

MP0486
Acceso a datos
UF4. Bases de datos XML

4.3. Bases de datos XML vs bases de datos relacionales

Índice

☰	Objetivos	3
☰	Qué son las bases de datos XML	4
☰	Document Type Definition (DTD)	6
☰	XML Schema Definition (XSD)	12
☰	Tecnologías asociadas a las bases de datos XML	19
☰	Qué son las bases de datos documentales	21
☰	Base de datos XML vs Base de datos relacional	22
☰	Resumen	25

Objetivos

En esta lección perseguimos los siguientes objetivos:

1

Comprender qué son las bases de datos XML y su relación con las bases de datos documentales.

2

Comparar la estructura de una base de datos XML y una base de datos relacional.

3

Definir la estructura de una base de datos XML y aplicar restricciones mediante dos técnicas distintas: DTD (*Document Type Definition*) y XSD (*XML Schema Definition*)



¡Ánimo y adelante!

Qué son las bases de datos XML

Una **base de datos XML** está constituida por uno o varios documentos XML, que siguen una estructura lógica similar a la estructura de filas y columnas de una base de datos tradicional, pero con varios niveles.

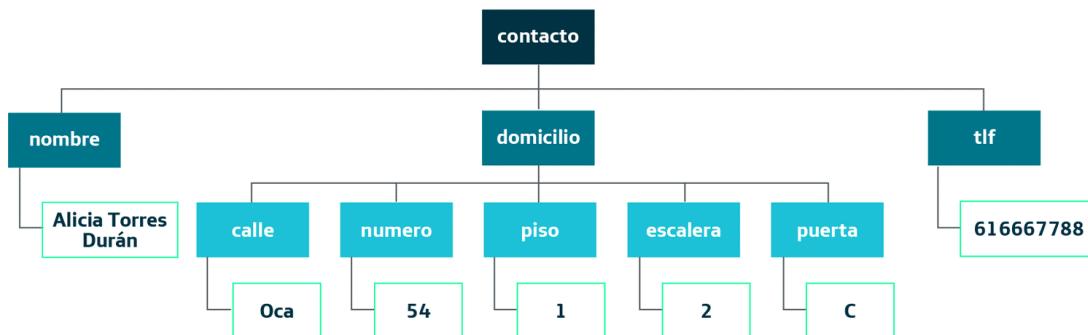
Esta estructura debe ser validada utilizando las tecnologías disponibles.

Cada archivo XML estará constituido por una etiqueta raíz, cuyo nombre debería identificar el tipo de objetos o entidades que se almacenan en él. El siguiente ejemplo refleja una agenda compuesta por objetos *contacto*. Podemos considerar que cada *contacto* es un registro.

```
<agenda>
    <contacto>
        <nombre> Alicia Torres Durán</nombre>
        <domicilio>
            <calle>Oca</calle>
            <numero>54</numero>
            <piso>1</piso>
            <escalera>2</escalera>
            <puerta>C</puerta>
        </domicilio>
        <tlf> 616667788</tlf>
    </contacto>
    .....
</agenda>
```

Ejemplo de una agenda compuesta por objetos *contacto*.

Cada registro (*contacto* en nuestro ejemplo) sigue una estructura jerárquica en varios niveles que puede representarse como en la siguiente imagen:



Estructura jerárquica en varios niveles.

Además, para que un archivo XML pueda ser considerado base de datos, o parte de una base de datos, debe cumplir unos criterios de validación. Para nuestro ejemplo, estos criterios de validación exigirían que cada uno de los contactos tenga la misma estructura, y que datos como el número, piso y escalera sean valores numéricos.

En los siguientes apartados aprenderás a **establecer criterios de validación para un documento XML**, utilizando dos técnicas distintas:

1

Document Type Definition (DTD).

2

XML Schema Definition (XSD).

Document Type Definition (DTD)

Un **DTD (Document Type Definition)** es un archivo que define la estructura de un documento XML, es decir, el nombre de los elementos que contiene, sus atributos, el orden en el que tiene que aparecer cada elemento, si se repiten, y si pueden o no tener elementos hijos.

A continuación te mostraremos cómo sería el archivo DTD para validar la estructura del documento *cruceros.xml* con el que trabajamos anteriormente.

