

1º Usando la base de datos de productos. Buscar todos los elementos "producto" que tengan un precio mayor a 1000:

```
<producto>
<nombreproducto>Portatil lenovo x1 Carbon</nombreproducto>
<fabricante>Lenovo</fabricante>
<precio>1200</precio>
</producto>
<producto>
<nombreproducto>Tablet Apple</nombreproducto>
<fabricante>Apple</fabricante>
<precio>1100</precio>
</producto>
<producto>
<nombreproducto>Portatil lenovo x1 Carbon</nombreproducto>
<fabricante>Lenovo</fabricante>
<precio>1200</precio>
</producto>
<producto>
<nombreproducto>Tablet Apple</nombreproducto>
<fabricante>Apple</fabricante>
<precio>1100</precio>
</producto>
<producto>
<nombreproducto>Portatil lenovo x1 Carbon</nombreproducto>
<fabricante>Lenovo</fabricante>
<precio>1200</precio>
</producto>
```

2º Buscar el nombre del fabricante del primer producto de la lista

```
1 Result, 31 b
<fabricante>Lenovo</fabricante>
```

3º Buscar todos los productos que sean fabricados por Apple o Samsung

```
OK
3 Results, 379 b
Resu

<producto>
<nombreproducto>Tablet Apple</nombreproducto>
<fabricante>Apple</fabricante>
<precio>1100</precio>
</producto>
<producto>
<nombreproducto>Tablet Apple</nombreproducto>
<fabricante>Apple</fabricante>
<precio>1100</precio>
</producto>
<producto>
<nombreproducto>Tablet Apple</nombreproducto>
<fabricante>Apple</fabricante>
<precio>1100</precio>
</producto>
```

4º Buscar el precio de todos los productos que tengan un nombre de producto que contenga la palabra "Tablet":

The screenshot shows the XMLSpy application interface. The top toolbar includes icons for file operations, editing, and navigation. The 'Find' toolbar at the top has a search icon and a text input field containing the XPath expression: `/productos/producto[contains(nombreproducto,"Tablet")]/precio`. Below this, the 'Find contents...' dropdown is set to `*.*`. The left pane displays a file tree for the project 'phoneland', with 'producto.xml' selected. The right pane shows the XML content of 'producto.xml' with line numbers 69 through 81. The XML structure is as follows:

```
69 <producto>
70 <nombreproducto>Smartphone San:
  nombreproducto>
71 <fabricante>Samsung</fabricante>
72 <precio>900</precio>
73 </producto>
74
75 <producto>
76 <nombreproducto>Monnitor LG</n
  nombreproducto>
77 <fabricante>LG</fabricante>
78 <precio>100</precio>
79 </producto>
80
81 <producto>
```

Below the XML view, a status bar indicates '3 Results, 67 b'. The 'Result' pane at the bottom shows the search results, displaying the price element for each match:

```
<precio>1100</precio>
<precio>1100</precio>
<precio>1100</precio>
```

5° Buscar todos los nombres de los productos que tengan un precio menor a 500:

Find

C:/baseX/baseX/data/phone

.

Find contents...

atv.baseX · phoneland/data/baseX
atvl.baseX · phoneland/data/baseX
atvr.baseX · phoneland/data/baseX
inf.baseX · phoneland/data/baseX
producto.xml · phoneland/data/baseX
tbl.baseX · phoneland/data/baseX
tblt.baseX · phoneland/data/baseX
txt.baseX · phoneland/data/baseX
txtl.baseX · phoneland/data/baseX
txtr.baseX · phoneland/data/baseX

phoneland

producto.xml

```

69 <producto>
70 <nombreproducto>Smart;
   nombreproducto>
71 <fabricante>Samsung</f
72 <precio>900</precio>
73 </producto>
74
75 <producto>
76 <nombreproducto>Monni
77 <fabricante>LG</fabri
78 <precio>100</precio>
79 </producto>
80
81 </producto>

```

OK

6 Results, 271 b

Res

```

<nombreproducto>Monnitor LG</nombreproducto>
<nombreproducto>Altavoz Bm</nombreproducto>
<nombreproducto>Monnitor LG</nombreproducto>
<nombreproducto>Altavoz Bm</nombreproducto>
<nombreproducto>Monnitor LG</nombreproducto>
<nombreproducto>Altavoz Bm</nombreproducto>

```

6. Palabras que contengan "e"

The screenshot shows a file search interface with the following components:

- Search Bar:** Contains the query `/productos/producto[contains(nombreproducto,"e")]`.
- File List:** Displays a list of files in the directory `C:/baseX/baseX/data/phone`. The file `producto.xml` is selected.
- Search Results:** Shows the XML content of `producto.xml` with line numbers 69 through 81. The XML structure is as follows:

```
<producto>
<nombreproducto>Smartphone
nombreproducto>
<fabricante>Samsung</fabricante>
<precio>900</precio>
</producto>

<producto>
<nombreproducto>Monitor
<fabricante>LG</fabricante>
<precio>100</precio>
</producto>

<producto>
```
- Summary:** At the bottom, it indicates "9 Results, 1216 b".
- Result Preview:** A detailed view of the XML results, showing the following structure:

```
<producto>
<nombreproducto>Portatil lenovo x1 Carbon</nombreproducto>
<fabricante>Lenovo</fabricante>
<precio>1200</precio>
</producto>

<producto>
<nombreproducto>Smartphone Samsung S21</nombreproducto>
<fabricante>Samsung</fabricante>
<precio>900</precio>
</producto>

<producto>
<nombreproducto>Tablet Apple</nombreproducto>
<fabricante>Apple</fabricante>
<precio>1100</precio>
</producto>

<producto>
<nombreproducto>Portatil lenovo x1 Carbon</nombreproducto>
<fabricante>Lenovo</fabricante>
<precio>1200</precio>
```

