LENGUAJES DE CONSULTAS XPATH Y XQUERY

Ambos lenguajes son estándares para acceder y obtener datos desde documentos XML, estos lenguajes tienen en cuenta que la información en los documentos está semiestructurada o jerarquizada como árbol.

XPath es el lenguaje de rutas XML, se utiliza para navegar dentro de la estructura jerárquica de un fichero XML.

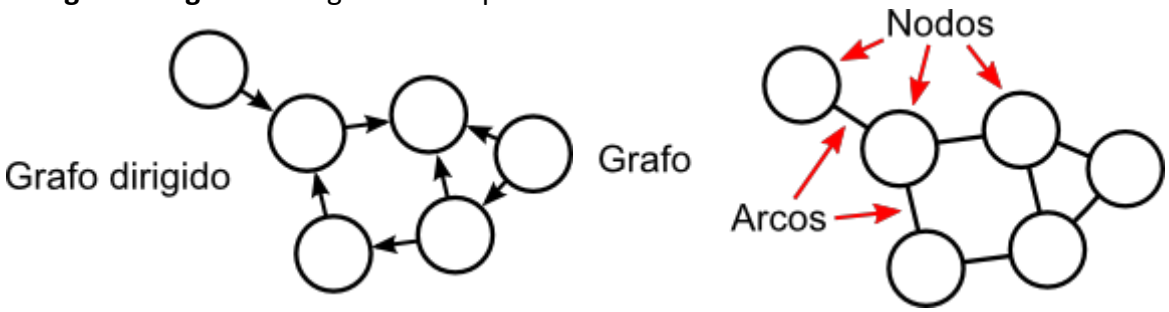
XQuery es a XML lo mismo que SQL es a las bases de datos relacionales, es decir, es un lenguaje de consulta diseñado para consultar documentos XML. Abarca desde ficheros XML hasta bases de datos relacionales con funciones de conversión de registros a XML. XQuery contiene a XPath, toda expresión de consulta en XPath es válida en XQuery, pero XQuery permite mucho más.

http://www.tic2.org/WebTecnica/Programacion/XPath/XPathFunciones/XPathFunciones.htm

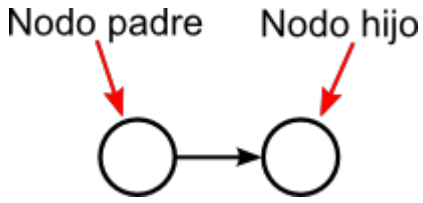
XPATH

XPath considera un documento XML como un árbol de nodos. En Informática, un árbol es una estructura de datos que equivale a un árbol matemático. En Matemáticas un árbol es un caso particular de grafo. Los siguientes términos definidos en teoría de grafos se utilizan también en Informática y en XPath:

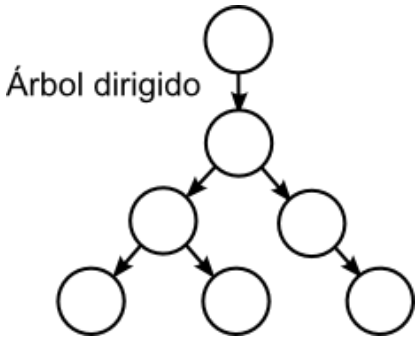
- Un **grafo** es un conjunto de objetos llamados nodos o vértices unidos por enlaces llamados arcos o aristas. Un **grafo dirigido** es un grafo en el que los arcos tienen dirección.

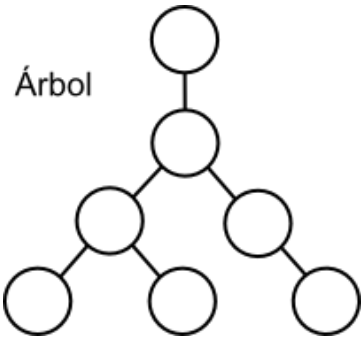


- Cuando dos nodos están unidos por un arco con dirección, el **nodo padre** es el nodo del que parte el arco y el **nodo hijo** es el nodo al que llega el arco.

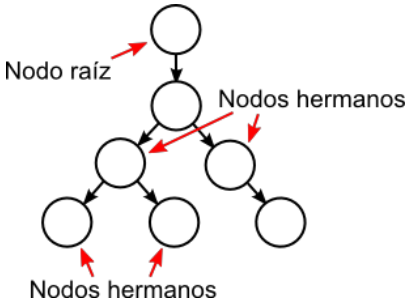


- Un **árbol** es un grafo en el que cualquier pareja de vértices están conectada por un único camino (es decir, que no hay ciclos). Un **árbol dirigido** es un árbol en el que las aristas tienen dirección y todos los nodos menos uno tienen un único padre.

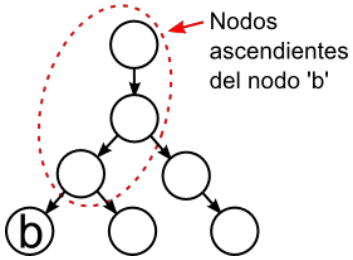


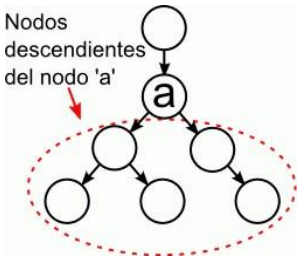


x El **nodo raíz** de un árbol dirigido es el único nodo sin padre. Los **nodos hermanos** son los nodos que tienen el mismo padre.



x Los **nodos descendientes** de un nodo son todos los nodos a los que se llega desde el nodo: los hijos, los hijos de los hijos, etc. Los **nodos ascendientes** de un nodo son todos los nodos de los que un nodo es descendiente: el padre, el padre del padre, etc.





Tipos de nodos

Un documento XML puede representarse como un árbol dirigido, considerando por ejemplo los elementos como nodos y que un elemento es padre de los elementos que contiene. Pero en XPath no sólo los elementos son nodos, en realidad hay siete tipos de nodos:

x **Raíz** es la raíz del árbol se representa por **/**.

x **Nodo raíz**: es la raíz del árbol se representa por **/**.

x **Nodos elementos**: cualquier elemento de un documento XML, son las etiquetas del árbol. x **Nodos texto**: los caracteres que están entre las etiquetas.

x **Nodos atributo**: son como propiedades añadidas a los nodos elementos, se representan con @.

x **Nodos comentario**: las etiquetas de comentario. x **Nodos espacio de nombres**: contienen espacios de nombres.

