras funciones para obtener la información deseada.

También es posible añadir información al código por medio de comentarios:

```
(: Esto es un comentario XQuery :)
```

Y para limitar y filtrar resultados de las consultas, se pueden utilizar predicados XPath. Por ejemplo, para obtener el nombre del segundo alumno:

doc("clase.xml")/clase/alumnos/alumno[2]/nombre



<nombre>Maite</nombre>

Elaboración propia.

LMSGI05 BaseX-XQuery_I



Variables, condicionales y nuevos documentos

En XQuery es posible utilizar variables y utilizar instrucciones condicionales para crear nuevos documentos.

Nuevos documentos

En XQuery los caracteres {} delimitan las expresiones que son evaluadas para crear un documento nuevo.

The screenshot shows an XQuery editor interface. At the top, there's a toolbar with various icons. To the right of the toolbar, the context is displayed as "Context: fn:doc("C:/Users/"). Below the toolbar, a tab bar shows "fichero*". The main code editor area contains the following XQuery code:

```
1 <Modulo>
2 {doc("clase.xml")//asignatura/text()}
3 </Modulo>
4
```

Below the code editor is a status bar with a green checkmark icon and the word "OK". At the bottom of the editor window, there are three small icons: a blue square with a white symbol, a grey house-like icon, and a magnifying glass icon.

<Modulo>Lenguajes de marcas</Modulo>

Elaboración propia.

Variabes

En XQuery uno de los usos fundamentales de las variables es almacenar elementos para poderlos utilizar posteriormente.

The screenshot shows an XQuery editor interface. At the top, there is a toolbar with various icons: file, folder, print, copy, paste, search, and others. To the right of the toolbar, it says "Context: fn:doc("C:/Users/ulhi/Dov"). Below the toolbar, the title bar reads "fichero*". The main code area contains the following XQuery code:

```
1 let $NumAlumnos:=doc("clase.xml")//alumnos/alumno
2 return concat("Número de alumnos:",count($NumAlumnos))
3
```

Below the code, a status bar indicates "OK" with a green checkmark icon. At the bottom of the editor window, there are three small icons: a blue square, a grey house, and a magnifying glass.

Número de alumnos:2

Elaboración propia.

Instrucción condicional

XQuery admite también expresiones condicionales del tipo if-then-else con la misma semántica que tienen en los lenguajes de programación habituales.

The screenshot shows a Java-based IDE interface with a central code editor window. The title bar of the window says "fichero*". Below the title bar, there is a status bar with the text "Context: fn:doc("C:/Users/ulhi/Downloads/clase.xml")". The code editor contains the following XSLT code:

```
1 (: En la variable tipo se hace la elección
2   del tipo de listado que se desea      :)
3 let $tipo := "alumno"
4 return
5   if ($tipo="alumno") then
6     <listadoAlumno>{doc("clase.xml")//alumno/nombre}</listadoAlumno>
7   else
8     <listadoProfesor>{doc("clase.xml")//profesor/nombre}</listadoProfesor>
```

