

**Utiliza BaseX con los ejercicios de la unidad anterior de  
CICLOS**



## 1. Ejemplo con Tree y Folder

BaseX 10.7 interface showing a query on `ejercicio2.xml`. The query is `//ies/ciclos/ciclo/@id`. The result shows 6 items with IDs: `id="ASIR"`, `id="DAW"`, `id="SMR"`, `id="ESTETICABELLEZA"`, `id="FPBPelu"`, and `id="FPBINF"`. The tree view shows the XML structure with `ies` as the root, containing `ciclos` and `modulos` elements. The `ciclos` element has a `ciclo` child with an `@id` attribute.

## 2. Otro ejemplo.

BaseX 10.7 interface showing a query on `ejercicio2.xml`. The query is `//ciclo/decretoTitulo[@año < 2010]/nombre/text()`. The result shows 2 items with names: `<ciclo id="SMR"><nombre>Sistemas Microinformáticos y Redes</nombre><grado>Medio</grado><decretoTitulo año="2008"></decretoTitulo><familia>Informática</familia></ciclo>` and `<ciclo id="ESTETICABELLEZA"><nombre>Estética y Belleza</nombre><grado>Medio</grado><decretoTitulo año="2008"></decretoTitulo><familia>Imagen Personal</familia></ciclo>`. The tree view shows the XML structure with `ies` as the root, containing `ciclos` and `modulos` elements. The `ciclos` element has a `ciclo` child with a `decretoTitulo` attribute and a `nombre` child.

### 3. Último ejemplo

file\* [ejercicio2] - BaseX 10.7

Database Editor View Visualization Options Help

Find Find...

OK 1:1

ejercicio2.xml

ies

@nombre="IES Muxamel"

@web="http://www.iesmuxamel.com"

ciclos

modulos

1 //modulo[ciclo = //ciclo[decretoTitulo/@año="2008"]/@id]/nombre/text()

2

ejercicio2.xml

ies

ciclos

ASIR

dec.

ESTETICABELLE

0228

hor.

0373

cur.

4

cic.

no.

cur.

1

DAW

dec.

FBPelu

dec.

0372

hor.

ciclo

cur.

5

cic.

ciclo

SMR

dec.

FBPINF

dec.

0376

cur.

hor.

ciclo

1 Result, 16 b

Aplicaciones web

Result

Q All Total Time: 2.06 ms

Info

Compiling:

- merge\_descendant: ciclo[(decretoTitulo/@año = "2008")]
- merge\_descendant: modulo[ciclo = util:root().]descendant: ciclo[(decretoTitulo/@año = "2008")]/@id]

Optimizing:

- rewrite context value: -> db.get-pre("ejercicio2", 0)
- rewrite util:root(nodes): util:root(db.get-pre("ejercicio2", 0)) -> db.get-pre("ejercicio2", 0)
- convert to child steps: descendant: modulo[ciclo = util:root().]descendant: ciclo[(decretoTitulo/@año = "2008")]/@id]

Optimized Query:

db.get-pre("ejercicio2", 0)/\*ies/\*modulos/\*modulo[ciclo = util:root().]descendant: ciclo[(decretoTitulo/@año = "2008")]/@id/nombre/text()

Query:

//modulo[ciclo = //ciclo[decretoTitulo/@año="2008"]/@id]/nombre/text()

db:act["ejercicio2", "ejercicio2.xml"]/ies/ciclos/ciclo/text()

40 MB

4. /concesionario/coche[precio > 8000.00]/modelo

The screenshot displays the BaseX 10.7 application window. The top menu bar includes Database, Editor, View, Visualization, Options, and Help. Below it is a toolbar with various icons for file operations and editing.

The main workspace is divided into several panes:

- Left Pane:** Shows the project structure under "C:\Program Files (x86)\BaseX\". It lists files like \*.xml, \*.xq\*, Find contents..., and folders such as bin, data, etc., lib, src, webapp, and BaseX.jar (4596 kb).
- Editor Pane:** Displays the context: db:get("ejercicio\_conc...". The editor shows a query snippet:

```
1 /concesionario/coche[precio > 8000.00]/modelo
```
- Right Pane:** Visualizes the query results as a tree structure. The root node is "ejercicio\_concesionario.xml", which contains a "concesionario" element. This element has three children: "coche", "coche", and "coche". Each "coche" node further branches into "marca", "modelo", "color", "precio", and "ano". The first two "coche" nodes have values like "Volkswa..", "Passat", "Azul", "7900.00", and "2005". The third "coche" node has values like "Astra", "Negro", "8490.00", and "2007".

At the bottom, there are two more sections:

- Results:** Shows the query results in XML format:

```
<modelo>Touran</modelo>  
<modelo>Astra</modelo>
```
- Query Log:** Provides details about the query execution, including the compiled query, optimization steps, and the final optimized query.

```
Compiled:  
- rewrite > comparison: (precio > 8000) -> precio >= 8000.000000000001  
Optimizing:  
- rewrite context value: .. -> db.get-pre("ejercicio_concesionario", 0)  
- rewrite util.root(nodes): util.root(db.get-pre("ejercicio_concesionario", 0)) -> db.get-pre("ejercicio_concesionario", 0)  
Optimized Query:  
db.get-pre("ejercicio_concesionario", 0)/concesionario/coche[precio >= 8000.000000000001]/modelo  
Query:  
/concesionario/coche[precio > 8000.00]/modelo  
Result:  
- Hit(s): 2 Items  
- Updated: 0 Items  
- Printed: 47 b  
- Read Locking: ejercicio_concesionario  
- Write Locking: (none)
```

**5. /concesionario/coche/modelo | /concesionario/coche/precio**

[illegible]

## 6. ¿Cómo puedes mostrar solo los coches con combustibles tipo D?

The screenshot shows the BaseX 10.7 interface with the following components:

- Editor:** Contains the XQuery `1 /concesionario/coche[@combustible="D"]`.
- Visualizer:** Displays the XML result structure for three cars. The first car is a Volkswagen Passat (Azul, 2005, 7900.00). The second is a Volkswagen Touran (Azul, 2007, 8200.00). The third is an Opel Astra (Negro, 2013, 8490.00).
- Result:** Shows the XML output for 2 results, 386 b. The XML is as follows:

```
<coche combustible="D">
  <marca>Volkswagen</marca>
  <modelo>Passat</modelo>
  <color>Azul</color>
  <año>2005</año>
  <precio>7900.00</precio>
</coche>
<coche combustible="D">
  <marca>Opel</marca>
  <modelo>Astra</modelo>
  <color>Negro</color>
  <año>2013</año>
  <precio>8490.00</precio>
</coche>
```
- Info:** Provides details about the query execution, including optimization steps, the query itself, and timing information (Total Time: 1.43 ms).

1. am going to play
2. is having
3. am lighting
- 4.