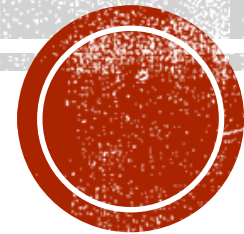


ALMACENAMIENTO DE INFORMACIÓN



ALMACENAMIENTO DE INFORMACIÓN

- Sistemas de almacenamiento de información.
- Inserción y extracción de información en XML.
- Técnicas de búsqueda de información en documentos XML.
- Lenguajes de consulta y manipulación.
- Almacenamiento XML nativo.
- Herramientas de tratamiento y almacenamiento de información en formato XML



ALMACENAMIENTO DE INFORMACIÓN

- El almacenamiento y la posterior lectura de ficheros XML son tareas necesarias para trabajar con documentos XML
- Existen diferentes alternativas para realizar estas funciones:
 - Procesamiento manual mediante el uso de editor de textos
 - Creación de programas para la construcción y lectura de ficheros XML
 - Utilización de herramientas específicas
 - Existen tecnologías y herramientas para habilitar el acceso a los datos de los documentos XML
 - Sistemas de consulta capaces de filtrar la información basándose en reglas lógicas
 - Procesos de transformación de los datos en XML a otros formatos



ALMACENAMIENTO DE INFORMACIÓN

- Creación de documentos XML desde programa
 - Los lenguajes de programación más extendidos (como son Java, Python, C, VisualBasic, ...) disponen de una serie de APIs (Application Programming Interface) que proporcionan mecanismos para analizar, validar, consultar documentos XML y transformar
 - Las herramientas o programas que leen el lenguaje XML y comprueban si el documento es correcto sintácticamente, se denominan **analizadores o “parsers”**
 - Un parser XML es un módulo, biblioteca o programa que se ocupa de transformar un archivo de texto en una **representación interna**
 - En el caso de XML, como el formato siempre es el mismo, no se necesita crear un parser cada vez que se hace un programa, sino que existen un gran número de parsers o analizadores sintácticos disponibles, entre esos analizadores o parsers cabe destacar DOM y SAX
 - **DOM y SAX, son dos herramientas que sirven para analizar el lenguaje XML y definir la estructura de un documento**, aunque existen otras muchas



ALMACENAMIENTO DE INFORMACIÓN

- Creación de documentos XML desde programa
 - **SAX** es un mecanismo simple y eficiente desde el punto de vista del uso de memoria ya que va realizando el procesamiento del documento según se realiza la lectura:
 - El documento se lee **secuencialmente**
 - Según se suceden **eventos** (comienzo de documento, comienzo de elemento, ...) se **van invocando métodos** por el analizador
 - La lectura siempre se realiza hacia adelante, el uso de memoria es muy bajo ya que no se crea en memoria la estructura de árbol (que sí se produce con el analizador DOM)
 - Sólo permite lectura, por tanto no es apropiado para modificar documentos existentes o crear nuevos



ALMACENAMIENTO DE INFORMACIÓN

- Creación de documentos XML desde programa
 - **DOM:**
 - **Almacena en memoria todo el documento XML**
 - Proporciona **mayor versatilidad** ya que permite utilizar lenguajes de búsqueda y selección como XPath
 - Permite crear documentos de cero o modificar documentos existentes
 - Se utiliza para el almacenamiento en memoria tanto de documentos XML como HTML y XHTML
 - Un analizador SAX es una herramienta más veloz y menos potente que un analizador DOM. El uso de SAX requiere una mayor programación, pero es muy útil si lo que interesa es rescatar un fragmento de un documento o buscar sólo un elemento en particular
 - Un analizador DOM es más lento, pero requiere menos desarrollo. Con DOM se obtiene el árbol ya construido



ALMACENAMIENTO DE INFORMACIÓN

- Sistemas de almacenamiento de documentos XML
 - El manejo de los datos contenidos en un documento XML no siempre se realiza en el momento en que dichos datos son creados, por lo que lo más habitual es que sea **necesario guardar la información en un sistema de almacenamiento persistente** para poder recuperarla y procesarla posteriormente, en este sentido se distinguen tecnologías diferentes:
 - Uso de **ficheros de texto**
 - **Bases de datos relacionales** (tradicionales)
 - **Bases de datos nativas XML**
 - La tecnología elegida condiciona el proceso de creación-almacenamiento-recuperación-procesado de los documentos xml, que se podrá implementar mediante varias alternativas, entre las que habrá que elegir. Los parámetros a tener en cuenta para la toma de decisión son:
 - Dificultad de desarrollo y uso
 - Independencia de otras tecnologías
 - Posibilidad de operaciones directas



ALMACENAMIENTO DE INFORMACIÓN

- Sistemas de almacenamiento de documentos XML
 - Características de las soluciones para el manejo de documentos XML

Implementación	Dificultad	Independencia de otras tecnologías	Permite operaciones directas
Ficheros	Baja	Sí	No
Bases de datos convencionales	Media	Opcionalmente	Opcionalmente
Bases de datos nativas XML	Media	No	Sí



ALMACENAMIENTO DE INFORMACIÓN

- Sistemas de almacenamiento de documentos XML
 - Implementación **en base a ficheros**
 - El proceso completo de creación-almacenamiento-recuperación-procesado se puede realizar en base a ficheros
 - Este proceso de tratamiento de información se podría hacer en base a cualquier formato diferente de XML, ya que lo que requiere es lectura de caracteres desde un fichero y cargarlos en memoria para su procesamiento y/o escritura de caracteres a un fichero
 - La ventaja de hacerlo con XML es que hay una **variedad de herramientas que facilitan el trabajo**, por ejemplo los analizadores SAX y DOM
 - Por ejemplo **DOM presenta una colección de métodos que facilitan el acceso al contenido de documentos XML** (elementos, atributos, textos, ...) desde los diferentes lenguajes de programación
 - **SAX permite el acceso al contenido de documentos XML en base a una serie de eventos predefinidos** (startDocument, startElement, characters, endElement, ...) que pueden ser reprogramados



ALMACENAMIENTO DE INFORMACIÓN

- Sistemas de almacenamiento de documentos XML
 - Implementación en base a ficheros
 - El almacenamiento de documentos XML en ficheros es el mecanismo más **independiente** y más **sencillo conceptualmente**
 - Sin embargo es el que presenta mayores **dificultades para el acceso a la información** ya que **cualquier manipulación se debe hacer en base a la información cargada en la memoria** del programa que lo procesa
 - Cualquier lenguaje de programación con capacidad para gestionar ficheros puede trabajar de esta forma. **La dificultad en el procesamiento se palía en parte mediante el uso de herramientas XML**



ALMACENAMIENTO DE INFORMACIÓN

- Sistemas de almacenamiento de documentos XML
 - Almacenamiento de documentos XML en **bases de datos convencionales**
 - Los sistemas de gestión de bases de datos convencionales como son los relacionales (Oracle, MySQL, Microsoft SQL Server, PostgreSQL, ...) están diseñados para almacenar y gestionar datos a través de tablas y sus relaciones
 - Permiten organizar la información de forma eficiente y fiable
 - Son muy importantes para el funcionamiento de multitud de aplicaciones
 - Sin embargo en origen **no están concebidos para para gestionar documentos estructurados XML**, ni el acceso a la información contenida en los mismos
 - No obstante **disponen de mecanismos para almacenar y recuperar documentos de texto en cualquier formato** (e incluso ficheros no de texto utilizando campos de tipo BLOB)
 - **Algunos de los SGBDs** más avanzados **presentan además características específicas para el manejo XML**