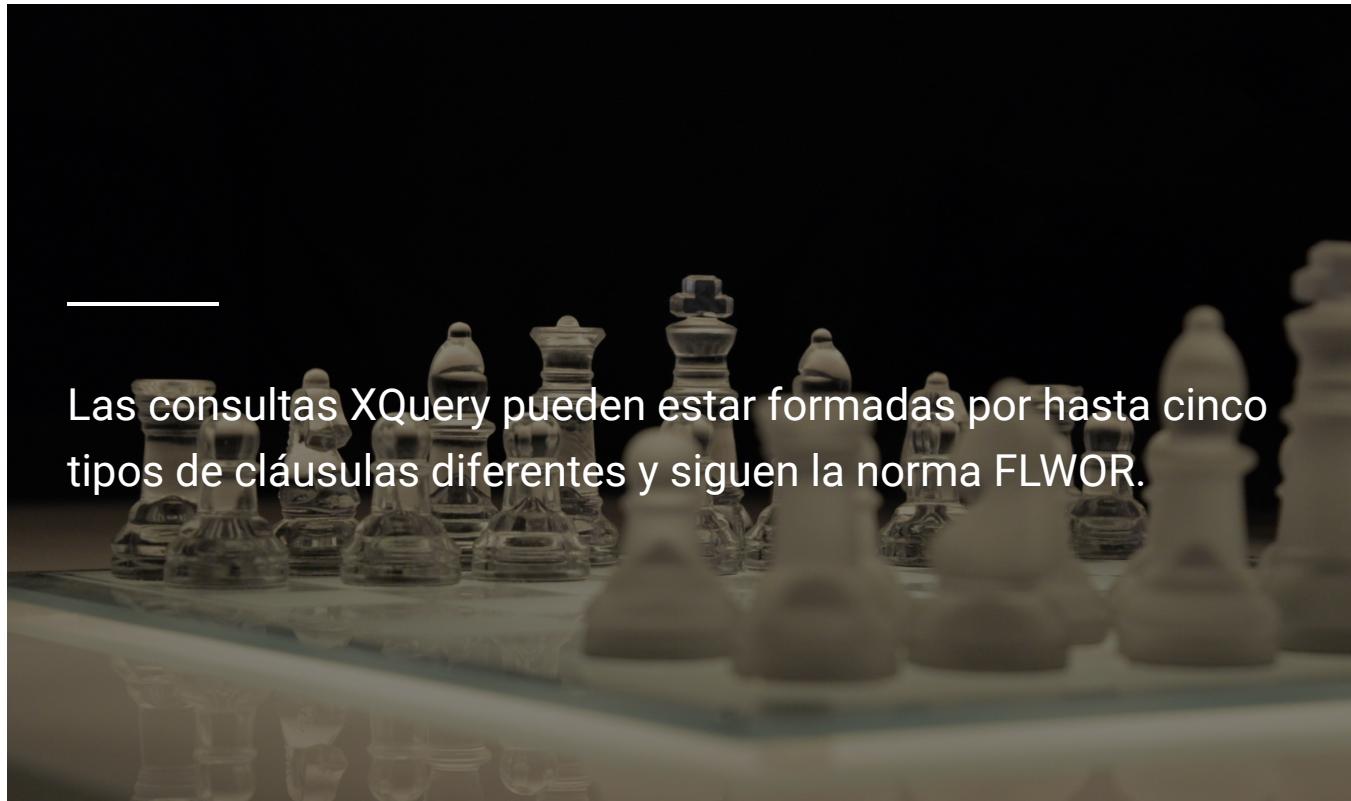
Line numbers 1 through 10 are visible on the left side of the code editor. At the bottom right of the code editor, there is a green checkmark icon followed by the text "OK".



```
<listadoAlumno>
  <nombre>Felipe</nombre>
  <nombre>Maite</nombre>
</listadoAlumno>
```

Elaboración propia.

Cláusulas FLWOR



Las consultas XQuery pueden estar formadas por hasta cinco tipos de cláusulas diferentes y siguen la norma FLWOR.

Introducción

FLWOR es un acrónimo que representa las cinco cláusulas principales de XQuery: For, Let, Where, Order by, y Return. Estas cláusulas se utilizan en conjunto para construir consultas complejas y realizar operaciones sobre conjuntos de datos XML. Así:

1

For: Se utiliza para iterar sobre un conjunto de nodos en un documento XML o en una secuencia de datos. Se puede definir una variable que toma diferentes valores en cada iteración.

2

Let: Se utiliza para definir variables en la consulta. Estas variables pueden ser utilizadas en otras partes de la consulta, permitiendo la reutilización y simplificación del código.

3

Where: Se utiliza para aplicar condiciones de filtro a los datos seleccionados en la consulta. Permite restringir los resultados basados en ciertas condiciones lógicas.