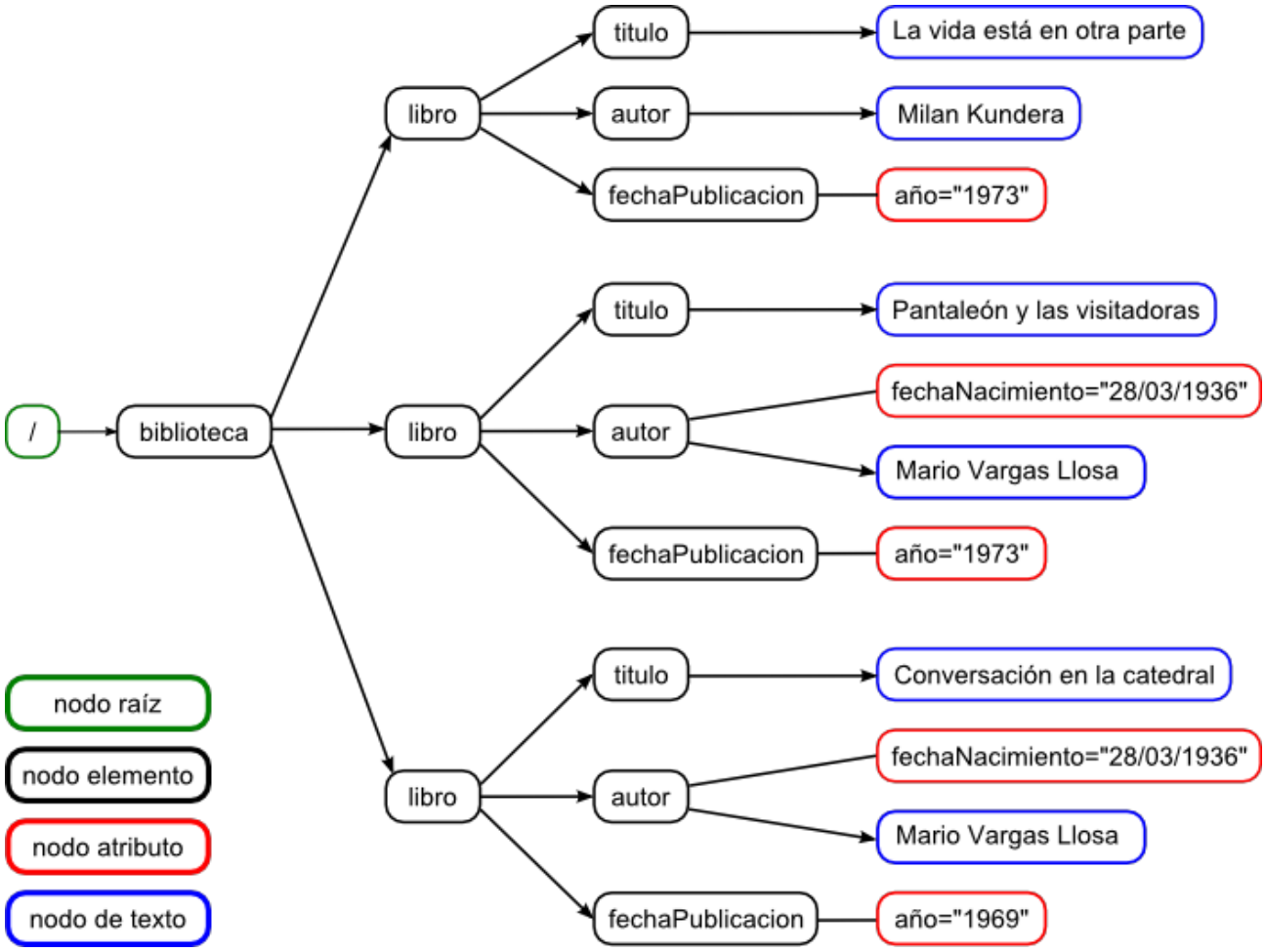
x **Nodos instrucción de proceso**: contienen instrucciones de proceso, van entre las etiquetas <?.....?>.

Nota: La declaración DOCTYPE no se considera como nodo.

Por ejemplo, el documento XML siguiente:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<biblioteca>
  <libro>
    <titulo>La vida está en otra parte</titulo>
    <autor>Milan Kundera</autor>
    <fechaPublicacion año="1973"/>
  </libro>
  <libro>
    <titulo>Pantaleón y las visitadoras</titulo>
    <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor>
    <fechaPublicacion año="1973"/>
  </libro>
  <libro>
    <titulo>Conversación en la catedral</titulo>
    <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor>
    <fechaPublicacion año="1969"/>
  </libro>
</biblioteca>
```

se puede representar mediante el siguiente grafo:



Los nodos atributos y de texto no son como los nodos elemento. Por ejemplo, los nodos atributo y de texto no pueden tener descendientes. En realidad el nodo atributo ni siquiera se considera como hijo, sino como una etiqueta adosada al elemento. El texto contenido por una etiqueta sí que se considera hijo

del elemento, aunque las expresiones XPath suelen trabajar con nodos elementos y para referirse a los atributos o al texto se utilizan notaciones especiales.

Otro ejemplo. Dado el siguiente documento XML:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<universidad>
<espacio xmlns=http://www.misitio.com xmls:prueba=http://www.misitio.com/pruebas
/>
      <!—DEPARTAMENTO 1 -->
      <departamento telefono=”112233” tipo=”A”>
            <codigo>IFC1</codigo>
            <nombre>Informática</nombre>
      </departamento>
      <!—DEPARTAMENTO 2 -->
      <departamento telefono=”223344” tipo=”A”>
            <codigo>MAT1</codigo>
            <nombre>Matemáticas</nombre>
      </departamento>
      <!—DEPARTAMENTO 3 -->
      <departamento telefono=”334455” tipo=”B”> <codigo>MAT2</codigo>
            <nombre>Análisis</nombre>
      </departamento>
</universidad>
```

Nos encontramos con los siguientes tipos de nodos:

TIPO NODO	EJEMPLO
Elemento	<universidad>, <departamento>, <codigo>, <nombre>
Texto	IFC1, Informática, MAT1, Matemáticas, MAT2, Análisis
Atributo	telefono=”112233”, tipo=”A”, telefono=”223344”, tipo=”A”, telefono=”334455”, tipo=”B”,
Comentario	<!—DEPARTAMENTO 1 -->, <!—DEPARTAMENTO 2 -->, <!—DEPARTAMENTO 3 -->
Espacio de nombres	<espacio xmlns=http://www.misitio.com xmls:prueba=http://www.misitio.com/pruebas />
Instrucción de proceso	<?xml version="1.0" encoding="ISO-8859-1"?>

SINTAXIS DE LA EXPRESIONES XPATH

Una expresión XPath es una cadena de texto que representa un recorrido en el árbol del documento. Las expresiones más simples se parecen a las rutas de los archivos en el explorador de Windows o en la shell de GNU/Linux.