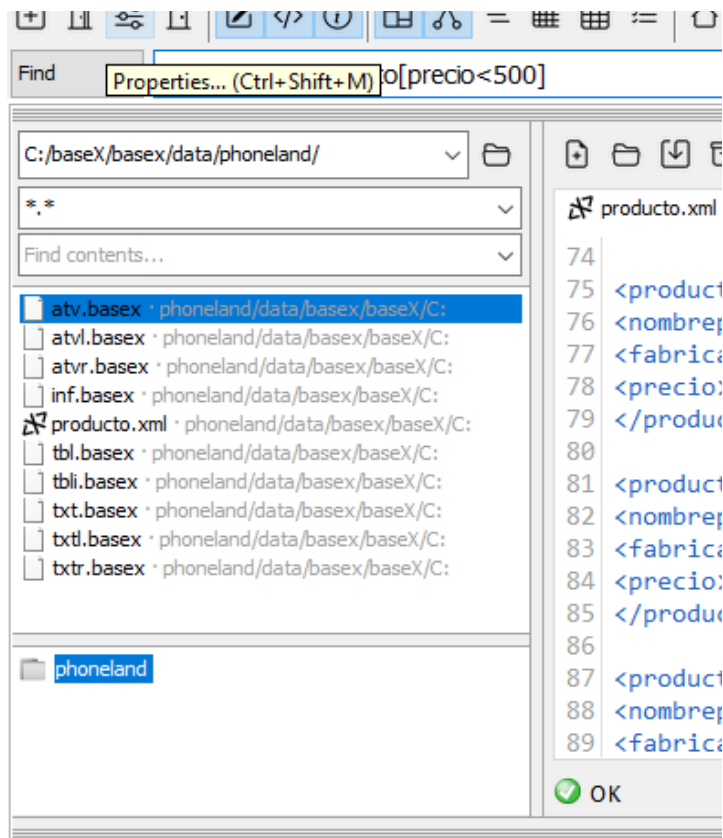
7. Palabras que terminan en "n".

The screenshot shows a database search interface. At the top, a search bar contains the query: `/productos/producto[substring(nombreproducto, string-length(nombreproducto))="n"]`. Below the search bar, a file explorer on the left shows a directory structure with files like `atv.base`, `atvl.base`, `atvr.base`, `inf.base`, `producto.xml`, `tbl.base`, `tbli.base`, `txt.base`, and `txtr.base`. The main window displays XML data for `producto.xml`, showing product details like `<nombreproducto>Smartphone Samsung S21</nombreproducto>` and `<precio>900</precio>`. A status bar at the bottom indicates "OK".

Below the main window, a "Result" panel shows the search results. It displays the XML structure for the products found, including `<nombreproducto>Portatil lenovo x1 Carbon</nombreproducto>` and `<precio>1200</precio>`. The panel also includes a summary of the results:

- Optimizing:**
 - rewrite context value: . ->
 - rewrite util:root(nodes): ut
- Optimized Query:**
`db:get-pre("phoneland", 0)/nombreproducto, string-ler`
- Query:**
`/productos/producto[subst`
`nombreproducto))="n"]`
- Result:**
 - Hit(s): 3 Items
 - Updated: 0 Items
 - Printed: 421 b
 - Read Locking: phoneland
 - Write Locking: (none)

8.-Mostrar precio menor de 500



6 Results, 736 b

```
<product>
<nombreproducto>Monnitor LG</nombreproducto>
<fabricante>LG</fabricante>
<precio>100</precio>
</product>
<product>
<nombreproducto>Altavoz Bm</nombreproducto>
<fabricante>Sonos</fabricante>
<precio>400</precio>
</product>
<product>
<nombreproducto>Monnitor LG</nombreproducto>
<fabricante>LG</fabricante>
<precio>100</precio>
</product>
<product>
<nombreproducto>Altavoz Bm</nombreproducto>
<fabricante>Sonos</fabricante>
<precio>400</precio>
</product>
<product>
<nombreproducto>Monnitor LG</nombreproducto>
<fabricante>LG</fabricante>
<precio>100</precio>
</product>
```

9.-Precio de Apple que son mayores de 500 euros.

The image shows a file explorer window with the search path `/productos/producto[fabricante="Apple" and precio>500]`. The file list shows several files, including `producto.xml`. The XML editor displays the content of `producto.xml`, showing three product entries. The first entry is for a product named "Monni" with a price of 100. The second entry is for a product named "Tablet" with a price of 1100. The third entry is for a product named "Altav" with a price of 1100. The search results show 3 results, 379 b.

```
<producto>
<nombreproducto>Monni
<fabricante>LG</fabri
<precio>100</precio>
</producto>

<producto>
<nombreproducto>Tablet
<fabricante>Apple</fa
<precio>1100</precio>
</producto>

<producto>
<nombreproducto>Altav
<fabricante>Sonos</fa
```

3 Results, 379 b

```
<producto>
<nombreproducto>Tablet Apple</nombreproducto>
<fabricante>Apple</fabricante>
<precio>1100</precio>
</producto>
<producto>
<nombreproducto>Tablet Apple</nombreproducto>
<fabricante>Apple</fabricante>
<precio>1100</precio>
</producto>
<producto>
<nombreproducto>Tablet Apple</nombreproducto>
<fabricante>Apple</fabricante>
<precio>1100</precio>
</producto>
```

10.-Búsqueda del ultimo,