4

Order by: Se utiliza para ordenar los resultados de la consulta basándose en el valor de una o más variables. Permite especificar el orden en el que se deben devolver los resultados.

5

Return: Se utiliza para especificar qué datos deben ser devueltos como resultado de la consulta. Puede contener expresiones que calculan valores o seleccionan datos específicos de los resultados de la consulta.

Como norma:

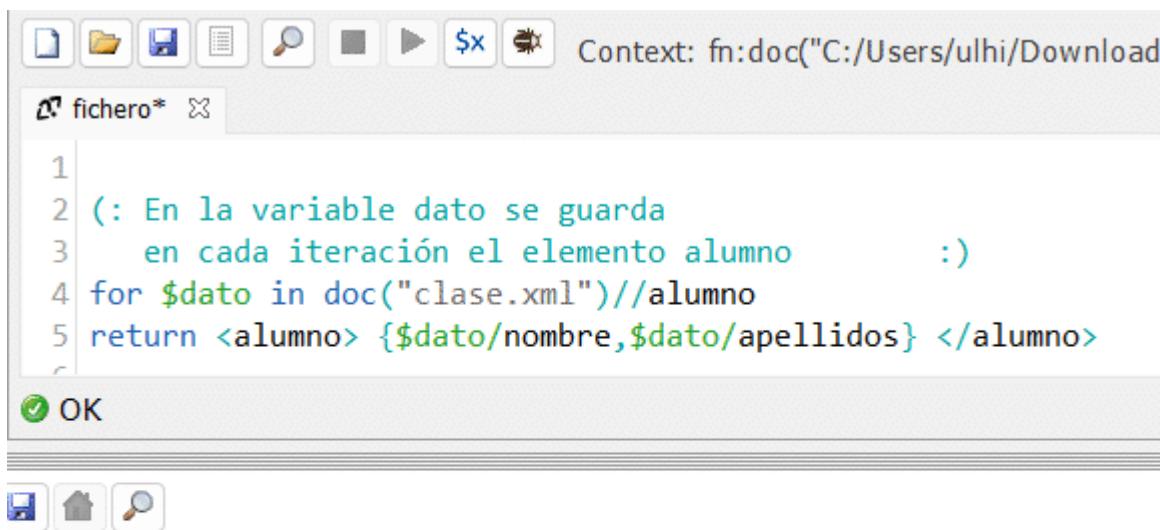
- En una sentencia FLWOR al menos ha de existir una cláusula FOR ó una LET.
- El resto, si existen, han de respetar escrupulosamente el orden dado por el nombre: F-L-W-O-R.

FOR

La instrucción for se utiliza para evaluar individualmente cada uno de los resultados de una secuencia de nodos. En un for se definen dos cosas:

- Una variable, que será la que irá cogiendo los resultados de la secuencia uno por uno.
- La palabra clave in, en la que se define cuál es la secuencia de elementos para procesar.

La manera más sencilla de expresión FLWOR es combinar el for con el return para devolver los valores obtenidos de forma secuencial:



```
Context: fn:doc("C:/Users/ulhi/Download/clase.xml")
```

```
1 (: En la variable dato se guarda
2   en cada iteración el elemento alumno      :)
3 for $dato in doc("clase.xml")//alumno
4 return <alumno> {$dato/nombre,$dato/apellidos} </alumno>
```

OK

```
<alumno>
  <nombre>Felipe</nombre>
  <apellidos>Aranburu</apellidos>
</alumno>
<alumno>
  <nombre>Maite</nombre>
  <apellidos>Garcia</apellidos>
</alumno>
```

Elaboración propia.

LET

Permite declarar variables y asignarles un valor. Sobre todo se utiliza para almacenar valores que se utilizarán más tarde.