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT cruceros (crucero*)>
<!ELEMENT crucero (destino, detalles, escalas)>

<!ELEMENT destino (#PCDATA)>

<!ELEMENT detalles (cia, dias, fechaSalida)>
<!ELEMENT cia (#PCDATA)>
<!ELEMENT dias (#PCDATA)>
<!ELEMENT fechaSalida (#PCDATA)>

<!ELEMENT escalas (escala*)>
<!ELEMENT escala (parada, llegada, salida)>
<!ELEMENT parada (#PCDATA)>
<!ELEMENT llegada (#PCDATA)>
<!ELEMENT salida (#PCDATA)>

<!ATTLIST crucero codigo CDATA #REQUIRED>
<!ATTLIST escala dia CDATA #REQUIRED>
```

Vamos a analizarlo por partes:

1

```
<!ELEMENT cruceros (crucero*)>  
<!ELEMENT crucero (destino, detalles, escalas)>
```

Aquí, estamos indicando que el elemento *cruceros* estará compuesto por un conjunto de elementos *crucero*, cada uno de los cuales estará compuesto a su vez por los elementos internos *destino*, *detalles* y *escalas*.

2

```
<!ELEMENT destino (#PCDATA)>
```

De este modo indicamos que el elemento o etiqueta *destino* estará compuesto por un *dato elemental*, es decir, que no se descompone en más etiquetas hijas o elementos.

3

```
<!ELEMENT detalles (cia, dias, fechaSalida)>  
<!ELEMENT cia (#PCDATA)>  
<!ELEMENT dias (#PCDATA)>  
<!ELEMENT fechaSalida (#PCDATA)>
```

En este caso, indicamos que el elemento *detalles* estará compuesto por los elementos hijos *cia*, *dias* y *fechaSalida*. Por otro lado, cada uno de estos tres elementos hijo son datos elementales que no se descomponen en más niveles.

4

```
<!ELEMENT escalas (escala*)>  
<!ELEMENT escala (parada, llegada, salida)>  
<!ELEMENT parada (#PCDATA)>  
<!ELEMENT llegada (#PCDATA)>  
<!ELEMENT salida (#PCDATA)>
```

Aquí indicamos que el elemento *escalas* estará compuesto por un conjunto de elementos hijos *escala*. Cada uno de los elementos *escala* estará, a su vez, compuesto por los elementos hijo *parada*, *llegada* y *salida* de tipo elemental.

5