Por ejemplo, el siguiente *path* en Windows:

```
C:\Users\Usuario\Documents
```

hace referencia a un único directorio: `Documents` el cual cuelga del conjunto de directorios `C:\Users\Usuario.`

Sin embargo, la siguiente expresión en XPath:

```
/biblioteca/libro/titulo
```

hace referencia a TODOS los elementos `titulo` que cuelguen directamente de CUALQUIER elemento `libro` que cuelgue de CUALQUIER elemento `biblioteca` que, finalmente, cuelguen del nodo raíz, `/`.

Evaluar una expresión XPath es buscar si hay nodos en el documento que se ajustan al recorrido definido en la expresión. El resultado de la evaluación son todos los nodos que se ajustan a la expresión. Para poder evaluar una expresión XPath, el documento debe estar bien formado.

Las expresiones XPath se pueden escribir de dos formas distintas:

- sintaxis abreviada: más compacta y fácil de leer
  `/biblioteca/libro`
- sintaxis completa: más larga pero con más opciones disponibles
  `[root-node]/child::{}biblioteca/child::{}libro`

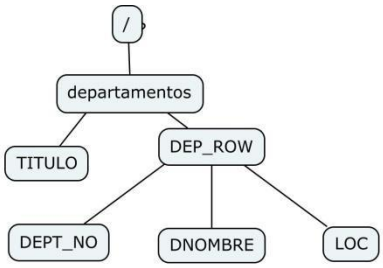
  Devuelven las carreras que tenemos en el fichero Universiadad.xml, la primera es la forma abreviada y la segunda es la completa.

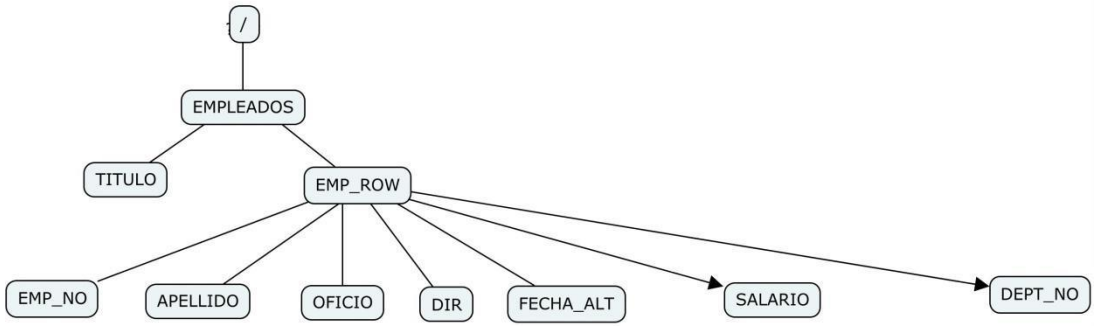
Las expresiones XPath se pueden dividir en pasos de búsqueda. Cada paso de búsqueda se puede a su vez dividir en tres partes:

- eje: indica el nodo o los nodos en los que se realiza la búsqueda
- nodo de comprobación: especifica el nodo o los nodos seleccionados dentro del eje x
  predicado: permite restringir los nodos de comprobación

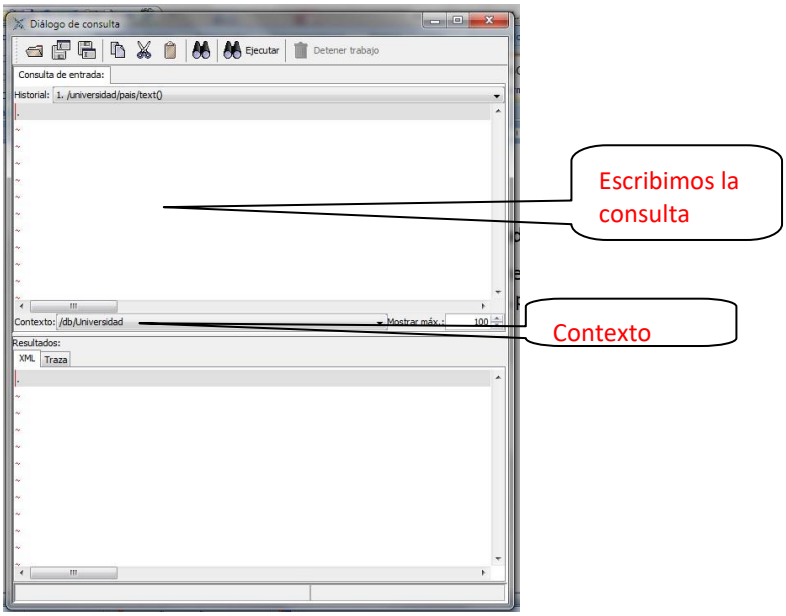
Ejemplos XPATH utilizando una sintaxis abreviada:

A partir de la colección **Pruebas**, que contiene los documentos departamentos.xml y empleados.xml, cuyas estructuras de árbol son las siguientes:





Realizaremos las siguientes consultas desde Query Dialog (Diálogo de Consulta)



Ejes

1º) / si está al principio de la expresión, indica el nodo raíz, si no, indica "hijo".

CONSULTA	RESULTADO
/	Si está dentro del contexto /db/Pruebas devuelve todos los datos de los departamentos y los empleados, es decir, incluye todas las etiquetas que cuelgan del nodo departamentos y del nodo EMPLEADOS, que están dentro de la colección Pruebas.
/APELLIDO	No devuelve nada porque APELLIDO no es hijo del nodo raíz
/departamentos	Devuelve todos los datos de los departamentos, es decir incluye todas las etiquetas que cuelgan del nodo raíz departamentos.

/departamentos/DEP_ROW	Devuelve todas las etiquetas dentro de DEP_ROW
/departamentos/D_NOMBRE	No devuelve nada porque D_NOMBRE no es hijo del nodo departamentos

2º) **//** indica "descendiente" (hijos, hijos de hijos, etc.)