Por ejemplo:

The screenshot shows a software interface for XSLT development. At the top, there's a toolbar with various icons. Below it is a menu bar with the text "Context: fn:doc("C:/Users/ulhi/D..."). The main window has a title bar "fichero*". Inside, there's a code editor with the following XSLT code:

```
1 (: FOR con LET      :)
2 for $alum in doc ("clase.xml")//alumno
3 let $nom := $alum/nombre
4 let $ape := $alum/apellidos
5 return <alumno> {concat ($nom, " ",$ape)}</alumno>
6
```

Below the code editor is a status bar with a green "OK" icon. At the bottom of the interface, there are three small icons: a blue square, a house, and a magnifying glass.

On the right side of the interface, there's a preview area displaying the output of the XSLT transformation:

```
<alumno>Felipe Aranburu</alumno>
<alumno>Maite Garcia</alumno>
```

Elaboración propia.

WHERE

Filtra tuplas producidas por las cláusulas FOR y LET, quedando solo las que cumplen con la condición. Normalmente en el filtro se utiliza algún tipo de predicado XPath.

Por ejemplo:

```
1 (: FOR con WHERE      :)
2 for $alumno in doc("clase.xml")//alumno
3 where $alumno[nombre="Maite"]/nombre
4 return $alumno/nombre
5 (: 
6 También se puede utilizar "if":
7
8 for $alumno in doc("clase.xml")//alumno
9 return if ($alumno/nombre="Maite") then $alumno/nombre
10 :)
```

OK

<nombre>Maite</nombre>

Elaboración propia.

ORDER BY

Ordena las tuplas generadas por FOR y LET después de que hayan sido filtradas por la cláusula WHERE. Por defecto el orden es ascendente, pero se puede usar el modificador "descending" para cambiar el sentido del orden.

Por ejemplo:

The screenshot shows the Oxygen XML Editor interface. The top bar displays the context as "Context: fn:doc('C:/Users...'). The main window has a tab labeled "fichero*" containing XQuery code:

```
1 (: FOR con ORDER BY      :)
2 for $alumno in doc ("clase.xml")//alumno
3 order by $alumno/apellidos descending
4 return $alumno
5
6
```

Below the code, there is a green checkmark icon followed by the word "OK". The bottom part of the editor shows the resulting XML output:

```
<alumno>
  <nOMBRE>Maite</nOMBRE>
  <apellidos>Garcia</apellidos>
</alumno>
<alumno>
  <nOMBRE>Felipe</nOMBRE>
  <apellidos>Aranburu</apellidos>
</alumno>
```

Elaboración propia.

RETURN

Construye el resultado de la expresión FLWOR para una tupla dada.

Context: in:doc C:/Users/umli/Downloads/clase.xml]

fichero* x

```
1 (: Para mostrar una salida numerada :)
2
3 for $i at $cont  in doc("clase.xml")//alumno/nombre
4 return <elemento><orden>{$cont}</orden><izena>{$i/text()}</izena></elemento>
5
```