```
<!ATTLIST crucero codigo CDATA #REQUIRED>  
<!ATTLIST escala dia CDATA #REQUIRED>
```

Por último, estamos definiendo los atributos *codigo* y *dia* para las etiquetas *crucero* y *escala* como requeridos.

Sigue estos pasos para crear el archivo DTD desde Eclipse:

1

Primer paso:

Abre el proyecto Eclipse, llamado *ProyectoXML*, que creaste en la primera lección de esta unidad didáctica. Recuerda que en ese proyecto tienes el archivo *cruceros.xml* que enlazarás con el archivo DTD.

2

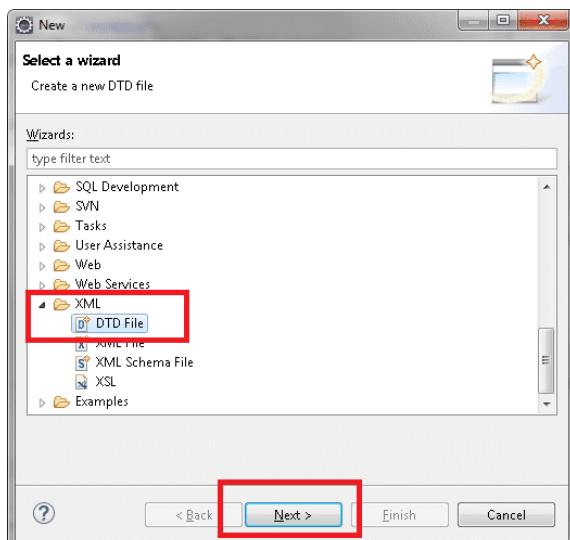
Segundo paso:

Haz clic derecho sobre el nombre del proyecto y selecciona *New / Other* en el menú contextual.

3

Tercer paso:

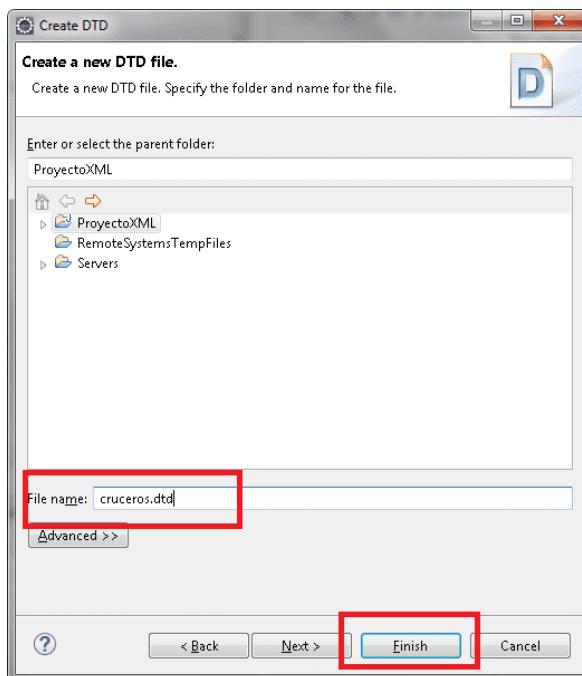
En el cuadro de diálogo *New*, abre la carpeta *XML* y selecciona la opción *DTD file*. Tendrás que utilizar la barra de desplazamiento vertical. Luego pulsa el botón *Next >*.



4

Cuarto paso:

Escribe *cruceros.dtd* como nombre del nuevo fichero y pulsa el botón *Finish*.



5

Quinto paso:

Completa el código del archivo DTD y guarda los cambios. Puedes copiar el código de esta misma lección que aparece más arriba.

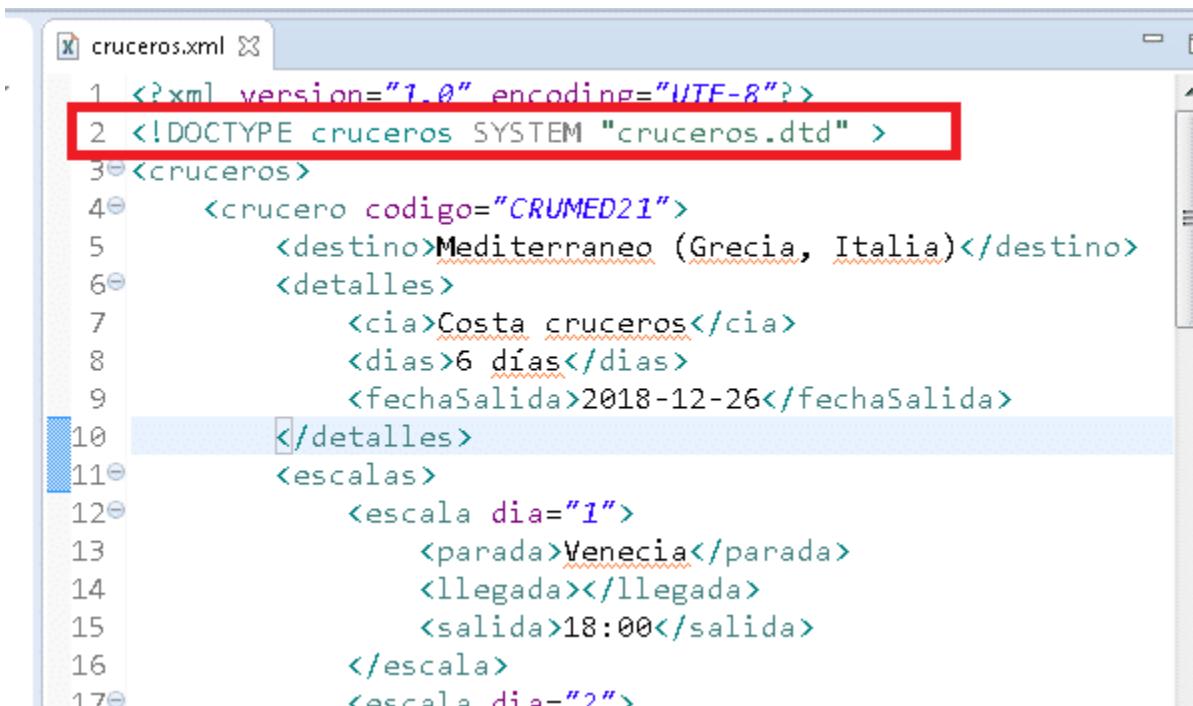
A screenshot of a code editor window. The title bar shows two tabs: 'cruceros.xml' and 'cruceros.dtd'. The 'cruceros.dtd' tab is active, showing the following XML code:

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT cruceros (crucero*)>
<!ELEMENT crucero (destino, detalles, escalas)>
<!ELEMENT destino (#PCDATA)>
<!ELEMENT detalles (cia, dias, fechaSalida)>
<!ELEMENT cia (#PCDATA)>
<!ELEMENT dias (#PCDATA)>
<!ELEMENT fechaSalida (#PCDATA)>
<!ELEMENT escalas (escala*)>
<!ELEMENT escala (parada, llegada, salida)>
<!ELEMENT parada (#PCDATA)>
<!ELEMENT llegada (#PCDATA)>
<!ELEMENT salida (#PCDATA)>
<!ATTLIST crucero codigo CDATA #REQUIRED>
<!ATTLIST escala dia CDATA #REQUIRED>
```

The XML file 'cruceros.xml' is partially visible at the top of the editor.