ALMACENAMIENTO DE INFORMACIÓN

- Sistemas de almacenamiento de documentos XML
 - Almacenamiento de documentos XML en bases de datos convencionales
 - Por tanto, una solución posible en todos los SGBDs consiste en **almacenar el documento XML en una columna de tipo texto** (en un registro de una tabla de la base de datos) de esta forma se dispone de
 - Una **persistencia** básica **similar a la proporcionada por los ficheros**
 - Las **facilidades** que proporcionan las bases de datos **para el manejo de grandes volúmenes de información**
 - En estos casos se debe implementar un código para almacenar la información (contenido del documento XML) tipo texto en el campo de texto definido para ello
 - Para el procesamiento posterior se debe implementar un código que realice la operación inversa de recuperación del contenido del campo de texto correspondiente al documento XML, previo a este procesamiento



ALMACENAMIENTO DE INFORMACIÓN

- Sistemas de almacenamiento de documentos XML
 - Almacenamiento de documentos XML en bases de datos convencionales
 - En general el almacenamiento en una base de datos convencional **difiere del almacenamiento en fichero en el lugar de almacenamiento de la información y en la programación de las acciones para su almacenamiento y recuperación**, pero **no en las facilidades** que proporcionan **para operar con los datos contenidos** en el documento
 - Algunos gestores de bases de datos como Oracle o MySQL permiten realizar consultas XPath sobre documentos XML almacenados en sus tablas, lo que supone una forma de integrar XML de forma “nativa” en un gestor no creado con esta finalidad



ALMACENAMIENTO DE INFORMACIÓN

- Sistemas de almacenamiento de documentos XML
 - Almacenamiento de documentos XML en **bases de datos nativas XML**
 - Las bases de datos nativas XML están **orientadas al almacenamiento de documentos XML**, mientras que las bases de datos relacionales están orientadas a datos
 - Permiten **acceder directamente a la información** contenida sin necesidad de leer todo el documento **utilizando técnicas de acceso como XPath** (que ofrece una funcionalidad similar a SQL en las bases de datos relacionales)
 - Existen varios sistemas de gestión de bases de datos XML, como son:
 - BaseX, software libre
 - eXist-db, software libre
 - MarkLogic, comercial
 - Qualcomm Qizx, comercial



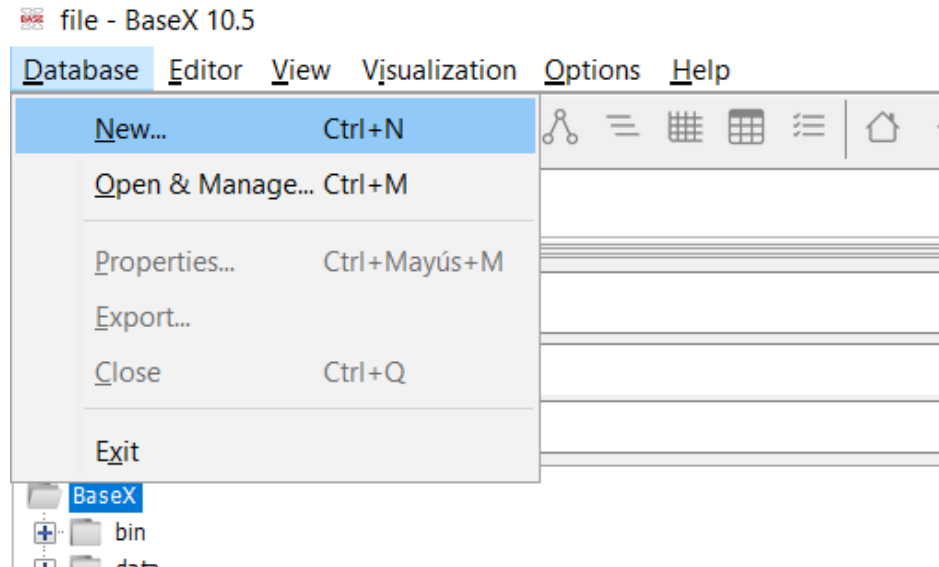
ALMACENAMIENTO DE INFORMACIÓN

- Sistemas de almacenamiento de documentos XML
 - Almacenamiento de documentos XML en bases de datos nativas XML
 - **BaseX** es uno de los sistemas gestores de base de datos nativos XML más populares, permite crear bases de datos con documentos XML, mostrarlos desde diferentes perspectivas, realizar consultas y editar la información contenida en dichos documentos
 - El proceso de instalación de BaseX es muy sencillo:
 - Descarga de la aplicación del sitio web <https://basex.org/>, y acceso a menú Download
 - Instalación en S.O. Windows mediante Windows Instaler aceptando la configuración propuesta por dicho asistente



ALMACENAMIENTO DE INFORMACIÓN

- Sistemas de almacenamiento de documentos XML
 - Almacenamiento de documentos XML en bases de datos nativas XML
 - Creación de una base de datos en BaseX → Acceder a la opción de menú general Database + New



ALMACENAMIENTO DE INFORMACIÓN

- Sistemas de almacenamiento de documentos XML
 - Almacenamiento de documentos XML en bases de datos nativas XML
 - Una vez buscado y seleccionado el fichero XML que se desea procesar, BaseX muestra la interfaz gráfica con la información correspondiente al documento XML seleccionado
 - En el ejemplo está formado por un conjunto de empleados (de la Empresa Estudios MJSV) con información sobre su nombre, puesto y fecha de incorporación
 - **Las vistas mostradas** (ver figura siguiente) **son varias**, de izquierda a derecha y de arriba abajo:
 - Carpeta de trabajo con el listado de documentos XML (A)
 - Documento XML sin procesar (B)
 - Documento XML mostrado como bloques contenedores que, a su vez, pueden ser contenedores (C)
 - Segmento del documento XML correspondiente a la selección realizada en cualquiera de sus vistas (D)
 - Comando ejecutado y resultado (F)
 - Documento XML mostrado como árbol (G)



ALMACENAMIENTO DE INFORMACIÓN

- Sistemas de almacenamiento de documentos XML
 - Almacenamiento de documentos XML en bases de datos nativas XML

The screenshot displays the BaseX 10.5 interface, which is used for managing XML files and databases. The interface is divided into several panes:

- (A) File Explorer:** Shows the file system structure, including folders like 'bin', 'data', 'etc', 'ico', 'lib', 'repo', 'src', 'webapp' and files like 'BaseX.jar', 'CHANGELOG', 'Ejercicio1.xml', 'LICENSE', 'readme.txt', and 'uninst.exe'.
- (B) Editor:** Displays the XML file 'Ejercicio1.xml' with the following content:

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet href="ejercicio_10.xsl" type="text/xsl"?>
<empresa nombre="Estudios MJSV" web="http://www.estudios_mjsv.com/" >
  <empleados>
    <empleado id="ASG">
      <nombre>Antonio Sánchez Gutiérrez</nombre>
      <puesto>Director</puesto>
      <incorporacion año="2015"/>
    </empleado>
    <empleado id="ARD">
      <nombre>Ana Reyes Descalzo</nombre>
      <puesto>Comercial</puesto>
      <incorporacion año="2015"/>
    </empleado>
    <empleado id="JLA">
      <nombre>Juan López Álvarez</nombre>
      <puesto>Comercial</puesto>
      <incorporacion año="2015"/>
    </empleado>
  </empleados>
</empresa>
```
- (C) Visualizer:** Provides a graphical representation of the XML data, showing a tree structure with nodes for 'empresa', 'empleados', and individual employee records.
- (D) Result:** Shows the output of a query, displaying the XML content for the employee with id 'ASG':