CONSULTA	**RESULTADO**
/departamentos//DNOMBRE	Devuelve los nombres de los departamentos. <DNOMBRE>CONTABILIDAD</DNOMBRE> <DNOMBRE>INVESTIGACION</DNOMBRE> <DNOMBRE>VENTAS</DNOMBRE> <DNOMBRE>PRODUCCION</DNOMBRE>
//DNOMBRE	Devuelve los nombres de los departamentos
//DNOMBRE/LOC	No devuelve nada porque "LOC" no es descendiente de DNOMBRE
//DEPT_NO	Devuelve todos los números de departamentos entre etiquetas. Devuelve 18 filas en lugar de 4, es porque recoge todos los elementos DEPT_NO de la colección y en la colección en el documento empleados.xml tiene 14 empleados con su DEPT_NO.

3º) **@atributo**: selecciona el atributo

CONSULTA
/universidad/carreras/carrera/@id Da error. @id no tiene ningún nodo padre.
/universidad/alumnos/estudios/asignaturas/asignatura/@id No devuelve nada en XPath 1.0 no se pueden seleccionar únicamente el valor del atributo, sino que se obtienen respuestas del tipo nombreDelAtributo=ValorDelAtributo

4º) **|** permite elegir varios recorridos.

CONSULTA	**RESULTADO**
//LOC|//DNOMBRE	Devuelve lo localidad y los nombres de los departamentos <DNOMBRE>CONTABILIDAD</DNOMBRE> <LOC>SEVILLA</LOC> <DNOMBRE>INVESTIGACION</DNOMBRE> <LOC>MADRID</LOC> <DNOMBRE>VENTAS</DNOMBRE> <LOC>BARCELONA</LOC> <DNOMBRE>PRODUCCION</DNOMBRE> <LOC>BILBAO</LOC>

//APELLIDO|//OFICIO|//SALARIO	<APELLIDO>SANCHEZ</APELLIDO> <OFICIO>EMPLEADO</OFICIO> <SALARIO>1040</SALARIO> <APELLIDO>ARROYO</APELLIDO> <OFICIO>VENDEDOR</OFICIO> <SALARIO>1500</SALARIO>

Nodos de comprobación

5º) **node()** selecciona todos los nodos (elementos y texto)

CONSULTA	RESULTADO
//DEP_ROW/node()	Devuelve todas las etiquetas dentro de cada DEP_ROW, no incluye DEP_ROW
//DNOMBRE/node()	Devuelve los nombres de los departamentos sin etiqueta. CONTABILIDAD INVESTIGACION VENTAS PRODUCCION
//DEP_ROW//node()	<DEPT_NO>10</DEPT_NO> 10 <DNOMBRE>CONTABILIDAD</DNOMBRE> CONTABILIDAD <LOC>SEVILLA</LOC> SEVILLA <DEPT_NO>20</DEPT_NO> 20 <DNOMBRE>INVESTIGACION</DNOMBRE> INVESTIGACION <LOC>MADRID</LOC> MADRID <DEPT_NO>30</DEPT_NO> 30 <DNOMBRE>VENTAS</DNOMBRE> VENTAS <LOC>BARCELONA</LOC> BARCELONA <DEPT_NO>40</DEPT_NO> 40 <DNOMBRE>PRODUCCION</DNOMBRE> PRODUCCION <LOC>BILBAO</LOC> BILBAO

6º) **text():** selecciona el contenido del elemento (texto)

CONSULTA	RESULTADO
/departamentos/DEP_ROW/DNOMBRE/text()	Devuelve los nombres de los departamentos de cada DEP_ROW sin etiquetas.
//LOC/text()	Devuelve todas las localidades de toda la colección, solo hay 4 SEVILLA MADRID BARCELONA BILBAO

/departamentos/DEP_ROW/text()	No devuelve nada porque DEP_ROW no contiene texto.

7º) *: selecciona todos los elementos

CONSULTA	RESULTADO
/*/TITULO	Selecciona las etiquetas *TITULO* que se encuentran a 1 nivel de profundidad desde la raíz, en este caso *departamentos y EMPLEADOS*. <TITULO>DATOS DE LA TABLA EMPLE</TITULO> <TITULO>DATOS DE LA TABLA DEPART</TITULO>
/*/*/DEPT_NO	Selecciona las etiquetas *DEPT_NO* que se encuentran a 2 niveles de profundidad desde la raíz en nuestro caso 18.
/departamentos/*	Selecciona las etiquetas que van dentro de la etiqueta *departamentos* y sus subetiquetas
/departamentos//*	Selecciona las etiquetas que van dentro de la etiqueta *departamentos* y sus subetiquetas
/DNOMBRE/*	No devuelve nada porque DNOMBRE sólo contiene texto.

8º) @*: selecciona todos los atributos

CONSULTA	RESULTADO
//@*	Devuelve todos los atributos de todos los nodos
//EMP_ROW/@*	No devuelve nada porque EMP_ROW NO tiene elementos
//alumnos/@*	Devuelve todos los atributos de los alumnos

Predicados

Los predicados permiten restringir los nodos de comprobación, se escriben entre corchetes. 9º) **[@atributo]**: selecciona los elementos que tienen el atributo

CONSULTA	RESULTADO
//asignatura[@codigo]	Devuelve todas los nodos que tengan el atributo código

10º) **[número]**: si hay varios resultados selecciona uno de ellos por número de orden; **last()** selecciona el último de ellos.

CONSULTA	RESULTADO
//carrera[3]	<carrera id="c03"> <nombre>Dipl. Relaciones Laborales</nombre> <plan>2008</plan> <creditos>280</creditos> <centro>Facultad de Ciencias Sociales</centro> <subdirector>Alfonso Martín Luque</subdirector> </carrera>

`//asignatura[last()]`	`<asignatura id="a09" titulacion="c06">`
	`  <nombre>Historia del Pensamiento</nombre>`
	`  <creditos_teoricos>6</creditos_teoricos>`
	`  <creditos_practicos>0</creditos_practicos>`
	`  <trimestre>2</trimestre>`
	`</asignatura>`
	`<asignatura codigo="a09"/>`
	`<asignatura codigo="a01"/>`
	`<asignatura codigo="a09"/>`
	`<asignatura codigo="a07"/>`

`//asignatura[last()-1]`	`<asignatura id="a08" titulacion="c01">`
	`  <nombre>Bases de Datos</nombre>`
	`  <creditos_teoricos>4.5</creditos_teoricos>`
	`  <creditos_practicos>5.5</creditos_practicos>`
	`  <trimestre>1</trimestre>`
	`</asignatura>`
	`<asignatura codigo="a05"/>`
	`<asignatura codigo="a02"/>`
	`<asignatura codigo="a04"/>`
	`<asignatura codigo="a01"/>`

11º) [condición]: selecciona los nodos que cumplen la condición.