Sigue los siguientes pasos para enlazar el archivo DTD con el documento XML:

Abre el archivo *cruceros.xml* y añade la siguiente línea justo después de la cabecera del documento XML: <!DOCTYPE cruceros SYSTEM "cruceros.dtd" >



```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE cruceros SYSTEM "cruceros.dtd" >
3<cruceros>
4    <crucero codigo="CRUMED21">
5        <destino>Mediterraneo (Grecia, Italia)</destino>
6        <detalles>
7            <cia>Costa cruceros</cia>
8            <dias>6 días</dias>
9            <fechaSalida>2018-12-26</fechaSalida>
10       </detalles>
11       <escalas>
12           <escala dia="1">
13               <parada>Venecia</parada>
14               <llegada></llegada>
15               <salida>18:00</salida>
16           </escala>
17           <escala dia="2">
```

Con la nueva línea estamos aplicando las restricciones especificadas en el archivo *cruceros.dtd* a la etiqueta raíz *cruceros*.

Ahora, puedes comprobar que las restricciones se están aplicando, probando a editar el código del documento *cruceros.xml*. Las siguientes acciones provocarían error en el código:

- Eliminar el atributo *codigo* en alguno de los cruceros, ya que es un atributo requerido.
- Eliminar el atributo *dia* en alguna de las escalas, ya que es un atributo requerido.
- Intercambiar el orden de alguno de los atributos, por ejemplo: en *detalles* poner *cia, fechaSalida, dias*, en lugar de *cia, dias, fechaSalida* como está establecido.
- Eliminar alguna de las etiquetas en uno de los cruceros, ya que todas las etiquetas definidas en el archivo DTD deben estar presentes en cada *crucero* y en el orden establecido.

XML Schema Definition (XSD)

XSD es un lenguaje utilizado para definir la estructura y las restricciones de un documento XML.

Fue desarrollado por el W3C (*World Wide Web Consortium*) y alcanzó popularidad a partir del año 2001, desplazando a la tecnología DTD.

Con DTD, definíamos las etiquetas que debía tener el documento *cruceros.xml* y el orden en que debían aparecer, pero no era posible especificar el tipo de dato asociado a las etiquetas elementales, es decir, no había manera de indicar que el *destino* del *crucero* es un campo de texto y los *días* es un dato numérico. Esta limitación fue superada por los archivos XSD.



Para facilitar la comprensión de esta nueva tecnología, no vamos a desarrollar el esquema XSD completo, sino que iremos dando pasos hasta llegar a la versión completa.

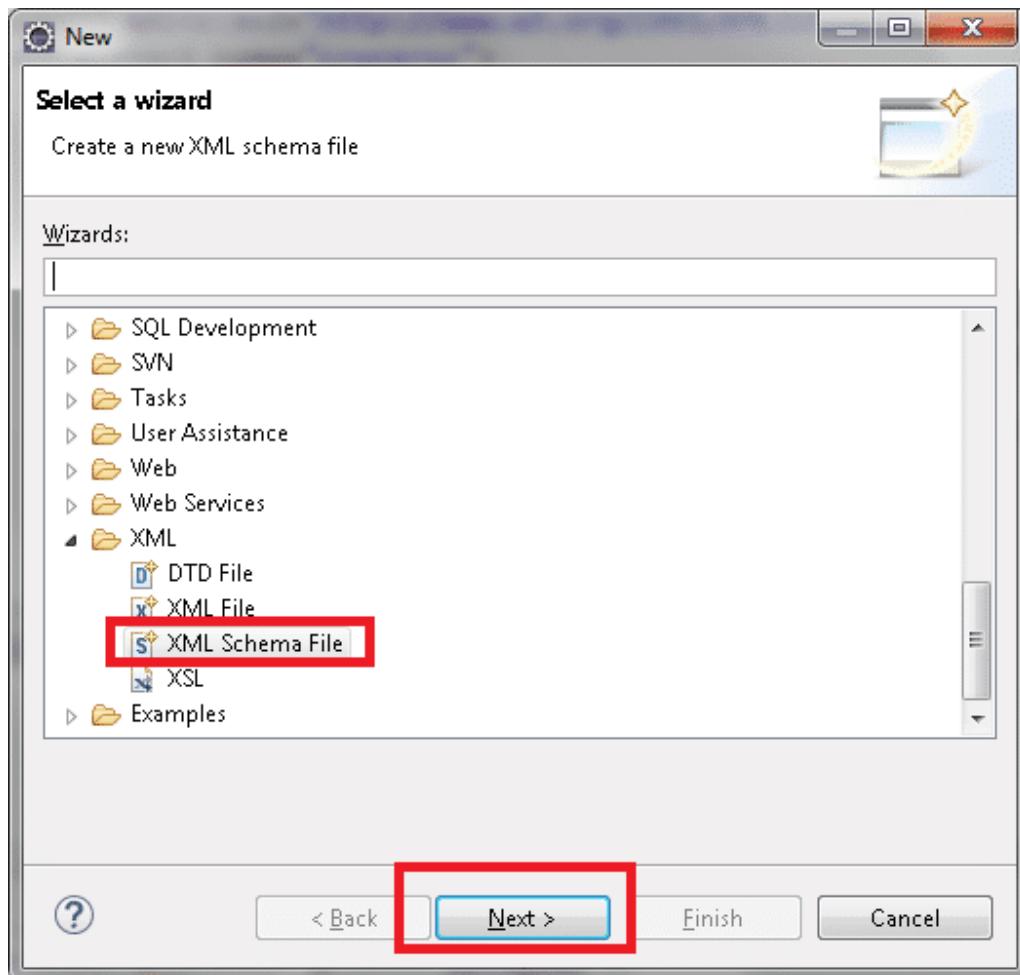
Sigue estos pasos para crear la primera versión del archivo XSD en Eclipse:

1

Haz clic derecho sobre el nombre del proyecto y selecciona **New / Other** en el menú contextual.

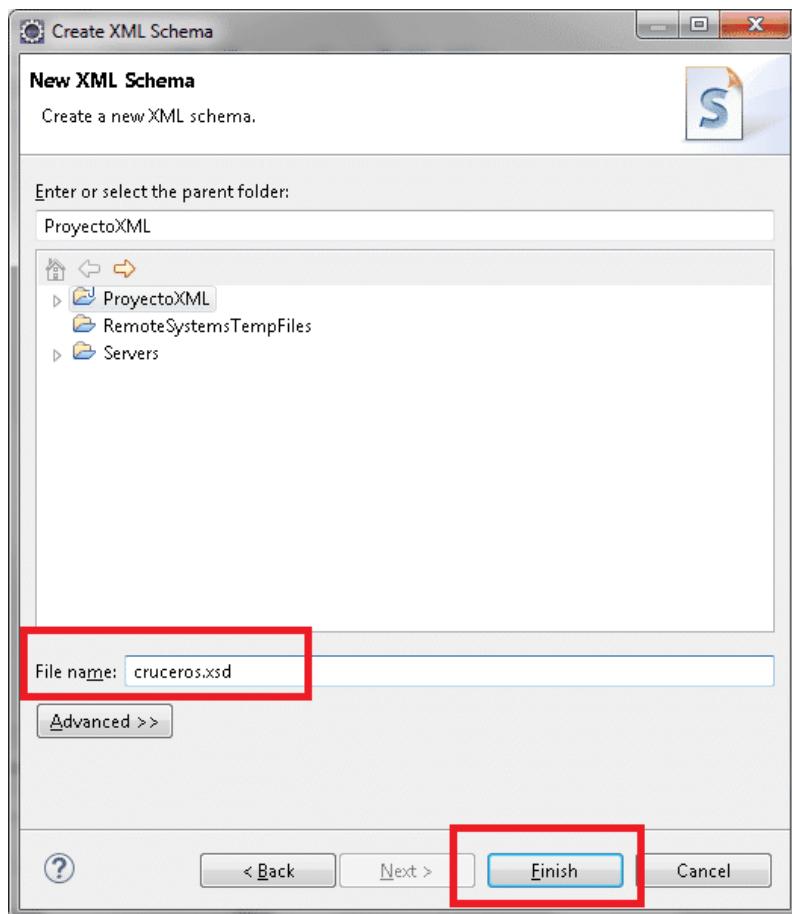
2

Utiliza la barra de desplazamiento vertical en el cuadro de diálogo **New** si es necesario hasta encontrar la carpeta **XML**, y selecciona la opción **XML Schema File**. Luego pulsa el botón **Next >**.



3

Escribe **cruceros.xsd** como nombre de archivo y pulsa el botón **Finish**.