```
<empleado id="ASG">
  <nombre>Antonio Sánchez Gutiérrez</nombre>
  <puesto>Director</puesto>
  <incorporacion año="2015"/>
</empleado>
```
- (E) Command/Result:** Displays the command 'CREATE DB Ejercicio1 C:/Program Files (x86)/BaseX/Ejercicio1.xml' and the result 'Database 'Ejercicio1' created in 60.17 ms.'.
- (F) Info:** Shows a detailed tree diagram of the XML structure, highlighting the hierarchy from 'Ejercicio1.xml' down to individual employee data.

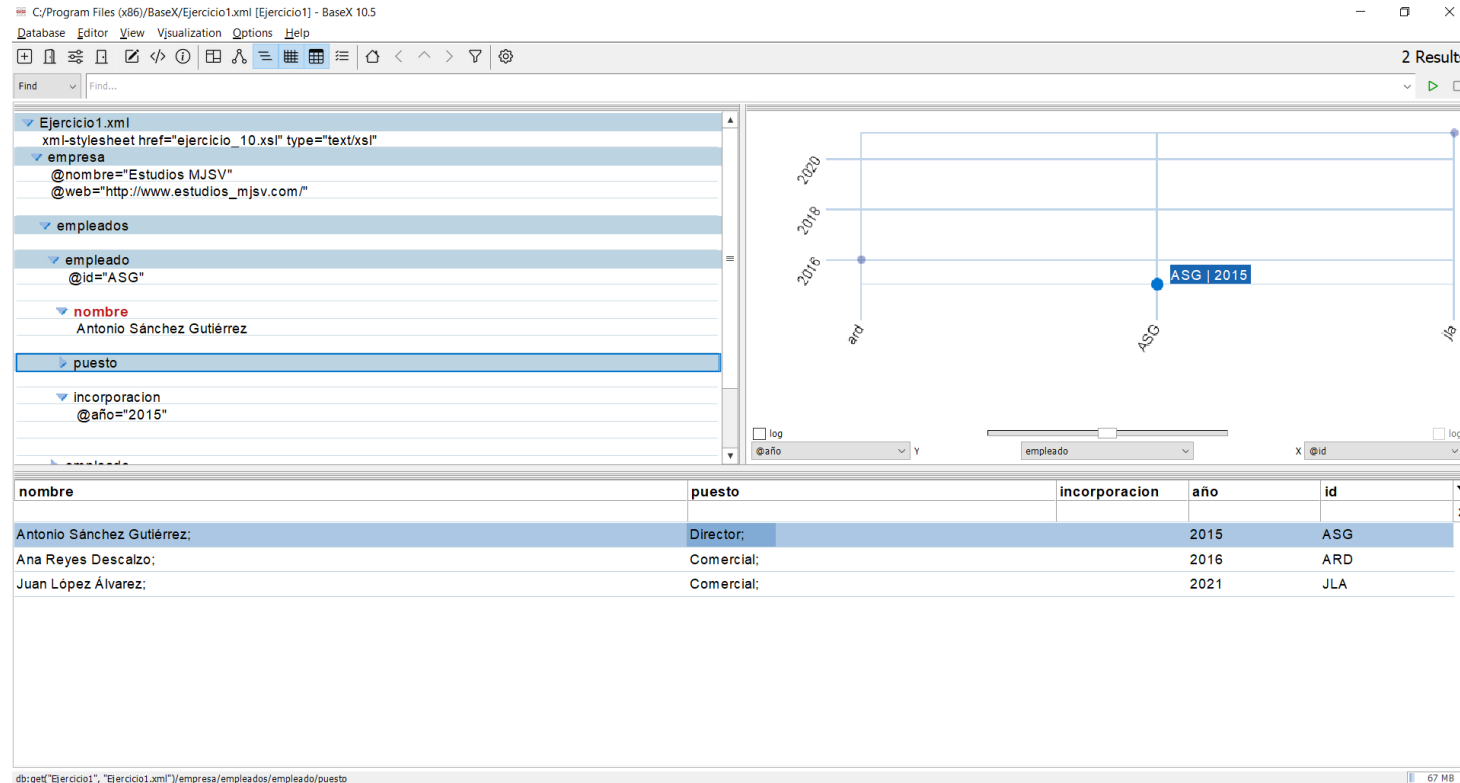
ALMACENAMIENTO DE INFORMACIÓN

- Sistemas de almacenamiento de documentos XML
 - Almacenamiento de documentos XML en bases de datos nativas XML
 - BaseX:
 - Dispone de una **amplia variedad de paneles con diferentes presentaciones, configurable** (ver figura siguiente)
 - Permite la **ejecución de consultas utilizando el lenguaje XQuery** (lenguaje de consulta equivalente a SQL en las bases de datos relacionales)



ALMACENAMIENTO DE INFORMACIÓN

- Sistemas de almacenamiento de documentos XML
 - Almacenamiento de documentos XML en bases de datos nativas XML



ALMACENAMIENTO DE INFORMACIÓN

- Sistemas de almacenamiento de documentos XML
 - Almacenamiento de documentos XML en bases de datos nativas XML
 - BaseX:
 - En la siguiente figura se muestra la ejecución de una consulta XQuery en Base X, en la que se muestran el elemento empleado que tiene el atributo id 'ASG'
 - En el panel superior derecho se incluye la consulta en XQuery:

```
/empresa/empleados/empleado[@id='ASG']
```
 - En el panel inferior izquierdo se muestra el resultado de la consulta:

```
<empleado id="ASG">  
  <nombre>Antonio Sánchez Gutiérrez</nombre>  
  <puesto>Director</puesto>  
  <incorporacion año="2015"/>  
</empleado>
```



ALMACENAMIENTO DE INFORMACIÓN

- Sistemas de almacenamiento de documentos XML
 - Almacenamiento de documentos XML en bases de datos nativas XML

The screenshot displays the BaseX 10.5 application window. The top menu bar includes Database, Editor, View, Visualization, Options, and Help. The main interface is divided into several panes:

- Left Pane:** A file explorer showing the directory structure of the BaseX installation, including folders like bin, data, etc, and files like BaseX.jar, CHANGELOG, Ejercicio1.xml, LICENSE, readme.txt, and uninstall.exe.
- Editor Pane:** Displays the XML document 'Ejercicio1.xml' with the following path: `/empresa/empleados/empleado[@id='ASG']`.
- Result Pane:** Shows the query result for the selected path. The result is an XML element: `<empleado id="ASG"><nombre>Antonio Sánchez Gutiérrez</nombre><puesto>Director</puesto><incorporacion año="2015"/></empleado>`.
- Bottom Right Pane:** Provides detailed information about the query execution, including the optimized query, the original query, and the result statistics.

The bottom status bar indicates the time required for the query: 1.68 ms.

Query Details:

- Optimizing:**
 - rewrite context value: . -> db:get-pre("Ejercicio1", 0)
 - rewrite util:root(nodes): util:root(db:get-pre("Ejercicio1", 0)) -> db:get-pre("Ejercicio1", 0)
 - apply attribute index for "ASG"
- Optimized Query:**
`db:attribute("Ejercicio1", "ASG")/self::attribute(id)/parent::empleado`
- Query:**
`/empresa/empleados/empleado[@id='ASG']`
- Result:**
 - Hit(s): 1 Item
 - Updated: 0 Items
 - Printed: 157 b
 - Read Locking: Ejercicio1
 - Write Locking: (none)



ALMACENAMIENTO DE INFORMACIÓN

- Técnicas de búsqueda de información en documentos XML
 - El lenguaje XML se concibió como un lenguaje de intercambio de información en un formato estandarizado
 - Debido a que tiene una estructura conocida
 - Que cumple unas reglas
 - A partir de esta estructura se han desarrollado una serie de tecnologías relacionadas para automatizar la validación, la transformación y también la búsqueda de información concreta dentro de los documentos
 - **Una de las operaciones más complejas** a realizar sobre un documento XML **es la búsqueda de información**. Dentro del ámbito de la búsqueda existen **dos tecnologías** relacionadas:
 - **XPath**
 - **XQuery**



ALMACENAMIENTO DE INFORMACIÓN

- Técnicas de búsqueda de información en documentos XML
 - **XPath es un lenguaje que permite seleccionar nodos o conjuntos de nodos de un documento XML** y se utiliza como soporte para otras tecnologías como XQuery, XSLT y XSL-FO
 - Es una recomendación del consorcio W3C, la última versión presentada es la 3.1 (en 2017), no obstante la versión 1.0 (de 1999) se sigue utilizando de forma habitual
 - Sus expresiones **no tiene una sintaxis XML**, sino una **notación similar a la que se usa para especificar rutas de ficheros en sistemas de archivos de un computador**
 - **Se basa en la expresión de ruta (path expresión)** que es una cadena constituida a partir de palabras clave, símbolos y operandos
 - Es sensible a mayúsculas. Las palabras clave van en minúsculas



ALMACENAMIENTO DE INFORMACIÓN

- XPath

- Tipos de nodos

- Un documento XML puede representarse como un árbol dirigido, considerando los elementos como nodos y que un elemento es padre de los elementos que contiene
 - **En XPath no sólo los elementos son nodos**, en realidad hay siete tipos de nodos:
 - Raíz
 - Elemento
 - Atributo
 - Texto
 - Comentario
 - Instrucción de procesamiento
 - Espacio de nombres
 - La declaración DOCTYPE no se considera como nodo



ALMACENAMIENTO DE INFORMACIÓN

- XPath

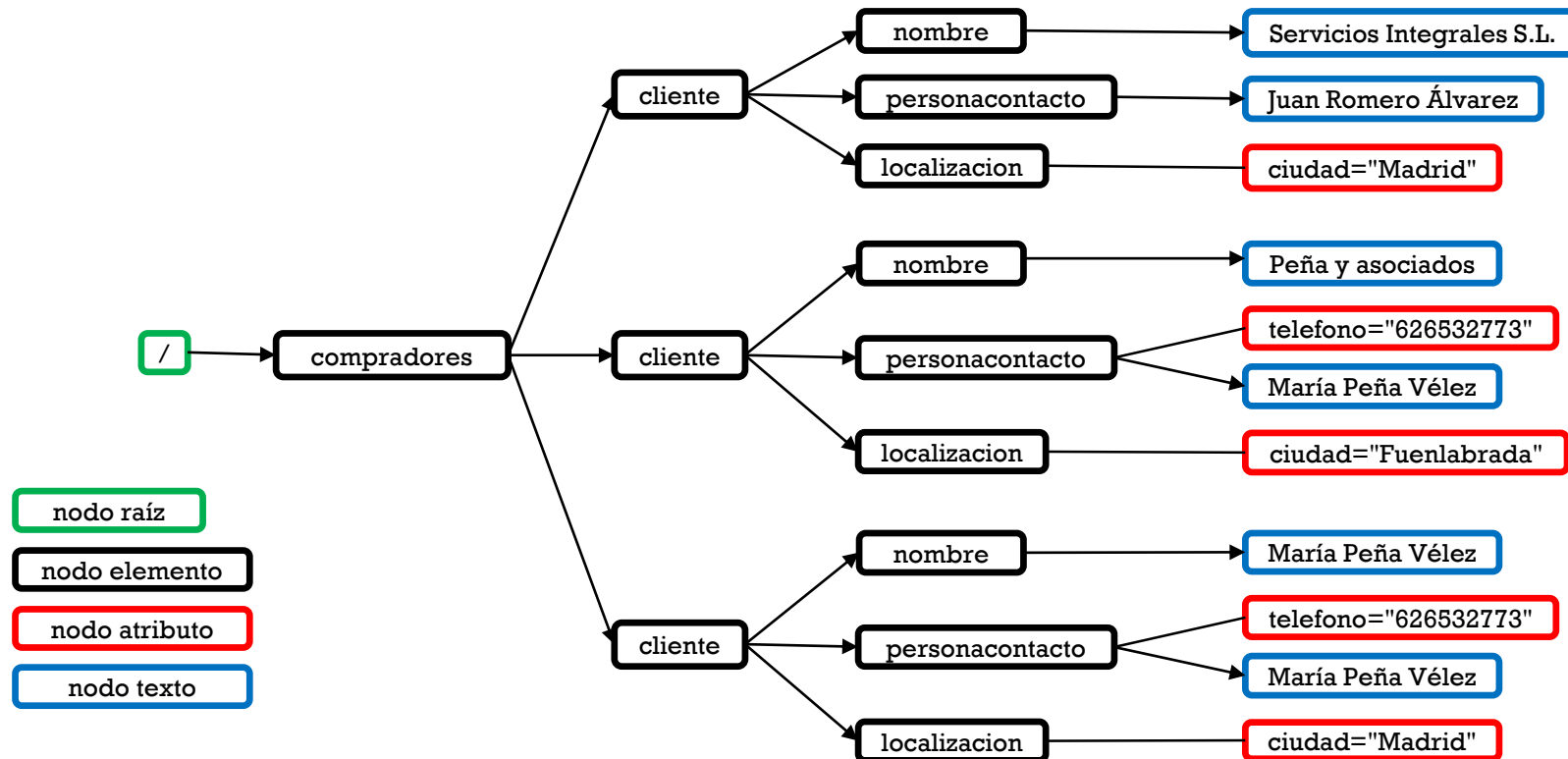
- Por ejemplo, considerando el XML siguiente

