

# EXAMEN LENGUAJES DE MARCAS

CHRISTIAN MILLÁN SORIA

1º DAW TARDE

## 1. Ejercicio de XPATH. Partimos del fichero "Universidad.xml":

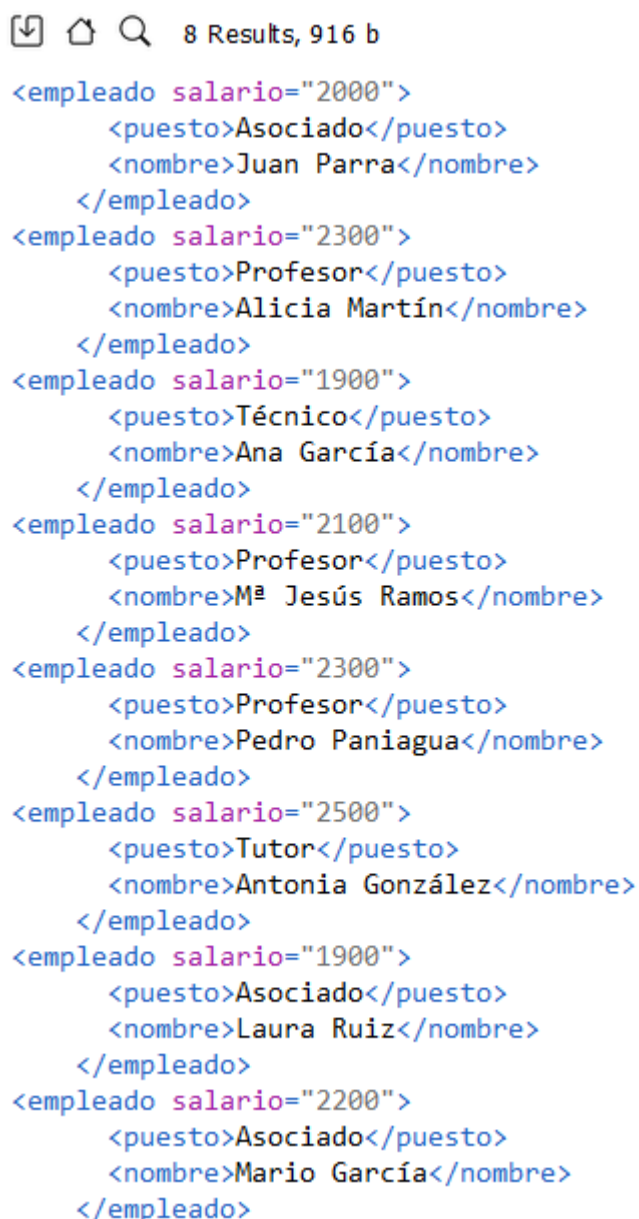
```
<universidad>
  <departamento telefono="112233" tipo="A">
    <codigo>IFC1</codigo>
    <nombre>Informática</nombre>
    <empleado salario="2000">
      <puesto>Asociado</puesto>
      <nombre>Juan Parra</nombre>
    </empleado>
    <empleado salario="2300">
      <puesto>Profesor</puesto>
      <nombre>Alicia Martín</nombre>
    </empleado>
  </departamento>
  <departamento telefono="990033" tipo="A">
    <codigo>MAT1</codigo>
    <nombre>Matemáticas</nombre>
    <empleado salario="1900">
      <puesto>Técnico</puesto>
      <nombre>Ana García</nombre>
    </empleado>
    <empleado salario="2100">
      <puesto>Profesor</puesto>
      <nombre>Mª Jesús Ramos</nombre>
    </empleado>
    <empleado salario="2300">
      <puesto>Profesor</puesto>
      <nombre>Pedro Paniagua</nombre>
    </empleado>
    <empleado salario="2500">
      <puesto>Tutor</puesto>
      <nombre>Antonia González</nombre>
    </empleado>
  </departamento>
  <departamento telefono="880833" tipo="B">
    <codigo>MAT2</codigo>
    <nombre>Análisis</nombre>
    <empleado salario="1900">
      <puesto>Asociado</puesto>
      <nombre>Laura Ruiz</nombre>
    </empleado>
    <empleado salario="2200">
      <puesto>Asociado</puesto>
```

```
<nombre>Mario García</nombre>
</empleado>
</departamento>
</universidad>
```

Realiza las siguientes cuestiones usando XPATH y especifica la ruta y el resultado obtenido para calcular las siguientes cuestiones:

1. Los datos con etiquetas de los empleados que tengan el atributo salario.

```
/universidad/departamento/empleado[@salario]
```





8 Results, 916 b

```
<empleado salario="2000">
  <puesto>Asociado</puesto>
  <nombre>Juan Parra</nombre>
</empleado>
<empleado salario="2300">
  <puesto>Profesor</puesto>
  <nombre>Alicia Martín</nombre>
</empleado>
<empleado salario="1900">
  <puesto>Técnico</puesto>
  <nombre>Ana García</nombre>
</empleado>
<empleado salario="2100">
  <puesto>Profesor</puesto>
  <nombre>Ma Jesús Ramos</nombre>
</empleado>
<empleado salario="2300">
  <puesto>Profesor</puesto>
  <nombre>Pedro Paniagua</nombre>
</empleado>
<empleado salario="2500">
  <puesto>Tutor</puesto>
  <nombre>Antonia González</nombre>
</empleado>
<empleado salario="1900">
  <puesto>Asociado</puesto>
  <nombre>Laura Ruiz</nombre>
</empleado>
<empleado salario="2200">
  <puesto>Asociado</puesto>
  <nombre>Mario García</nombre>
</empleado>
```

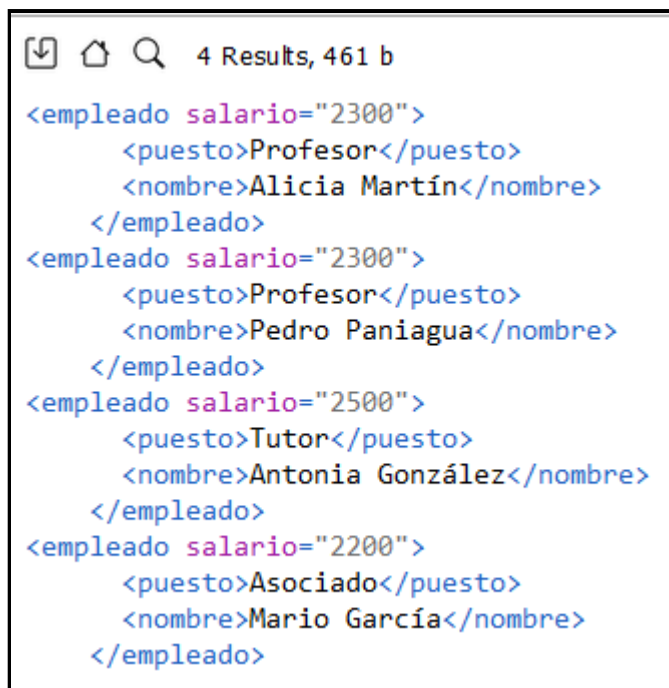
2. Los datos, y solo los datos, sin etiquetas, de los departamentos cuyo teléfono sea 990033.

```
/universidad/departamento[@telefono='990033']//text()
```

		29 Results, 287 b
MAT1		
Matemáticas		
Técnico		
Ana García		
Profesor		
M <sup>a</sup> Jesús Ramos		
Profesor		
Pedro Paniagua		

### 3. Los datos con etiquetas de los empleados cuyo salario sea mayor de 2100.

```
/universidad/departamento/empleado[@salario>2100]
```



```

4 Results, 461 b

<empleado salario="2300">
  <puesto>Profesor</puesto>
  <nombre>Alicia Martín</nombre>
</empleado>
<empleado salario="2300">
  <puesto>Profesor</puesto>
  <nombre>Pedro Paniagua</nombre>
</empleado>
<empleado salario="2500">
  <puesto>Tutor</puesto>
  <nombre>Antonia González</nombre>
</empleado>
<empleado salario="2200">
  <puesto>Asociado</puesto>
  <nombre>Mario García</nombre>
</empleado>

```

4. Devuelve la suma total del salario de todos los empleados.

```
sum(//empleado/@salario)
```



```

1 Result, 5 b

17200