4

Sustituye el contenido que viene por defecto por el código siguiente:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <xsd:element name="cruceros">
        <xsd:complexType>
            <xsd:sequence minOccurs="0" maxOccurs="unbounded">
                <xsd:element name="crucero">
                    <xsd:complexType>
                        <xsd:sequence>
                            <xsd:element name="destino" type="xsd:string">
                                <xsd:element name="detalles"> </xsd:element>
                                <xsd:element name="escalas"> </xsd:element>
                            </xsd:sequence>
                            <xsd:attribute name="codigo" type="xsd:string" />
                        </xsd:complexType>
                    </xsd:element>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
    </xsd:schema>
```

Vamos a analizar el código detenidamente:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    .....
</xsd:schema>
```

En primer lugar, un **archivo XSD** es un documento XML con etiquetas especiales que tienen como función definir la estructura y restricciones de otro documento XML. En nuestro caso, el archivo *cruceros.xsd* tendrá como objetivo definir la estructura del archivo *cruceros.xml*.

Los archivos XSD tienen que contar con una etiqueta principal *<xsd:schema>* que encierra el resto de etiquetas que irán configurando las distintas restricciones.

Con el atributo *xmlns:xsd* y el valor *http://www.w3.org/2001/XMLSchema* estamos indicando la URL donde se definen los tipos de datos y etiquetas especiales para los archivos XSD. También estamos indicando que debemos utilizar el prefijo *xsd* en cada una de las etiquetas.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <xsd:element name="cruceros">
        <xsd:complexType>
            </xsd:complexType>
    </xsd:element>
</xsd:schema>
```

Recuerda que se trata de definir la estructura del documento *cruceros.xml*. Pues bien, estamos indicando que debe constar, en primer lugar, una etiqueta o elemento llamado *cruceros* y que, además, es de *tipo complejo*, es decir, es una etiqueta que se subdivide en más etiquetas. Entre las etiquetas `<xsd:complexType> </xsd:complexType>` será necesario configurar la composición que debe tener la etiqueta *cruceros*.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <xsd:element name="cruceros">
        <xsd:complexType>
            <xsd:sequence minOccurs="0" maxOccurs="unbounded">
                <xsd:element name="crucero">
                </xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
</xsd:schema>
```

En las nuevas líneas añadidas indicamos que el elemento *cruceros* está compuesto por una secuencia que va desde 0 a infinitos elementos de tipo *crucero*.

```
.....
<xsd:element name="crucero">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element name="destino" type="xsd:string" />
            <xsd:element name="detalles"> </xsd:element>
            <xsd:element name="escalas"> </xsd:element>
        </xsd:sequence>
        <xsd:attribute name="codigo" type="xsd:string" />
    </xsd:complexType>
</xsd:element>
....
```

Ahora estamos indicando que cada elemento *crucero* es de tipo complejo, es decir, que se descompone en otras etiquetas. Además indicamos que cada *crucero* está compuesto por una secuencia de tres etiquetas denominadas *destino* (elemental y de tipo *string*), *detalles* y *escalas*.

Todavía queda trabajo pendiente, ya que será necesario definir, además, la composición de las etiquetas *detalles* y *escalas*, que también son de tipo complejo. No obstante, aunque no hayamos terminado del todo, ya podemos enlazar el archivo XSD con el documento *cruceros.xml* y comenzar a ver el resultado de nuestro trabajo.

Para enlazar el archivo *cruceros.xml* con el esquema XSD que acabamos de crear, tienes que añadir unos atributos especiales a la etiqueta *cruceros*.

Para empezar, debes editar el archivo *cruceros.xsd* para que comience de la siguiente manera:

```
<?xml version="1.0" encoding="UTF-8"?>
<cruceros xmlns:xsd="http://www.w3.org/2001/XMLSchema-instance"
    xsd:noNamespaceSchemaLocation="cruceros.xsd">
```



Como prueba, puedes editar el archivo *cruceros.xsd* en Eclipse y realizar algún cambio que contradiga al archivo XSD. Por ejemplo, sustituye el nombre de la etiqueta *destino* por *destin* y comprueba cómo eclipse marca dicha etiqueta con subrayado rojo como error.