Los predicados permiten definir condiciones sobre los valores de los atributos. En las condiciones se pueden utilizar los operadores siguientes: x Operadores lógicos: **or, and** y **not** x Operadores aritméticos: **+**, **-**, ***, **div**, **mod**. x Operadores de comparación: <, **>**, **<=**, **>=**, **=**, **¡=**

Las comparaciones se pueden hacer entre valores de nodos y atributos o con cadenas de texto o numéricas. En el caso de las cadenas de texto deben estar rodeadas por comillas simples o dobles. En el caso de las cadenas numéricas, las comillas son optativas.

La condición puede utilizar el valor de un atributo utilizando @ o el texto que contiene el elemento.

CONSULTA	**RESULTADO**
`/EMPLEADOS/EMP_ROW[DEPT_NO=10]` `//EMP_ROW[DEPT_NO=10]`	Selecciona todos los elementos o nodos (etiquetas) dentro de EMP_ROW de los empleados del departamento 10 DEPT_NO =10
`/EMPLEADOS/EMP_ROW[OFICIO = "VENDEDOR"]`	Selecciona todos los elementos o nodos (etiquetas) dentro de EMP_ROW de los empleados cuyo OFICIO sea VENDEDOR

La condición puede utilizar el valor de un atributo (utilizando @ o el texto que contiene el elemento.

CONSULTA
`//alumno[@beca = "si"]` Selecciona los alumnos que tienen beca
`/universidad/asignaturas/asignatura[@titulacion="c01"]` Selecciona las asignaturas que tienen un atributo titulación = c01
`/universidad/asignaturas/asignatura[@titulacion]` Selecciona las asignaturas que tienen atributo titulación

El predicado puede estar situado en cualquier parte de la expresión XPath, no sólo al final del eje.

CONSULTA
`/universidad/asignaturas/asignatura[@titulacion="c01"]/nombre` Selecciona el nombre de todas las asignaturas cuya titulación se c01

Se pueden escribir varios predicados seguidos, cada uno de los cuales restringe los resultados del anterior, como si estuvieran encadenados por la operación lógica **and**.

CONSULTA
`/universidad/asignaturas/asignatura[@titulacion="c01"][creditos_teoricos>5]` Selecciona las asignaturas de la titulación c01 y con más de 5 créditos teóricos.

Un predicado puede contener condiciones compuestas

CONSULTA
`//asignatura[@titulacion="c01" and trimestre=1]` Selecciona las asignaturas de la titulación c01 que se imparten en el primer trimestre
`/EMPLEADOS/EMP_ROW[OFICIO="VENDEDOR" and SALARIO>1600]` Selecciona los nodos EMPLEADO cuyo OFICIO sea VENDEDOR y tengan un SALARIO mayor de 1600
`//asignatura[@titulacion="c01" or trimestre=1]` Selecciona las asignaturas de la titulación c01 y las asignatura que se imparten en el primer trimestre
`/EMPLEADOS/EMP_ROW[OFICIO="VENDEDOR" or SALARIO>1600]` Selecciona los nodos EMPLEADO cuyo OFICIO sea VENDEDOR y los nodos EMPLEADO que tengan un SALARIO mayor de 1600
`//EMP_ROW[not (DEPT_NO = 10)]` Selecciona todos los EMP_ROW (etiquetas) que NO son del DEPT_NO igual a 10
`//EMP_ROW[not (OFICIO = "ANALISTA")]/APELLIDO/text()` Selecciona los APELLIDOS de los empleados que NO son ANALISTA

Las condiciones pueden hacer referencia al propio elemento, a descendientes o ascendentes, lo que nos permite obtener el mismo resultado de varias maneras distintas.

CONSULTA
`/EMPLEADOS/EMP_ROW[OFICIO="ANALISTA"]/` Selecciona los nodos EMPLEADO cuyo OFICIO sea ANALISTA

`/EMPLEADOS/EMP_ROW/OFICIO[.="ANALISTA"]/..`
Selecciona los elementos OFICIO cuyo contenido sea ANALISTA (para referirse al contenido del propio elemento se utiliza el punto ".") y para obtener los elementos EMP_ROW se sube de nivel con los dos puntos "/.."

CONSULTA
En las siguientes consultas obtiene el nombre de las asignaturas de la titulación c04 de varias formas distintas.
`//@titulacion[.="c04"]/../nombre`
`//asignatura[@titulacion ="c04"]/nombre`
`//nombre[../@titulacion="c04"]`

Ejemplos:

1º) Seleccionar el APELLIDO y el OFICIO de los empleados del DEPT_NO = 10

```
//EMP_ROW[DEPT_NO=10]/APELLIDO | //EMP_ROW[DEPT_NO=10]/OFICIO
```

2º) Obtener el nombre del departamento 10

```
//*[DEPT_NO=10]/DNOMBRE/text()
```

```
//DEP_ROW[DEPT_NO=10]/DNOMBRE/text()
```

3º) Seleccionar los empleados con OFICIO = EMPLEADO

a) `//*[OFICIO="EMPLEADO"]`
b) `/EMPLEADOS/EMP_ROW[OFICIO="EMPLEADO"]`

4º) Seleccionar los datos de los empleados con SALARIO mayor de 1300 y del departamento 10

```
/EMPLEADOS/EMP_ROW[SALARIO>1300 and DEPT_NO=10]
```

5º) Seleccionar el APELLIDO y el OFICIO de los empleados con SALARIO mayor de 1300 y del departamento 20.

```
/EMPLEADOS/EMP_ROW[SALARIO>1300 and DEPT_NO=20]/APELLIDO
|/EMPLEADOS/EMP_ROW[SALARIO>1300 and DEPT_NO=20]/OFICIO
```

Utilización de funciones y expresiones matemáticas

1º) Un número dentro de los corchetes **[número]**representa la posición del elemento en el conjunto seleccionado.

CONSULTA	**RESULTADO**
`/EMPLEADOS/EMP_ROW[1]`	Devuelve todos los datos del primer empleado
`/EMPLEADOS/EMP_ROW[5]/APELLIDO/text()`	Devuelve el APELLIDO del 5º empleado

2º) La función **last()** selecciona el último elemento del conjunto seleccionado.