```
<?xml version="1.0" encoding="UTF-8"?>
<compradores>
  <cliente>
    <nombre>Servicios Integrales S.L.</nombre>
    <personacontacto>Juan Romero Álvarez</personacontacto>
    <localizacion ciudad="Madrid"/>
  </cliente>
  <cliente>
    <nombre>Peña y asociados</nombre>
    <personacontacto telefono="626532773">María Peña Vélez</personacontacto>
    <localizacion ciudad="Fuenlabrada"/>
  </cliente>
  <cliente>
    <nombre>María Peña Vélez</nombre>
    <personacontacto telefono="626532773">María Peña Vélez</personacontacto>
    <localizacion ciudad="Madrid"/>
  </cliente>
</compradores>
```



ALMACENAMIENTO DE INFORMACIÓN

- XPath
 - Se puede representar mediante el grafo



ALMACENAMIENTO DE INFORMACIÓN

- XPath
 - El **nodo raíz** o nodo documento (root node) no debe confundirse con el elemento raíz.
Es un nodo virtual, padre del elemento raíz
 - Los nodos atributos y de texto no son como los nodos elemento: no pueden tener descendientes
 - 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 elemento y para referirse a los atributos o al texto se utilizan notaciones especiales



ALMACENAMIENTO DE INFORMACIÓN

- XPath
 - 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 un sistema de archivos de un computador
 - Evaluar una expresión XPath consiste en 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
 - **sintaxis completa:** más larga pero con más opciones disponibles



ALMACENAMIENTO DE INFORMACIÓN

- XPath
 - Sintaxis de la expresiones XPath
 - Las expresiones XPath se pueden dividir en pasos de búsqueda. Cada paso de búsqueda se puede a su vez dividir en tres partes:
 - eje: selecciona nodos elemento o atributo basándose en sus nombres
 - predicado: restringe la selección del eje a que los nodos cumplan ciertas condiciones
 - selección de nodos: de los nodos seleccionados por el eje y predicado, selecciona los elementos, el texto que contienen o ambos

eje::selección_nodos[predicado]



ALMACENAMIENTO DE INFORMACIÓN

- XPath

- **Eje**

- **Permite seleccionar un subconjunto de nodos del documento y corresponde a recorridos en el árbol** del documento

- Los nodos elemento se indican mediante el nombre del elemento. Los nodos atributo se indican mediante @ y el nombre del atributo.
 - / → si está al principio de la expresión, indica el nodo raíz, si no, indica "hijo". Debe ir seguido del nombre de un elemento
 - // → indica "descendiente" (hijos, hijos de hijos, etc.)
 - /.. → indica el elemento padre
 - | → permite indicar varios recorridos



ALMACENAMIENTO DE INFORMACIÓN

- XPath

- Eje, ejemplos de uso:

Expresión	Resultado
/compradores/cliente/personacontacto	<personacontacto>Juan Romero Álvarez</personacontacto> <personacontacto telefono="626532773">María Peña Vélez</personacontacto> <personacontacto telefono="626532773">María Peña Vélez</personacontacto>
/personacontacto	No devuelve nada porque <personacontacto> no es hijo del nodo raíz
/compradores/personacontacto	No devuelve nada porque <personacontacto> no es hijo de <compradores>
/compradores/cliente/personacontacto/@telefono	telefono="626532773" telefono="626532773"
/compradores/cliente/@telefono	No devuelve nada porque <cliente> no tiene el atributo telefono
/compradores//personacontacto	<personacontacto>Juan Romero Álvarez</personacontacto> <personacontacto telefono="626532773">María Peña Vélez</personacontacto> <personacontacto telefono="626532773">María Peña Vélez</personacontacto>
//personacontacto	<personacontacto>Juan Romero Álvarez</personacontacto> <personacontacto telefono="626532773">María Peña Vélez</personacontacto> <personacontacto telefono="626532773">María Peña Vélez</personacontacto>
//personacontacto/cliente	No devuelve nada porque <cliente> no es descendiente de <personacontacto>
//@ciudad	ciudad="Madrid" ciudad="Fuenlabrada" ciudad="Madrid"

ALMACENAMIENTO DE INFORMACIÓN

- XPath

- Eje, ejemplos de uso:

Expresión	Resultado
/compradores/cliente/personacontacto/@telefono/..	<personacontacto telefono="626532773">María Peña Vélez</personacontacto> <personacontacto telefono="626532773">María Peña Vélez</personacontacto>
//@telefono/../../	<cliente> <nombre>Peña y asociados</nombre> <personacontacto telefono="626532773">María Peña Vélez</personacontacto> <localizacion ciudad="Fuenlabrada"/> </cliente> <cliente> <nombre>María Peña Vélez</nombre> <personacontacto telefono="626532773">María Peña Vélez</personacontacto> <localizacion ciudad="Madrid"/> </cliente>

- Se obtienen nodos que tienen el atributo telefono



ALMACENAMIENTO DE INFORMACIÓN

- XPath
 - Eje, ejemplos de uso varios recorridos:

Expresión	Resultado
//personacontacto //nombre	<nombre>Servicios Integrales S.L.</nombre> <personacontacto>Juan Romero Álvarez</personacontacto> <nombre>Peña y asociados</nombre> <personacontacto telefono="626532773">María Peña Vélez</personacontacto> <nombre>María Peña Vélez</nombre> <personacontacto telefono="626532773">María Peña Vélez</personacontacto>
//personacontacto //nombre //@ciudad	<nombre>Servicios Integrales S.L.</nombre> <personacontacto>Juan Romero Álvarez</personacontacto> ciudad="Madrid" <nombre>Peña y asociados</nombre> <personacontacto telefono="626532773">María Peña Vélez</personacontacto> ciudad="Fuenlabrada" <nombre>María Peña Vélez</nombre> <personacontacto telefono="626532773">María Peña Vélez</personacontacto> ciudad="Madrid"



ALMACENAMIENTO DE INFORMACIÓN

- XPath
 - Actividad 5.1. Indicar la expresión XPath para obtener todos los nodos <nombre> de los clientes que contiene el fichero XML de trabajo
 - Actividad 5.2. Indicar la expresión XPath para obtener todos los nodos para los que está definido el atributo teléfono
 - Actividad 5.3. Indicar la expresión XPath para obtener todos los nodos <nombre> y <localización>



ALMACENAMIENTO DE INFORMACIÓN

- XPath

- **Predicado**

- Se escribe entre corchetes. En la notación abreviada se incluye a continuación del eje
 - **Permite restringir la selección de los nodos realizada por el eje** a aquellos que cumplan determinadas condiciones
 - [**@atributo**] → selecciona los elementos que tienen el atributo
 - [**número**] → si hay varios resultados selecciona uno de ellos por número de orden, el que ocupa la posición del número indicado; last() selecciona el último de ellos
 - [**condición**] → selecciona los nodos que cumplen la condición



ALMACENAMIENTO DE INFORMACIÓN

- XPath
 - Predicado / Condición (I)
 - Los predicados permiten definir condiciones sobre los valores de los nodos y de los atributos. En las condiciones **se pueden utilizar operadores**:
 - operadores lógicos: and, or, not()
 - operadores de comparación: =, !=, <, >, <=, >=
 - operadores aritméticos: +, -, *, div, idiv, mod
 - Las comparaciones se pueden hacer entre valores de nodos o atributos con cadenas de texto o numéricas. Las cadenas de texto deben escribirse entre comillas simples o dobles. En el caso de las cadenas numéricas, las comillas son optativas



ALMACENAMIENTO DE INFORMACIÓN

- XPath
 - Predicado / Condición (II)
 - La condición puede utilizar el valor de un atributo (utilizando @) o el texto que contiene el elemento
 - Para hacer referencia al propio valor del nodo seleccionado se utiliza el punto (.)
 - En un predicado se pueden incluir **condiciones compuestas mediante el uso de operadores lógicos**
 - Se pueden incluir varios predicados seguidos, cada uno de los cuales restringe los resultados del anterior, lo que equivale al uso de la operación lógica and



ALMACENAMIENTO DE INFORMACIÓN

- XPath

- Predicado, ejemplos de uso:

Expresión	Resultado
//personacontacto[@telefono]	<personacontacto telefono="626532773">María Peña Vélez</personacontacto> <personacontacto telefono="626532773">María Peña Vélez</personacontacto>
//cliente[1]	<cliente> <nombre>Servicios Integrales S.L.</nombre> <personacontacto>Juan Romero Álvarez</personacontacto> <localizacion ciudad="Madrid"/> </cliente>
//cliente[last()]	<cliente> <nombre>María Peña Vélez</nombre> <personacontacto telefono="626532773">María Peña Vélez</personacontacto> <localizacion ciudad="Madrid"/> </cliente>
//cliente[last()-1]	<cliente> <nombre>Peña y asociados</nombre> <personacontacto telefono="626532773">María Peña Vélez</personacontacto> <localizacion ciudad="Fuenlabrada"/> </cliente>