OK

Result

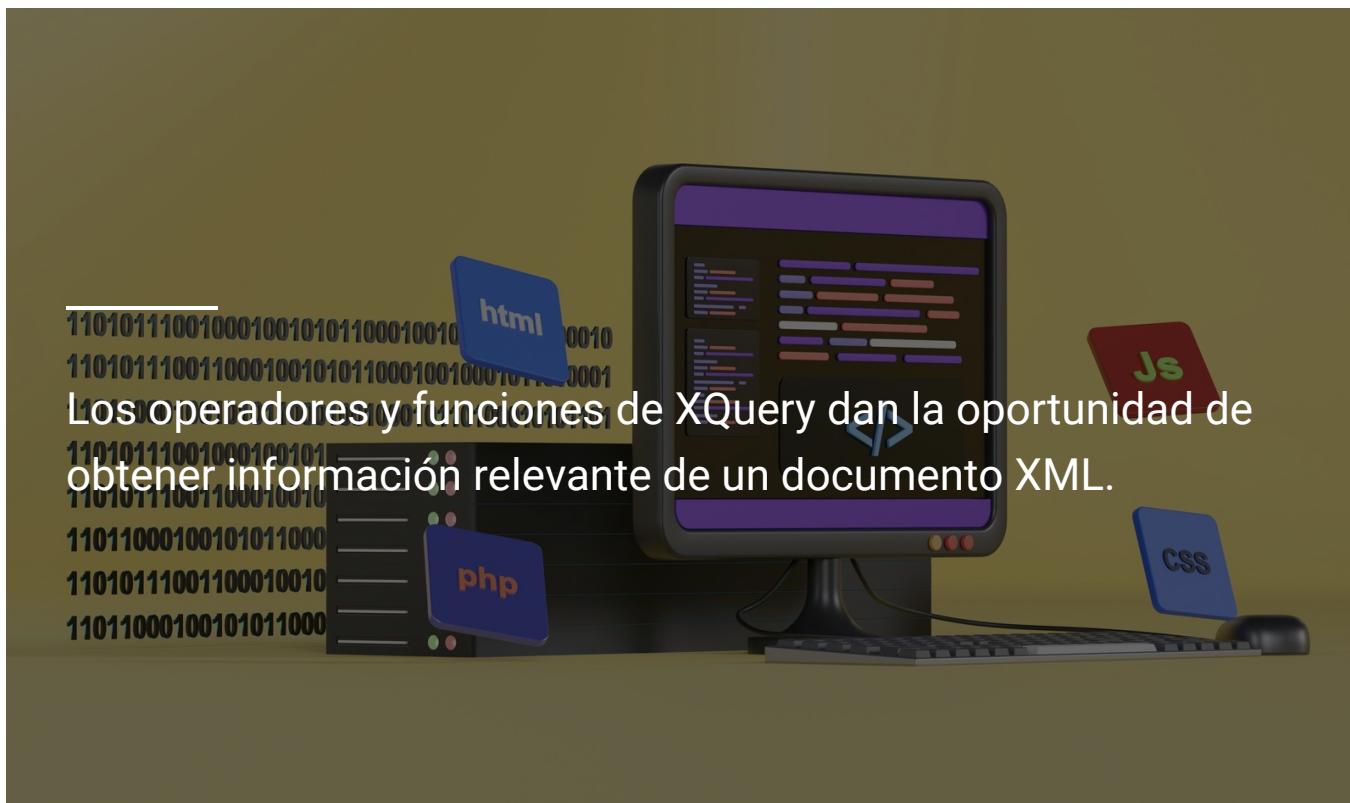
```
<elemento>
<orden>1</orden>
<izena>Felipe</izena>
</elemento>
<elemento>
<orden>2</orden>
<izena>Maite</izena>
</elemento>
```

Elaboración propia.

LMSGI05 BaseX-XQuery_II



Operadores y funciones



Los operadores y funciones de XQuery dan la oportunidad de obtener información relevante de un documento XML.

Operadores

Veamos ahora algunos de los operadores más importantes agrupados según su funcionalidad:

Comparación de valores

Comparan dos valores y produce un error si alguno de los operandos es una secuencia de longitud mayor de 1. Estos operadores son:

Operador	Descripción
eq	Igual (equal).
ne	Distinto (not equal).
lt	Menor que (less than).
le	Menor que o igual (less than or equal).
gt	Mayor que (greater than).
ge	Mayor que o igual (greater than or equal).

Comparación generales

Permiten comparar operandos que sean secuencias. Y si para secuencias de tamaño 1 funcionan igual que las de arriba, para secuencias de mayor tamaño, pueden tener resultados que parecen extraños.

Operador	Descripción
=	Igual.
!=	Distinto.

Operador	Descripción
<	Menor que.
<=	Menor que o igual.
>	Mayor que.
>=	Mayor que o igual.