CONSULTA	**RESULTADO**

/EMPLEADOS/EMP_ROW[last()]	Selecciona todos los datos del último empleado
/EMPLEADOS/EMP_ROW[last()-1]/APELLIDO/text()	Devuelve el APELLIDO del penúltimo empleado

3º) La función **position()** devuelve un número igual a la posición del elemento actual.

CONSULTA	RESULTADO
/EMPLEADOS/EMP_ROW[position()= 3]	Selecciona todos los datos del empleado que ocupa la posición 3.
/EMPLEADOS/EMP_ROW[position()<3]/APELLIDO	Selecciona el APELLIDO de los elementos cuya posición es menor que 3.

4º) La función **count()** cuenta el número de elementos seleccionados.

CONSULTA	RESULTADO
/EMPLEADOS/count(EMP_ROW)	Devuelve el número de empleados
/EMPLEADOS/count(EMP_ROW[DEPT_NO = 10])	Cuenta el número de empleados del departamento 10.
/EMPLEADOS/count(EMP_ROW[OFICIO = "EMPLEADO" and SALARIO > 1600])	Cuenta el número de empleados con OFICIO EMPLEADO y SALARIO mayor de 1600
//*[count(*)=3]	Devuelve elementos que tienen 3 hijos
//*[count(DEP_ROW)=4]	Devuelve los elementos que contienen 4 hijos DEP_ROW, devolverá la etiqueta departamentos y todas las subetiquetas.

5º) La función **sum()** devuelve la suma del elemento seleccionado.

CONSULTA	RESULTADO
sum(/EMPLEADOS/EMP_ROW/SALARIO)	Devuelve la suma de SALARIO.
sum(/EMPLEADOS/EMP_ROW[DEPT_NO = 20]/SALARIO)	Devuelve la suma de SALARIO de los empleados del departamento 20.

Si la etiqueta a sumar la considera *string* hay que convertirla a número utilizando la función **number**.

6º) La función **max()** devuelve el máximo del elemento seleccionado, min() devuelve el mínimo.

CONSULTA	RESULTADO
max(/EMPLEADOS/EMP_ROW/SALARIO)	Devuelve el salario máximo.
min(/EMPLEADOS/EMP_ROW/SALARIO)	Devuelve el salario mínimo.
min(/EMPLEADOS/EMP_ROW[OFICIO = "ANALISTA"]/SALARIO)	Devuelve el salario mínimo de los empleados con OFICIO ANALISTA.

7º) La función **avg()** devuelve la media del elemento seleccionado.

CONSULTA	RESULTADO
`avg(/EMPLEADOS/EMP_ROW/SALARIO)`	Devuelve la media del salario.
`avg(/EMPLEADOS/EMP_ROW[DEPT_NO = 20]/SALARIO)`	Devuelve el salario medio de los empleados del departamento 20.

8º) La función **name()** devuelve el nombre del elemento seleccionado.

CONSULTA	RESULTADO
`//*[name() = 'APELLIDO']`	Devuelve todos los apellidos, entre sus etiquetas .
`count(//*[name()='APELLIDO'])`	Cuenta las etiquetas con nombre APELLIDO

9º) La función **concat(cadena1, cadena2)** concatena las cadenas.

CONSULTA	RESULTADO
`/EMPLEADOS/EMP_ROW[DEPT_NO = 10]/concat(APELLIDO,' - ', OFICIO)`	Devuelve los apellidos y el OFICIO concatenado de los empleados del departamento 10.
`/EMPLEADOS/EMP_ROW/concat(APELLIDO,' - ', OFICIO, ' - ', SALARIO)`	Devuelve los apellidos, el OFICIO y el SALARIO concatenados de todos los empleados.

10º) La función **starts-with(cadena1, cadena2)** es verdadera cuando la cadena1 tiene como prefijo a la cadena2.

CONSULTA	RESULTADO
`/EMPLEADOS/EMP_ROW[startswith(APELLIDO,'A')]`	Obtiene los elementos de los empleados cuyo APELLIDO empieza por 'A'.
`/EMPLEADOS/EMP_ROW[starts-with(OFICIO,'A')]/concat(APELLIDO, ' - ', OFICIO)`	Obtiene APELLIDO y OFICIO concatenados de los empleados cuyo OFICIO empieza por 'A'.

11º) La función **contains(cadena1, cadena2)** es verdadera cuando la cadena1 contiene a la cadena2.

CONSULTA	RESULTADO
`/EMPLEADOS/EMP_ROW[contains(OFICIO,'OR')]/OFICIO`	Devuelve los oficios que contienen la sílaba 'OR'
`/EMPLEADOS/EMP_ROW[contains(APELLIDO,'A')]/APELLIDO`	Devuelve los apellidos que contienen una 'A'

12º) La función **string-length(argumento)** devuelve el número de caracteres de su argumento.

CONSULTA	RESULTADO
`/EMPLEADOS/EMP_ROW/concat(APELLIDO,' - ', string-length(APELLIDO))` Devuelve concatenados el APELLIDO con su número de caracteres	
`/EMPLEADOS/EMP_ROW[string-length(APELLIDO) < 4]` Devuelve los datos de los empleados cuyo APELLIDO tiene menos de 4 caracteres.	

13º) La función **div()** realiza divisiones en punto flotante.

CONSULTA	RESULTADO
`/EMPLEADOS/EMP_ROW/concat(APELLIDO, ', ', SALARIO, ', ', SALARIO div(12))` Devuelve concatenados el APELLIDO el SALARIO y el SALARIO dividido entre 12.	
`sum(/EMPLEADOS/EMP_ROW/SALARIO) div count(/EMPLEADOS/EMP_ROW)` Devuelve la suma de salarios dividida por el contador de empleados	

14º) La función **mod()** calcula el resto de una división.