ALMACENAMIENTO DE INFORMACIÓN

- XPath

- Predicado, ejemplos de uso:

Expresión	Resultado
//localizacion[@ciudad="Madrid"]	<localizacion ciudad="Madrid"/> <localizacion ciudad="Madrid"/>
//cliente[personacontacto="María Peña Vélez"]	<cliente> <nombre>Peña y asociados</nombre> <personacontacto telefono="626532773">María Peña Vélez</personacontacto> <localizacion ciudad="Fuenlabrada"/> </cliente> <cliente> <nombre>María Peña Vélez</nombre> <personacontacto telefono="626532773">María Peña Vélez</personacontacto> <localizacion ciudad="Madrid"/> </cliente>
//@ciudad[.="Madrid"]	ciudad="Madrid" ciudad="Madrid"
//personacontacto[.="María Peña Vélez"]	<personacontacto telefono="626532773">María Peña Vélez</personacontacto> <personacontacto telefono="626532773">María Peña Vélez</personacontacto>



ALMACENAMIENTO DE INFORMACIÓN

- XPath

- Predicado, ejemplos de uso:

Expresión	Resultado
//cliente[localizacion/@ciudad="Madrid" and personacontacto="María Peña Vélez"]	<cliente> <nombre>María Peña Vélez</nombre> <personacontacto telefono="626532773">María Peña Vélez</personacontacto> <localizacion ciudad="Madrid"/> </cliente>
//cliente[localizacion/@ciudad="Madrid" or personacontacto="María Peña Vélez"]	<cliente> <nombre>Servicios Integrales S.L.</nombre> <personacontacto>Juan Romero Álvarez</personacontacto> <localizacion ciudad="Madrid"/> </cliente> <cliente> <nombre>Peña y asociados</nombre> <personacontacto telefono="626532773">María Peña Vélez</personacontacto> <localizacion ciudad="Fuenlabrada"/> </cliente> <cliente> <nombre>María Peña Vélez</nombre> <personacontacto telefono="626532773">María Peña Vélez</personacontacto> <localizacion ciudad="Madrid"/> </cliente>
//cliente[localizacion/@ciudad="Madrid"][personacontacto="María Peña Vélez"]	<cliente> <nombre>María Peña Vélez</nombre> <personacontacto telefono="626532773">María Peña Vélez</personacontacto> <localizacion ciudad="Madrid"/> </cliente>



ALMACENAMIENTO DE INFORMACIÓN

- XPath

- Actividad 5.4. Indicar la expresión XPath que permita obtener los nodos que corresponden al segundo elemento nombre de otro elemento
- Actividad 5.5. Indicar la expresión XPath que permita obtener los nodos que corresponden al último elemento nombre de otro elemento
- Actividad 5.6. Indicar la expresión XPath que permita obtener los elementos <nombre> cuyo valor es "María Peña Vélez"
- Actividad 5.7. Indicar la expresión XPath que permita obtener los elementos <cliente> con localización en ciudad de Madrid y que tienen persona de contacto con teléfono



ALMACENAMIENTO DE INFORMACIÓN

- XPath

- **Selección de nodos (node test)**

- Se escribe a continuación del eje y el predicado (en la sintaxis abreviada, no así en la sintaxis completa que va entre el eje y el predicado). El eje y el predicado seleccionan unos nodos y la selección de nodos indica la parte de esos nodos que se considera
 - /node() → selecciona todos los hijos (elementos o texto) del nodo
 - //node() → selecciona todos los descendientes (elementos o texto) del nodo
 - /text() → selecciona únicamente el texto contenido en el nodo
 - //text() → selecciona únicamente el texto contenido en el nodo y todos sus descendientes
 - /* → selecciona todos los hijos (sólo elementos) del nodo
 - //* → selecciona todos los descendientes (sólo elementos) del nodo
 - /@* → selecciona todos los atributos del nodo
 - //* → selecciona todos los atributos de los descendientes del nodo



ALMACENAMIENTO DE INFORMACIÓN

- XPath

- Selección de nodos, ejemplos de uso:

Expresión	Resultado
 //cliente/node()	<nombre>Servicios Integrales S.L.</nombre> <personacontacto>Juan Romero Álvarez</personacontacto> <localizacion ciudad="Madrid"/> <nombre>Peña y asociados</nombre> <personacontacto telefono="626532773">María Peña Vélez</personacontacto> <localizacion ciudad="Fuenlabrada"/> <nombre>María Peña Vélez</nombre> <personacontacto telefono="626532773">María Peña Vélez</personacontacto> <localizacion ciudad="Madrid"/>
//personacontacto/node()	Juan Romero Álvarez María Peña Vélez María Peña Vélez
 //cliente//node()	<nombre>Servicios Integrales S.L.</nombre> Servicios Integrales S.L. <personacontacto>Juan Romero Álvarez</personacontacto> Juan Romero Álvarez <localizacion ciudad="Madrid"/> <nombre>Peña y asociados</nombre> Peña y asociados <personacontacto telefono="626532773">María Peña Vélez</personacontacto> María Peña Vélez <localizacion ciudad="Fuenlabrada"/> <nombre>María Peña Vélez</nombre> María Peña Vélez <personacontacto telefono="626532773">María Peña Vélez</personacontacto> María Peña Vélez <localizacion ciudad="Madrid"/>



ALMACENAMIENTO DE INFORMACIÓN

- XPath

- Selección de nodos, ejemplos de uso:

Expresión	Resultado
//nombre/text()	Servicios Integrales S.L. Peña y asociados María Peña Vélez
//cliente/text()	No devuelve nada porque los elementos <cliente> no contienen texto
//cliente//text()	Servicios Integrales S.L. Juan Romero Álvarez Peña y asociados María Peña Vélez María Peña Vélez María Peña Vélez



ALMACENAMIENTO DE INFORMACIÓN

- XPath

- Selección de nodos, ejemplos de uso:

Expresión	Resultado
//cliente/*	<nombre>Servicios Integrales S.L.</nombre> <personacontacto>Juan Romero Álvarez</personacontacto> <localizacion ciudad="Madrid"/> <nombre>Peña y asociados</nombre> <personacontacto telefono="626532773">María Peña Vélez</personacontacto> <localizacion ciudad="Fuenlabrada"/> <nombre>María Peña Vélez</nombre> <personacontacto telefono="626532773">María Peña Vélez</personacontacto> <localizacion ciudad="Madrid"/>
//personacontacto/*	No devuelve nada porque los elementos <personacontacto> no tienen hijos
//@*	ciudad="Madrid" telefono="626532773" ciudad="Fuenlabrada" telefono="626532773" ciudad="Madrid"
//cliente/@*	No devuelve nada porque los elementos <cliente> no tienen atributos
//personacontacto/@*	telefono="626532773" telefono="626532773"

ALMACENAMIENTO DE INFORMACIÓN

- XPath

- Actividad 5.8. Indicar la expresión XPath que permita obtener los nodos hijos del primer elemento cliente
- Actividad 5.9. Indicar la expresión XPath que permita obtener los nodos hijos de los elementos localizacion
- Actividad 5.10. Indicar la expresión XPath que permita obtener los elementos cuyo valor es "María Peña Vélez"
- Actividad 5.11. Indicar la expresión XPath que permita obtener el valor de todos los atributos del elemento localización
- Actividad 5.12. Indicar la expresión XPath que permita obtener el valor de todos los elementos y atributos del documento



ALMACENAMIENTO DE INFORMACIÓN

- XPath

- **Pasos de búsqueda consecutivos**

- Una expresión XPath puede contener varios pasos de búsqueda consecutivos. Cada uno incluye su eje (y si es preciso su predicado) y el último paso de búsqueda se incluye una selección de nodos si es necesario
 - Cada paso de búsqueda considera los nodos seleccionados por el paso de búsqueda anterior
 - Un determinado resultado se puede obtener mediante un sólo paso de búsqueda o mediante varios pasos



ALMACENAMIENTO DE INFORMACIÓN

- XPath
 - Pasos de búsqueda consecutivos, ejemplos de uso
 - La siguiente expresión permite obtener la persona de contacto del cliente “Servicios Integrales S.L.”

Expresión	Resultado
//nombre[.="Servicios Integrales S.L."]/../personacontacto	<personacontacto>Juan Romero Álvarez</personacontacto>
//cliente[nombre="Servicios Integrales S.L."]/personacontacto	<personacontacto>Juan Romero Álvarez</personacontacto>

- Comentarios en XPath:
 - Un comentario es una serie delimitada por los símbolos (: y :)
- (: Esto es un comentario en XPath :)



ALMACENAMIENTO DE INFORMACIÓN

- XPath
 - Actividad 5.13. Indicar la expresión XPath que permita obtener el nombre (el valor del elemento) de los clientes cuya localización es Madrid
 - Actividad 5.14. Indicar la expresión XPath que permita obtener el nombre (el valor del elemento) de los clientes cuyo teléfono de contacto es 626532773



ALMACENAMIENTO DE INFORMACIÓN

- XPath

- **Expresiones anidadas**

- Las expresiones XPath pueden anidarse, lo que permite definir expresiones más complicadas
 - Si sobre el documento XML de compradores que se está considerando para los ejemplos se ejecuta la consulta:

```
//cliente[nombre="Servicios Integrales S.L."]/localizacion/@ciudad
```

- Y posteriormente con el resultado (ciudad="Madrid") de la consulta anterior se ejecuta una nueva consulta como la siguiente:

```
//cliente[localizacion/@ciudad="Madrid"]/nombre
```

- Se obtiene el resultado siguiente que muestra el elemento nombre de todos los clientes que tienen la misma ciudad de localización que el cliente “Servicios Integrales S.L.”:

```
<nombre>Servicios Integrales S.L.</nombre>
```

```
<nombre>María Peña Vélez</nombre>
```



ALMACENAMIENTO DE INFORMACIÓN

- XPath

- Expresiones anidadas

- Las dos expresiones anteriores se pueden unir en una única expresión, cambiando en la segunda expresión el valor Madrid por la primera expresión:

```
//cliente[localizacion/@ciudad=//cliente[nombre="Servicios Integrales S.L."]/localizacion/@ciudad]/nombre
```

- Con la que, igual que con la expresión anterior, se obtiene el resultado que muestra el elemento nombre de todos los clientes que tienen la misma ciudad de localización que el cliente “Servicios Integrales S.L.”:

```
<nombre>Servicios Integrales S.L.</nombre>
```

```
<nombre>María Peña Vélez</nombre>
```



ALMACENAMIENTO DE INFORMACIÓN

- XPath
 - Actividad 5.15. Indicar la expresión XPath que permita obtener el elemento nombre de todos los clientes que tienen la misma persona de contacto que el cliente "Peña y asociados"
 - Actividad 5.16. Indicar la expresión XPath que permita obtener todos los elementos cuyo valor empiece por "Ma"



ALMACENAMIENTO DE INFORMACIÓN

- XQuery
 - Es un **lenguaje de consulta para acceder a la información contenida en un documento XML**
 - Dispone de capacidades de los lenguajes de programación. **Es un lenguaje de programación funcional que permite la ejecución de instrucciones**
 - Es una recomendación del W3C
- Comparte con XPath el modelo de datos, las funciones y los operadores. **Es un superconjunto de XPath: toda expresión de XPath es una expresión de XQuery**
- Permite extraer información de un documento XML o transformarlo en HTML



ALMACENAMIENTO DE INFORMACIÓN

- XQuery
 - Algunas de las características más destacadas:
 - Distingue mayúsculas de minúsculas
 - Las cadenas de caracteres están delimitadas por comillas simples o dobles
 - Las variables comienzan por el símbolo \$
 - **Las expresiones de XQuery utilizan XPath para acceder a los contenidos de los documentos XML**



ALMACENAMIENTO DE INFORMACIÓN

- XQuery
 - Las instrucciones XQuery se basan en el uso de cláusulas
 - Las **cláusulas FLWOR** (For, Let, Where, Order By, Return) le permiten hacer una gran variedad de consultas sobre documentos XML:
 - for, para recorrer conjuntos de datos, asocia una variable por cada elemento devuelto por la consulta que tiene asociada
 - let, para asignar valores a variables
 - where, para incorporar condiciones a las cláusulas for
 - order by, para ordenar los resultados
 - return, para especificar los resultados que se han de devolver



ALMACENAMIENTO DE INFORMACIÓN

- XQuery
 - La **sintaxis** de la cláusula **FLWOR** es la siguiente:
 - Una o más cláusulas for o let (obligatoria)
 - Una cláusula where (opcional)
 - Una cláusula order by (opcional)
 - Una cláusula return (opcional)
 - **Estas cláusulas se ejecutan sobre un documento previamente referenciado mediante la función doc("nombre_doc.xml")**, que devuelve un documento sobre el que se puede aplicar una ruta XPath para acceder a un conjunto de elementos
 - Se utiliza BaseX como herramienta para la ejecución de consultas XQuery junto con el documento xml



ALMACENAMIENTO DE INFORMACIÓN

- XQuery
 - La cláusula for constituye un bucle de procesamiento que se ejecuta sobre cada uno de los elementos que devuelve la expresión XPath que se incluye en la misma al aplicarse sobre el documento referenciado en base a la siguiente sintaxis:

```
for $variable in doc("documento_referenciado.xml") [expresión_Xpath]
```



ALMACENAMIENTO DE INFORMACIÓN

- XQuery

- Un ejemplo básico:

```
for $a in doc("compradores.xml")  
return $a
```

- Devuelve todo el documento XML

```
<compradores>  
  <cliente>  
    <nombre>Servicios Integrales S.L.</nombre>  
    <personacontacto>Juan Romero Álvarez</personacontacto>  
    <localizacion ciudad="Madrid"/>  
  </cliente>  
  <cliente>  
    <nombre>Peña y asociados</nombre>  
    <personacontacto telefono="626532773">María Peña Vélez</personacontacto>  
    <localizacion ciudad="Fuenlabrada"/>  
  </cliente>  
  <cliente>  
    <nombre>María Peña Vélez</nombre>  
    <personacontacto telefono="626532773">María Peña Vélez</personacontacto>  
    <localizacion ciudad="Madrid"/>  
  </cliente>  
</compradores>
```



ALMACENAMIENTO DE INFORMACIÓN

- XQuery

- La siguiente consulta:

```
for $a in doc("compradores.xml") //cliente
return $a
```

- Devuelve únicamente los nodos cliente

```
<cliente>
  <nombre>Servicios Integrales S.L.</nombre>
  <personacontacto>Juan Romero Álvarez</personacontacto>
  <localizacion ciudad="Madrid"/>
</cliente>
<cliente>
  <nombre>Peña y asociados</nombre>
  <personacontacto telefono="626532773">María Peña Vélez</personacontacto>
  <localizacion ciudad="Fuenlabrada"/>
</cliente>
<cliente>
  <nombre>María Peña Vélez</nombre>
  <personacontacto telefono="626532773">María Peña Vélez</personacontacto>
  <localizacion ciudad="Madrid"/>
</cliente>
```



ALMACENAMIENTO DE INFORMACIÓN

- XQuery

- Una consulta más elaborada:

```
for $a in doc("compradores.xml") //cliente
where $a/personacontacto/@telefono!=" "
return $a/personacontacto
```

- Devuelve los nodos personacontacto que tiene el atributo @telefono informado

```
<personacontacto telefono="626532773">María Peña Vélez</personacontacto>
<personacontacto telefono="626532773">María Peña Vélez</personacontacto>
```

- Si se desea obtener únicamente el nombre de la persona de contacto (el contenido del elemento personacontacto) de los nodos personacontacto que tienen informado el atributo telefono se debe usar la función text():

```
for $a in doc("compradores.xml") //cliente
where $a/personacontacto/@telefono!=" "
return $a/personacontacto/text()
```

- Cuyo resultado es:

```
María Peña Vélez
María Peña Vélez
```



ALMACENAMIENTO DE INFORMACIÓN

- XQuery

- La siguiente consulta devuelve el nombre de la ciudad, ordenado de forma descendente, de los nodos localización hijos de nodos cliente cuya personacontacto tiene informado el atributo telefono:

```
for $a in doc("compradores.xml") //cliente
where $a/personacontacto/@telefono!=" "
let $ciudad := $a/localizacion/@ciudad
order by $a/localizacion/@ciudad descending
return $ciudad
```

- Cuyo resultado es:

```
ciudad="Madrid"
ciudad="Fuenlabrada
```



ALMACENAMIENTO DE INFORMACIÓN

- XQuery

- La siguiente consulta permite **obtener dos datos diferentes, para ello se deben generar dos elementos distintos**. En el ejemplo se devuelve el nombre de la persona de contacto y su teléfono:

```
for $a in doc("compradores.xml") //cliente
where $a/personacontacto/@telefono!=" "
return <resultado>
  <PersonaContacto>{$a/personacontacto/text()}</PersonaContacto>
  <Telefono>{data($a/personacontacto/@telefono)}</Telefono> </resultado>
```

- Cuyo resultado es:

```
<resultado><PersonaContacto>María Peña
Vélez</PersonaContacto><Telefono>626532773</Telefono></resultado>
<resultado><PersonaContacto>María Peña
Vélez</PersonaContacto><Telefono>626532773</Telefono></resultado>
```



ALMACENAMIENTO DE INFORMACIÓN

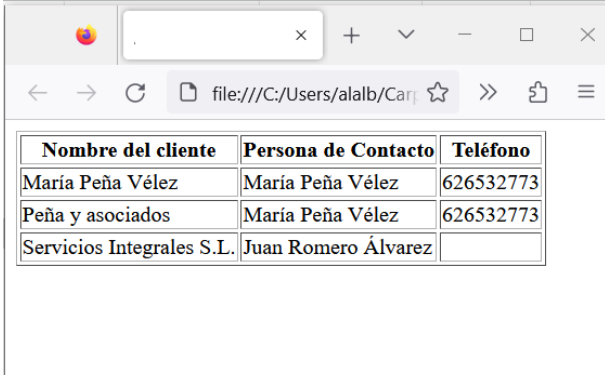
- XQuery

- De forma similar se puede generar el código HTML que permite mostrar una tabla con los nombres de los clientes, la persona de contacto y el número de teléfono:

```
<table border="1">
<tr> <th>Nombre del cliente</th> <th>Persona de Contacto</th>
<th>Teléfono</th></tr>
{
  for $a in doc("compradores.xml") //cliente
  order by $a/nombre
  return <tr><td>{$a/nombre/text()}</td>
<td>{$a/personacontacto/text()}</td>
<td>{data($a/personacontacto/@telefono)}</td></tr>
}
</table>
```

- Cuyo resultado es:

```
<table border="1"><tr><th>Nombre del cliente</th><th>Persona de
Contacto</th><th>Teléfono</th></tr><tr><td>María Peña
Vélez</td><td>María Peña
Vélez</td><td>626532773</td></tr><tr><td>Peña y
asociados</td><td>María Peña
Vélez</td><td>626532773</td></tr><tr><td>Servicios Integrales
S.L.</td><td>Juan Romero Álvarez</td><td></td></tr></table>
```



A screenshot of a web browser window showing an HTML table. The browser's address bar displays the file path: file:///C:/Users/alalb/Car... The table has three columns: 'Nombre del cliente', 'Persona de Contacto', and 'Teléfono'. It contains three data rows.

Nombre del cliente	Persona de Contacto	Teléfono
María Peña Vélez	María Peña Vélez	626532773
Peña y asociados	María Peña Vélez	626532773
Servicios Integrales S.L.	Juan Romero Álvarez	

ALMACENAMIENTO DE INFORMACIÓN

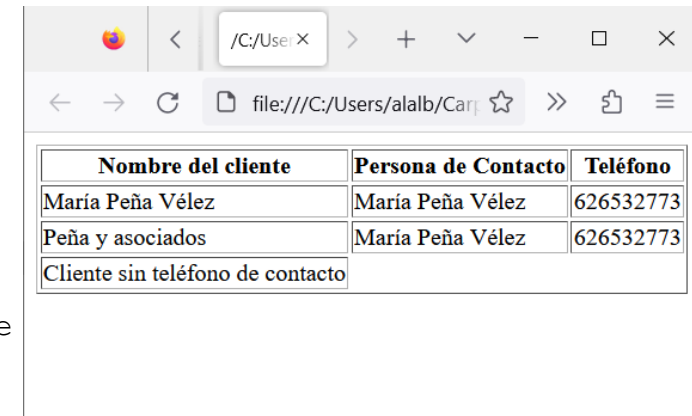
- XQuery

- También permite el uso de sentencias condicionales if-then-else
- En el ejemplo anterior se puede limitar los clientes a incluir en la tabla, de forma que muestre sólo los que tienen informado el atributo telefono

```
<table border="1">
<tr> <th>Nombre del cliente</th> <th>Persona de Contacto</th>
<th>Teléfono</th>
</tr>
{
  for $a in doc("compradores.xml") //cliente
  order by $a/nombre
  return if ($a/personacontacto/@telefono != "") then
<tr><td>{$a/nombre/text()}</td>
<td>{$a/personacontacto/text()}</td>
<td>{data($a/personacontacto/@telefono)}</td></tr> else
<tr><td>Cliente sin teléfono de contacto</td></tr>
}
</table>
```

- Cuyo resultado es:

```
<table border="1"><tr><th>Nombre del cliente</th><th>Persona de
Contacto</th><th>Teléfono</th></tr><tr><td>María Peña
Vélez</td><td>María Peña
Vélez</td><td>626532773</td></tr><tr><td>Peña y
asociados</td><td>María Peña
Vélez</td><td>626532773</td></tr><tr><td>Cliente sin teléfono de
contacto</td></tr></table>
```



The screenshot shows a web browser window with the address bar displaying 'file:///C:/Users/alalb/Car...'. The browser content area displays a table with three columns: 'Nombre del cliente', 'Persona de Contacto', and 'Teléfono'. The table contains three rows of data, with the last row being a summary row for clients without a phone number.

Nombre del cliente	Persona de Contacto	Teléfono
María Peña Vélez	María Peña Vélez	626532773
Peña y asociados	María Peña Vélez	626532773
Cliente sin teléfono de contacto		