Comparación de nodos

Comparan la identidad de dos nodos.

Operador	Descripción
is	Devuelve true si las dos variables que actúan de operandos están ligadas al mismo nodo.
<<	Compara la posición de dos nodos. Devuelve true si el nodo ligado al primer operando ocurre primero en el orden del documento que el nodo ligado al segundo. También se puede utilizar >> pero en sentido contrario.

Lógicos

Se emplean para combinar condiciones lógicas dentro de un predicado.

Operador	Descripción
and	Devuelve true si los dos operandos son true.
or	Devuelve true si cualquiera de los operandos es true.

Secuencias de nodos

Devuelven secuencias de nodos en el orden del documento y eliminan duplicados de las secuencias resultado.

Operador	Descripción
union	Devuelve una secuencia que contiene todos los nodos que aparecen en alguno de los dos operandos que recibe.
intersect	Devuelve una secuencia que contiene todos los nodos que aparecen en los dos operandos que recibe.
except	Devuelve una secuencia que contiene todos los nodos que aparecen en el primer operando que recibe y que no aparecen en el segundo.

Aritméticos

Los operadores **+**, **-**, *****, **div** y **mod**, devuelven respectivamente la suma, diferencia, producto, cociente y resto de operar dos números dados.

Funciones

Las funciones predefinidas aportan mayor eficiencia y significado a las consultas. Hay muchas, y por mencionar algunas:

Funciones numéricas

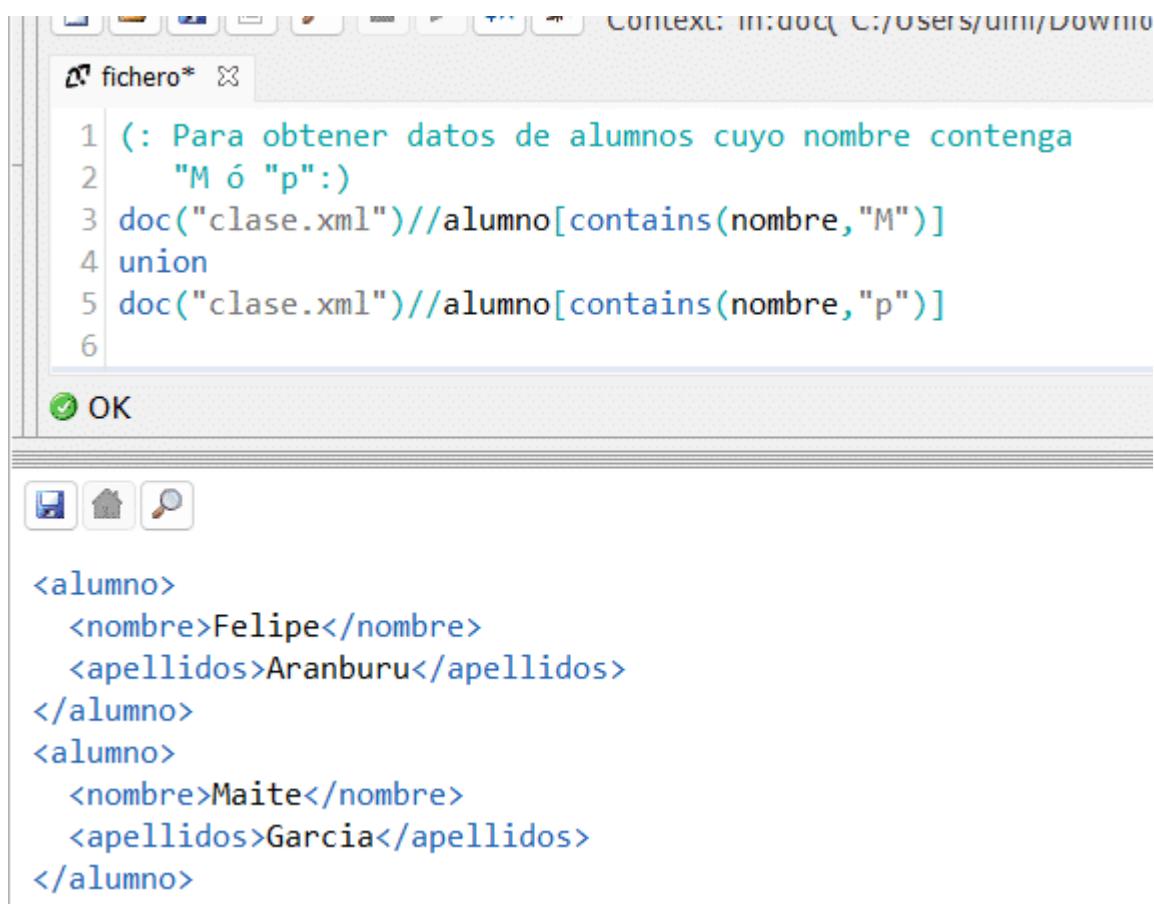
- **round()**, que redondea el valor dado al más próximo.
- **count()**, determina el número de ítems en una colección.
- **min()** devuelve el mínimo de los valores de los nodos dados.
- **max()**, devuelve el máximo de los valores de los nodos dados.
- **avg()**, calcula el valor medio de los valores dados.
- **sum()**, calcula la suma total de una cantidad de ítems dados.