CONSULTA	RESULTADO
`/EMPLEADOS/EMP_ROW/concat(APELLIDOS, ', ', SALARIO, ', ', SALARIO mod 12)` Devuelve los datos concatenados APELLIDO y SALARIO y el resto de dividir el SALARIO entre 12	
`/EMPLEADOS/EMP_ROW[(SALARIO mod 12) = 4]` Devuelve los datos de los empleados cuando el resto de dividir SALARIO entre 12 sea igual a 4	

Otras funciones

x **data(expresión Xpath)**: devuelve el texto de los nodos de la expresión sin las etiquetas.

x **number(argumento)**: convierte a número el argumento, que puede ser cadena, booleano o un nodo.

x **abs(número)**: devuelve el valor absoluto de número.

x **ceiling(número)**: devuelve el entero más pequeño mayor o igual que la expresión numérica especificada.

x **floor(número)**: devuelve el entero más grande que sea menor o igual que la expresión numérica especificada.

x **round(número)**: redondea el valor de número. x **string(argumento)**: convierte argumento a cadena.

x **compare(expresión1, expresión2)**: compara las dos expresiones, devuelve 0 si son iguales, 1 si expresion1 es mayor que expresion2 y -1 si expresion1 es menor que expresion2.

x **substring(cadena, comienzo, número)**: extrae de cadena, desde la posición indicada en comienzo el número de caracteres indicado en número. x **lower-case(cadena)**: convierte a minúsculas cadena. x **upper-case(cadena)**: convierte a mayúsculas cadena.

x **translate(cadena, caracter1, caracter2)**: reemplaza en cadena1, los caracteres que se indican en caracter1, por los correspondientes que aparecen en carácter2, uno por uno.

x **ends-with(cadena1, cadena2)**: devuelve true si la cadena1 termina en cadena2. x **year-from-date(fecha)**: devuelve el año de la fecha, el formato de fecha es aaaa-mm-dd. x **month-from-date(fecha)**: devuelve el año de la fecha. x **day-from-date(fecha)**: devuelve el día de la fecha.

Para ver más funciones http://www.w3schools.com/xpath/xpath_functions.asp

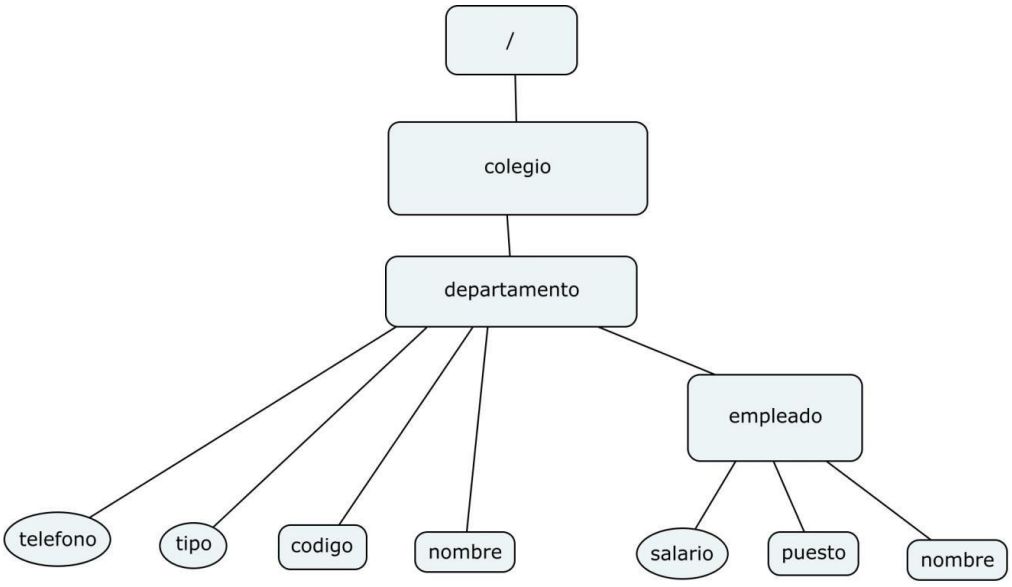
La página en español: http://www.sidar.org/recur/desdi/traduc/es/xml/xpath.html

NODOS ATRIBUTOS XPATH

Un nodo puede tener tantos atributos como se desee y para cada uno se le creará un nodo atributo. Los nodos atributo NO se consideran como hijos, sino más bien como etiquetas añadidas al nodo elemento. Cada nodo atributo consta de un nombre, un valor (que es siempre una cadena) y un posible "espacio de nombres".

Partimos del documento *colegio.xml*. Subimos el documento dentro de la colección Pruebas de la BD.

En el árbol del documento los atributos se representan mediante elipses.



Para referirnos a los atributos de los elementos se usa @ antes del nombre, por ejemplo, @telefono, @tipo, @salario. En un descriptor de ruta los atributos se nombran como si fueran etiquetas hijo pero anteponiendo @.

Ejemplos:

CONSULTA
`/colegio/departamento[@tipo]` Devuelve los datos de los departamentos que tengan el atributo tipo.
`data(/colegio/departamento[@tipo])` Devuelve los datos de los departamentos que tengan el atributo tipo sin las etiquetas.

`/colegio/departamento/empleado[@salario]` Se obtienen los datos de los empleados que tengan el atributo salario.
`/colegio/departamento[@telefono ="990033"]` Se obtienen los datos del departamento cuyo teléfono es 990033.
`data(/colegio/departamento[@telefono ="990033"])` Se obtienen los datos del departamento cuyo teléfono es 990033 sin las etiquetas de los elementos.
`/colegio/departamento[@tipo = 'B']` Se obtienen los datos de los departamentos de tipo B
`/colegio/departamento[@tipo = 'A']/empleado` Se obtienen los datos de los empleados del departamento de tipo A.
`/colegio/departamento/empleado[@salario > '2100']` Se obtienen los datos de los empleados cuyo salario es superior a 2100
`/colegio/departamento/empleado[@salario > '2100']/nombre/text()` Se obtienen los nombres de los empleados cuyo salario es superior a 2100.

`/colegio/departamento/empleado[@salario > '2100']/concat(nombre, ', ', @salario)` Se obtienen los datos concatenados del nombre del empleado y su salario, de los empleados que tengan un salario superior a 2100.
`/colegio/departamento[@tipo ='A']/count(empleado)` Devuelve el número de empleados del departamento de tipo A.
`/colegio/departamento[@tipo ='A']/concat(nombre, ', ', count(empleado))` Devuelve para cada departamento de tipo A, el nombre del departamento y su número de empleados.
`/colegio/departamento/concat(nombre, ', ', count(empleado))` Devuelve el número de empleados por cada departamento.
`sum(//empleado/@salario)` Devuelve la suma total del salario de todos los empleados.
`/colegio/departamento/concat(nombre, ', Total = ', sum(empleado/@salario))` Obtiene para cada departamento la concatenación de su nombre y el total de los salarios de los empleados del departamento.
`min(//empleado/@salario)` Devuelve el salario mínimo de todos los empleados.