The screenshot shows the Eclipse IDE interface with two tabs: *cruceros.xml* and cruceros.xsd. The cruceros.xsd tab is active, displaying the XML schema code. The cruceros.xml tab is also visible. The XML code in cruceros.xml includes a root element *cruceros* with attributes *version*, *encoding*, and *xsd:namespaceSchemaLocation*. Inside, there's an *crucero* element with attribute *codigo* set to "CRUMED21". This element contains a *destin* element with the value "Mediterraneo (Grecia, Italia)". A tooltip or validation message is displayed over the *destin* element, stating: "cvc-complex-type.2.4.a: Invalid content was found starting with element 'destin'. One of '{destino}' is expected." The cursor is positioned at the start of the word "días" in the *dias* element.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <cruceros xmlns:xsd="http://www.w3.org/2001/XMLSchema-instance"
3     xsd:namespaceSchemaLocation="cruceros.xsd">
4
5     <crucero codigo="CRUMED21">
6         <destin>Mediterraneo (Grecia, Italia)</destin>
7         <cvc-complex-type.2.4.a: Invalid content was found starting with element 'destin'. One of '{destino}' is expected.
8             Press 'F2' for focus
9             <dias>6 días</dias>
10            <fechaSalida>2018-12-26</fechaSalida>
11        </detalles>
```

Si sitúas el puntero de razón sobre el subrayado rojo, te aparecerá la descripción del error en un recuadro amarillo.

Crea la versión final de XML SCHEMA:

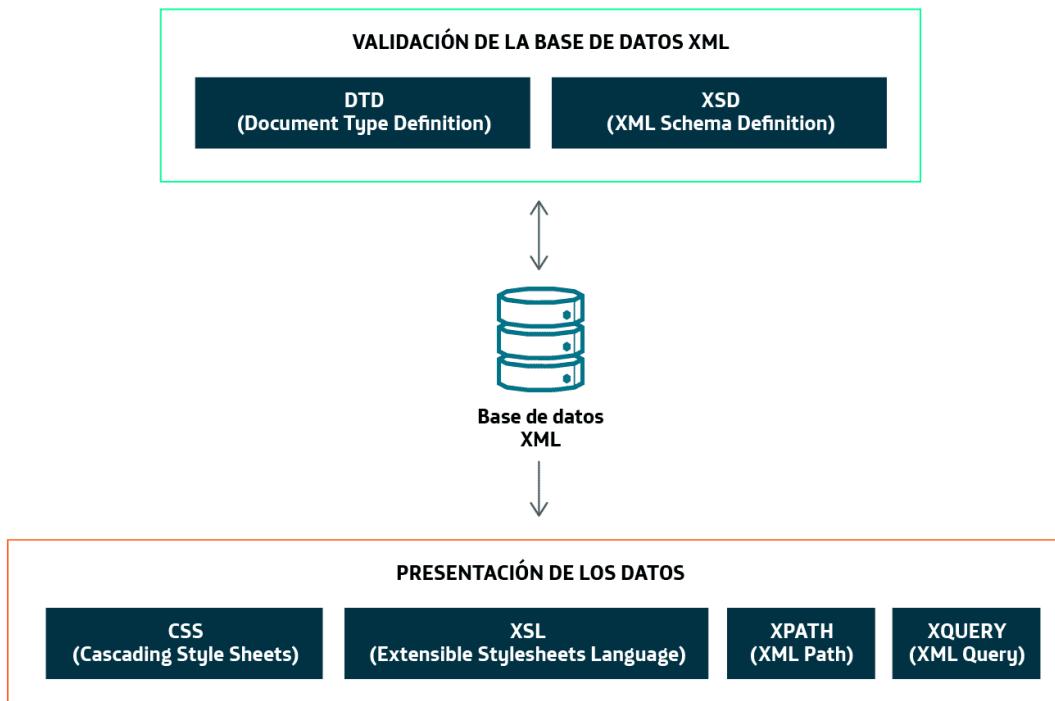
Todavía tenemos pendiente la definición de la estructura de los elementos *detalles* y *escalas*. Ambos son de tipo complejo, pero *detalles* se descompone en otros tres elementos que sólo se repiten una vez, mientras que *escalas* es una secuencia de 0 a infinitos elementos *escala*, cada uno de ellos elementos complejos compuestos por los subelementos *parada*, *llegada* y *salida*.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <xsd:element name="cruceros">
        <xsd:complexType>
            <xsd:sequence minOccurs="0" maxOccurs="unbounded">
                <xsd:element name="crucero">
                    <xsd:complexType>
                        <xsd:sequence>
                            <xsd:element name="destino" type="xsd:string" />
                            <xsd:element name="detalles">
                                <xsd:complexType>
                                    <xsd:sequence>
                                        <xsd:element name="cia" type="xs-
d:string" />
                                        <xsd:element name="dias" type="xs-
d:string" />
                                        <xsd:element name="fechaSalida"
type="xsd:date" />
                                    </xsd:sequence>
                                </xsd:complexType>
                            </xsd:element>
                            <xsd:element name="escalas">
                                <xsd:complexType>
                                    <xsd:sequence minOccurs="0" maxOccur-
s="unbounded">
                                        <xsd:element name="escala">
                                            <xsd:complexType>
                                                <xsd:sequence>
                                                    <xsd:element
name="parada" type="xsd:string" />
                                                    <xsd:element
name="llegada" type="xsd:string" />
                                                    <xsd:element
name="salida" type="xsd:string" />
                                                </xsd:sequence>
                                                <xsd:attribute name="-
dia" type="xsd:int" />
                                            </xsd:complexType>
                                        </xsd:element>
                                    </xsd:sequence>
                                </xsd:complexType>
                            </xsd:element>
                            <xsd:sequence>
                                <xsd:attribute name="codigo" type="xsd:string" />
                            </xsd:sequence>
                        </xsd:complexType>
                    </xsd:element>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
    </xsd:schema>
```