```

2. Ejercicio de XQUERY. Dado el siguiente fichero XML llamado "ejercicio2.xml":

```

<?xml version="1.0" encoding="UTF-8"?>
<marvel>
  <superheroe nombre="Thor" poderes="Martillo mágico, Dios, Fuerza" amigos="Iron Man y Hulk" nivel="7"/>
  <superheroe nombre="Hulk" poderes="Superfuerza, indestructible" amigos="Iron Man y Thor" nivel="8"/>
  <superheroe nombre="Iron Man" poderes="Armadura, intelecto superior" amigos="Hulk y Thor" nivel="6"/>
  <superheroe nombre="Capitán América" poderes="fuerza, resistencia, superguerrero" amigos="Iron Man" nivel="6"/>
  <superheroe nombre="Viuda Negra" poderes="coordinación, luchadora" amigos="No tiene" nivel="6"/>
  <superheroe nombre="Nick Furia" poderes="coordinación, luchadora" amigos="No tiene" nivel="6"/>
</marvel>

```

Explica qué se está realizando en esta consulta de XQUERY. Explica sentencia a sentencia y qué resultado es el que podría salir. Pon un ejemplo de resultado a salir:

## Ejercicio A

```
for $superheroe in doc('ejercicio2.xml')/marvel/superheroe
where ends-with($superheroe/@nombre, 'a')
return
  <superheroe>{$superheroe}</superheroe>
```

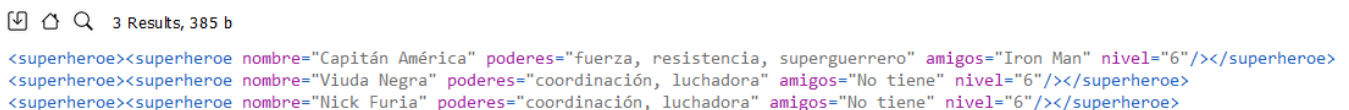
La anterior consulta realiza una búsqueda en un documento XML llamado "ejercicio2.xml". La sentencia "for \$superheroe in doc('ejercicio2.xml')/marvel/superheroe" establece una variable llamada "\$superheroe" que se utilizará para iterar sobre cada elemento < superheroe > dentro del elemento < marvel > en el documento XML.

Luego, se utiliza la sentencia "where ends-with(\$superheroe/@nombre, 'a')" para filtrar los elementos < superheroe >. La función "ends-with()" se utiliza para verificar si el valor del atributo "nombre" de cada elemento termina con la letra "a". Esto significa que solo se seleccionarán los superhéroes cuyos nombres terminen en "a".

Finalmente, la sentencia "return < superheroe >{ \$superheroe }" genera un nuevo elemento < superheroe > para cada resultado obtenido. El contenido de cada elemento < superheroe > seleccionado se incluirá dentro de este nuevo elemento.

Como resultado de esta consulta, obtendremos un conjunto de elementos < superheroe >, cada uno con el contenido de un superhéroe cuyo nombre termina en "a".

Un resultado posible es el siguiente:



```
3 Results, 385 b
<superheroe><superheroe nombre="Capitán América" poderes="fuerza, resistencia, superguerrero" amigos="Iron Man" nivel="6"/></superheroe>
<superheroe><superheroe nombre="Viuda Negra" poderes="coordinación, luchadora" amigos="No tiene" nivel="6"/></superheroe>
<superheroe><superheroe nombre="Nick Furia" poderes="coordinación, luchadora" amigos="No tiene" nivel="6"/></superheroe>
```

## Ejercicio B

```
for $superheroe in doc("ejercicio2.xml")//superheroe
return
  <html>
    <table border="2px" margin="2px" align="center">
      <tr>
        <th colspan="2" align="left">{data($superheroe/@nombre)}</th>
      </tr>
      <tr><td>Nombre:</td><td>{data($superheroe/@poderes)}</td></tr>
      <tr><td>Amigos:</td><td>{data($superheroe/@amigos)}</td></tr>
      <tr><td>Nivel:</td><td>{data($superheroe/@nivel)}</td></tr>
    </table>
    <br></br>
  </html>
```

Esta segunda consulta crea una variable "\$superheroe", al igual que en el ejercicio anterior.

El resultado de esta consulta XQUERY sería una serie de bloques de código HTML, cada uno representando una tabla con la información de un superhéroe. Cada tabla tendría filas que incluyen el nombre del superhéroe, sus poderes, sus amigos y su nivel. Estas tablas estarían separadas por saltos de línea en formato HTML.