Funciones de cadenas de texto

- **concat()**, devuelve una cadena construida por la unión de dos cadenas dadas.
- **string-length()**, devuelve la cantidad de caracteres que forman una cadena.
- **starts-with(), ends-with()**, determinan si una cadena dada comienza o termina, respectivamente, con otra cadena dada.
- **upper-case(), lower-case()**, devuelve la cadena dada en mayúsculas o minúsculas respectivamente.

Funciones de uso general

- **empty()**, devuelve “true” cuando la secuencia dada no contiene ningún elemento.
- **exists()**, devuelve “true” cuando una secuencia contiene, al menos, un elemento.
- **distinct-values()**, extrae los valores de una secuencia de nodos y crea una nueva secuencia con valores únicos, eliminando los nodos duplicados.



The screenshot shows the Oxygen XML Editor interface. The top window displays XQuery code:

```
1 (: Para obtener datos de alumnos cuyo nombre contenga
2   "M ó "p":)
3 doc("clase.xml")//alumno[contains(nombre,"M")]
4 union
5 doc("clase.xml")//alumno[contains(nombre,"p")]
6
```

The status bar at the bottom indicates the context: Context: in:doc C:/users/umy/Downloads/fichero*.xml

The bottom window shows the resulting XML output:

```
<alumno>
  <nOMBRE>Felipe</nOMBRE>
  <apellidos>Aranburu</apellidos>
</alumno>
<alumno>
  <nOMBRE>Maite</nOMBRE>
  <apellidos>Garcia</apellidos>
</alumno>
```

Elaboración propia.

LMSGI05 BaseX-XQuery_III



Autoevaluación

Pregunta

01/03

XQuery:

- Es un lenguaje de consulta diseñado para extraer y manipular datos de documentos XML.
- Fue desarrollado por el IEEE.
- Se basa en XPath y en la teoría de las bases de datos NoSQL.
- Fue desarrollado por el W3C.
- Se basa en XPath y en la teoría de las bases de datos relacionales.
- Se basa en XPath y en la teoría de las bases de datos jerárquicas.

Pregunta

02/03

En XQuery la mayoría de las expresiones están compuestas por la combinación de expresiones más simples unidas mediante operadores y palabras reservadas.