`/colegio/departamento/concat(nombre, ', Mínimo = ', min(empleado/@salario))` Devuelve la concatenación del nombre de cada departamento y el salario mínimo de sus empleados.
`/colegio/departamento/concat(nombre, ', Máximo = ', max(empleado/@salario))` Devuelve la concatenación del nombre de cada departamento y el salario máximo de sus empleados.
`/colegio/departamento/concat(nombre, ', Media = ', avg(empleado/@salario))` Devuelve la concatenación del nombre de cada departamento y el salario medio de sus empleados.
`/colegio/departamento[count(empleado) > 3]` Obtiene los datos de los departamentos con más de 3 empleados.
`/colegio/departamento[count(empleado) > 3]/nombre/text()` Devuelve el nombre de los departamentos con más de 3 empleados.
`/colegio/departamento[@tipo = 'A' and count(empleado) > 2]/nombre/text()` Devuelve el nombre de los departamentos del tipo A con más de 2 empleados.

AXIS XPATH

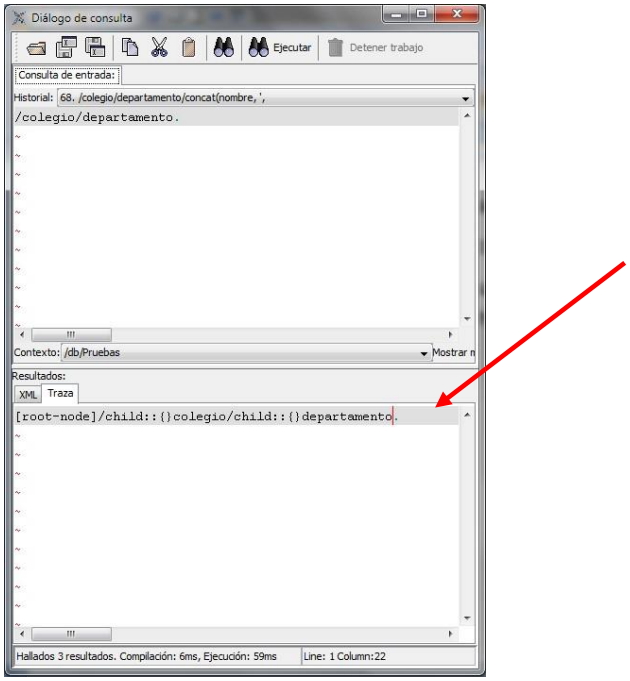
Un AXIS o eje, especifica la dirección que se va a evaluar, es decir, si nos vamos a mover hacia arriba en la jerarquía o hacia abajo, si va a incluir el nodo actual o no, es decir, define un conjunto de nodos relativo al nodo actual. Los nombres de los ejes son los siguientes:

NOMBRE AXIS	RESULTADO
ancestor	Selecciona los antepasados (padres, abuelos, etc.) del nodo actual.
ancestor-or-self	Selecciona los antepasados (padres, abuelos, etc.) del nodo actual y el nodo actual.
attribute	Selecciona los atributos del nodo actual.
child	Selecciona los hijos del nodo actual
descendant	Selecciona los descendientes (hijos, nietos, etc.) del nodo actual.
descendant-or-self	Selecciona los descendientes (hijos, nietos, etc.) del nodo actual y el nodo actual.
following	Selecciona todo el documento después de la etiqueta de cierre del nodo actual.
following-sibling	Selecciona todos los hermanos que siguen al nodo actual
parent	Selecciona el padre del nodo actual
self	Selecciona el nodo actual

La sintaxis para utilizar ejes es la siguiente:

```
NombreDelEje::nombreNodo[expresion]
```

En la ventana **Diálogo de Consulta** del *Cliente Administrador de eXist*, se puede ver la traza de las consultas y en la traza se pueden ver los ejes utilizados.



Ejemplos

CONSULTA

`/colegio/child::*`
`/child::colegio/child::element()`
Las dos consultas devuelven los hijos de colegio, es decir, los nodos de los departamentos.

`/colegio/departamento/descendant::*`
`/child::colegio/child::departamento/descendant::element()`
Ambas consultas devuelven los descendientes del nodo departamento.

`/colegio/departamento/descendant::empleado` Devuelve los nodos empleado descendientes de los nodos departamentos.
`/colegio/descendant::nombre` Devuelve todos los elementos nombre descendientes de colegio, tanto nombres de departamento como de empleados.
`data(/colegio/descendant::nombre)` Devuelve lo mismo que la anterior, pero únicamente el texto del nombre.

```
/colegio/departamento/following-sibling::*
```

Selecciona todos los hermanos de departamento a partir del primero siguiendo el orden del documento.

```
//empleado/following-sibling::node()
```

Selecciona todos los hermanos de los elementos empleado que encuentre en el contexto.

```
//empleado/following-sibling::empleado[@salario>2100]
```

Selecciona todos los hermanos de los elementos empleado que tienen un salario superior a 2100.

```
//empleado[nombre = 'Mª Jesús Ramos']/following-sibling::*
```

Selecciona los nodos hermanos de Mª Jesús Ramos.

```
//empleado[nombre = 'Mª Jesús
    Ramos']/followingsibling::empleado/nombre/text()
```

Selecciona los nombre de los empleados hermanos de Mª Jesús Ramos.

```
//empleado[nombre = 'Mª Jesús Ramos']/following-
sibling::empleado[puesto = 'Profesor']/nombre/text()
```

Selecciona los nombre de los empleados hermanos de Mª Jesús Ramos que son profesores.

```
//empleado/parent::departamento/nombre
```

Selecciona el nombre de los padres de los elementos empleado .

```
//empleado[nombre = 'Mª Jesús Ramos']/parent::departamento/nombre
```

Selecciona el nombre del padre de la empleada 'Mª Jesús Ramos' .

```
/descendant::departamento[1]
```

Selecciona los descendientes del departamento que ocupa la posición 1 en el documento.

```
/child::colegio/child::departamento[count(child::empleado) > 3]
```

Obtiene los departamentos con más de 3 empleados

```
/child::colegio/child::departamento/child::nombre
```

Selecciona las etiquetas con los nombres de los departamentos .

```
/child::colegio/child::departamento/child::nombre/child::text()
```

Obtiene los nombres de los departamentos .

```
/child::colegio/child::departamento[attribute::tipo =
'B'][count(child::empleado) >= 2]/child::nombre/child::text()
```

Devuelve el nombre de los departamentos de tipo ‘B’ y con 2 o más empleados. Es lo mismo que poner :

```
/colegio/departamento[@tipo = 'B' and count(empleado) >=
2]/nombre/text()
```