6 / 10

```

        </li>
      </xsl:if>
    </xsl:for-each>
  </ul>
</body>
</html>
</xsl:template>
</xsl:stylesheet>

```

Respuesta:

```

<?xml version="1.0" encoding="UTF-8"?>
<ciclos>
  <ciclo grado="Superior" codigo="grado_sup1">
    <nombre>Desarrollo de Aplicaciones Web</nombre>
  </ciclo>

  <ciclo grado="Superior" codigo="grado_sup2">
    <nombre>Desarrollo de Aplicaciones Multiplataforma</nombre>
  </ciclo>

  <ciclo grado="Superior" codigo="grado_sup3">
    <nombre>Administración de Sistemas Informáticos en Red</nombre>
  </ciclo>

  <ciclo grado="Medio" codigo="grado_med1">
    <nombre>Sistemas Microinformáticos y Redes</nombre>
  </ciclo>
</ciclos>

```

b. Tras realizar el proceso de transformación, explica y compón (con datos) lo que saldría tras el procedimiento de transformación con el fichero XSLT.

## GRADO SUPERIOR

- grado\_sup1Desarrollo de Aplicaciones Web
- grado\_sup2Desarrollo de Aplicaciones Multiplataforma
- grado\_sup3Administración de Sistemas Informáticos en Red

Para empezar, debido a cómo está escrito el XSLT, el resultado de cada elemento de la lista aparece mal, ya que salen los códigos y los nombres de los grados pegados.

\*El XSLT está copiado literalmente del examen.

Mi XML contiene 4 grados: 3 superiores y 1 medio. El resultado es el esperado, ya que solo se muestran los grados que contienen un atributo con el valor "Superior".

Debido a cómo está formado el XSLT, los datos "código" y "nombre" aparecen pegados, pero podría cambiarse el XSLT para que aparezcan mejor ordenados.

Se muestra un título que dice "GRADO SUPERIOR" y una lista desordenada ("ul") que se amplía conforme encuentra un elemento que contiene el atributo mencionado anteriormente.

Se excluyen los elementos con atributos que tengan otro valor ("Medio", por ejemplo) o no tengan atributo "codigo".

**c. ¿Cuál es la transformación en el código XSLT si quisiéramos tener una tabla con el contenido del proceso de transformación? Realiza una pequeña modificación añadiendo, al menos, tres propiedades de CSS en las etiquetas que prefieras.**

Para transformar el código XSLT para que genere una tabla con el contenido del proceso de transformación, he cambiado el código de la siguiente manera:

```
<?xml version="1.0" encoding="UTF-8"?>
<html>
  <head>
    <style>
      table{
        border-collapse: collapse;
        width: 100%;
        font-family: Arial, sans-serif;
      }
      th, td{
        text-align: left;
        padding: 10px;
        border: 1px solid #ccc;
      }
      th{
        background-color: #4CAF50;
        color: white;
        font-weight: bold;
      }
      tr:nth-child(even){
        background-color: #f2f2f2;
      }
      tr:hover{
        background-color: #ddd;
      }
      td{
        border-bottom: 1px solid #ddd;
      }
      h1{
        color: #333;
        font-size: 24px;
        margin-bottom: 20px;
      }
      .highlight{
        background-color: yellow;
        font-weight: bold;
      }
```



```
    }
    </style>
</head>
<body>
  <h1>GRADO SUPERIOR</h1>
  <table>
    <tr>
      <th>Código</th>
      <th>Nombre</th>
    </tr>
    <tr>
      <td>grado_sup1</td>
      <td class="highlight">Desarrollo de Aplicaciones Web</td>
    </tr>
    <tr>
      <td>grado_sup2</td>
      <td>Desarrollo de Aplicaciones Multiplataforma</td>
    </tr>
    <tr>
      <td>grado_sup3</td>
      <td>Administración de Sistemas Informáticos en Red</td>
    </tr>
  </table>
</body>
</html>
```

GRADO SUPERIOR	
Código	Nombre
grado_sup1	Desarrollo de Aplicaciones Web
grado_sup2	Desarrollo de Aplicaciones Multiplataforma
grado_sup3	Administración de Sistemas Informáticos en Red

4. Ejercicio de JavaScript.

Realiza un pequeño programa en JavaScript que inicialmente tenga este aspecto:

Escribe tu nombre:  Pulsa aquí para saludo inicial Pulsa aquí para saludo final

a. Tras presionar con un click el botón, recibiremos el mensaje tal cual en la página web.

Hola, buenos dias Rafa

b. Tras doble click en el botón, recibiremos el mensaje tal cual en la página web:

Hola, buenos noches Rafa

Respuesta:



**c. Inserta ahora una lista de selección con tres opciones: mañana, tarde y noche y cuando cambiemos de opción saldrá una alerta diciendo que has cambiado de franja horaria.**