Falso

Verdadero

Pregunta

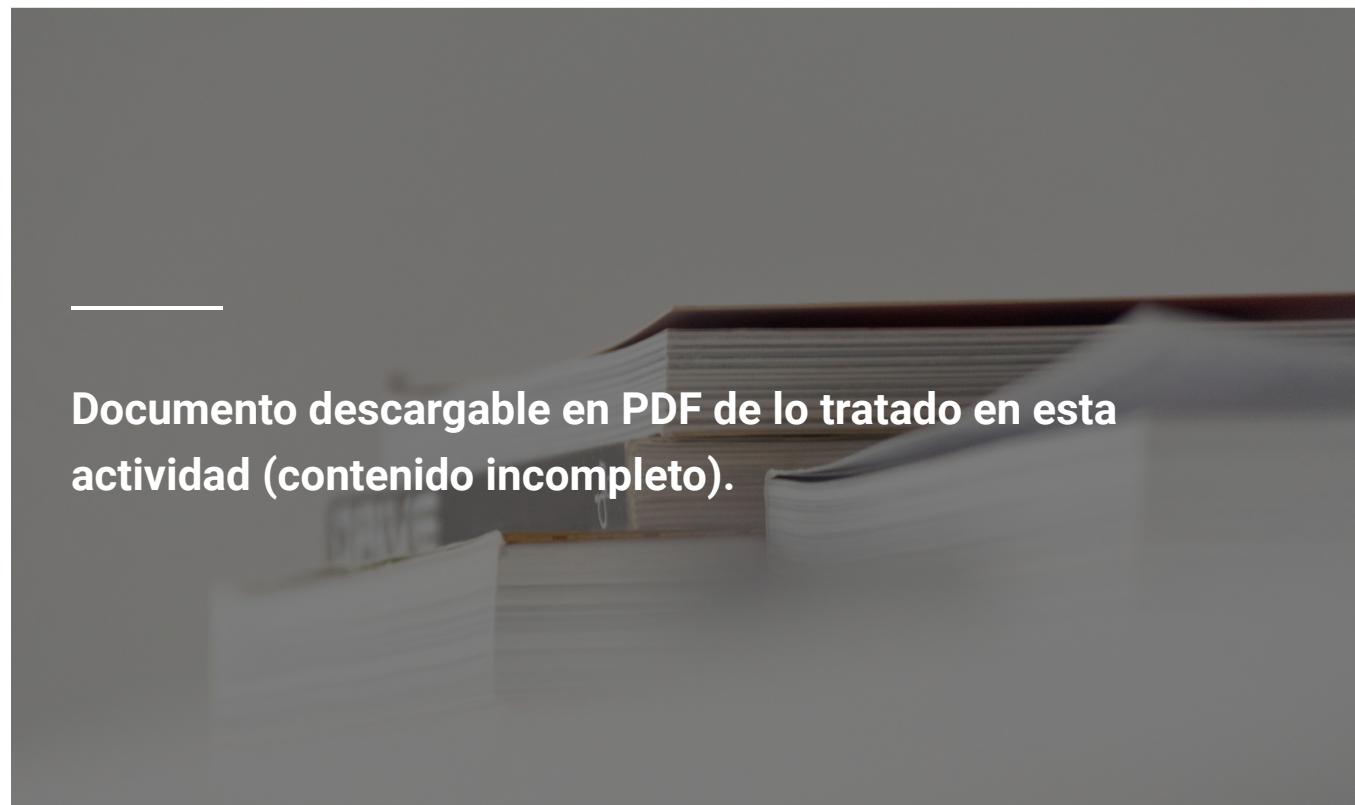
03/03

En XQuery, FLWOR es un acrónimo que representa las cinco cláusulas principales.

Falso

Verdadero

Contenido PDF



El contenido del siguiente documento PDF está incompleto al no ofrecer los enlaces que aparecen en los apuntes ni las funciones de los elementos interactivos de la actividad. Su uso ha de ser por lo tanto, completado por dichos apuntes.

Puedes visualizar y descargar el contenido PDF a través de este enlace

IR AL ENLACE