Tecnologías asociadas a las bases de datos XML

Ya conoces las tecnologías asociadas a las bases de datos XML: tecnologías para validación (DTD, XSD) y tecnologías para la presentación de los datos al usuario (CSS, XSL, XPATH, XQUERY).

A modo de resumen, la siguiente imagen esquematiza todas estas tecnologías:



Ya estudiaste en la primera lección de esta unidad las tecnologías para la presentación de los datos CSS, XSL y XPATH. Pero te falta por aprender el lenguaje para la selección de datos XQUERY, al que dedicaremos una lección completa más adelante.

Ahora que conoces las bases de datos XML, vamos a profundizar en las bases de datos documentales.

Qué son las bases de datos documentales

Una **base de datos documental** es un conjunto de información estructurada en registros, donde cada registro constituye una unidad autónoma de información que puede estar, a su vez, estructurada en diferentes niveles, constituyendo una estructura jerárquica o en árbol.

El concepto de base de datos XML está directamente relacionado con las **bases de datos documentales**.

En las bases de datos documentales, cada registro se corresponde con un documento completo de cualquier tipo: un albarán, una factura, la ficha de un libro, etc.

Considerando nuestro archivo *cruceros.xml* como una base de datos, cada etiqueta *crucero* constituye un registro a partir del cual podemos obtener un documento completo que describe a un crucero.

Base de datos XML vs Base de datos relacional

En este apartado vamos a **comparar la estructura de una base de datos relacional con la de una base de datos XML**.

- Para obtener un documento en una base de datos relacional, necesitamos consultar varias relaciones o tablas y, por cada una de ellas, varios registros o filas. Pensemos en la impresión de una factura: para obtener dicho documento necesitamos consultar en una base de datos las tablas de clientes, artículos, facturas, detalles de facturas, etc.
- Para obtener un documento en una base de datos XML, que corresponde con un formato documental, necesitamos consultar un único registro, que constituye una unidad autónoma.

Una base de datos relacional que contenga la información de nuestros cruceros podría tener la siguiente estructura:

CRUCERO

CODIGO	DESTINO	CIA	DIAS	FECHA
CRUMED21	Mediterraneo (Grecia, Italia)	Costa cruceros	6 días	2018-12-26
CRUATL22	Atlántico (España, Marruecos, Portugal)	MSC Cruceros	7 días	2019-02-13

ESCALA

CODIGO	DIA	PARADA	LLEGADA	SALIDA
CRUMED21	1	Venecia		18:00
CRUMED21	2	Navegación		
CRUMED21	3	Agostini	7:00	14:00
CRUMED21	4	Santorini	9:00	20:00
CRUMED21	5	Bari	8:00	14:00
CRUMED21	6	Venecia	8:30	
CRUATL22	1	Barcelona		18:00
CRUATL22	2	Navegación		
CRUATL22	3	Casablanca	7:00	22:00
CRUATL22	4	Navegación		
CRUATL22	5	Sta Cruz de Tenerife	9:00	16:00
CRUATL22	6	Funchal	8:00	19:00
CRUATL22	7	Barcelona	17:00	

Ejemplo de base de datos relacional.

Para obtener un documento descriptivo de uno de los cruceros, habrá que consultar una fila de la tabla *CRUCERO* y varias filas de la tabla *ESCALA*.

Cada registro en la base de datos XML de cruceros está encerrado en una etiqueta *<crucero>*, que tiene una estructura como esta:

```
<crucero codigo="CRUMED21">
    <destino>Mediterraneo (Grecia, Italia)</destino>
    <detalles>
        <cia>Costa cruceros</cia>
        <dias>6 días</dias>
        <fechaSalida>2018-12-26</fechaSalida>
    </detalles>
    <escalas>
        <escala dia="1">
            <parada>Venecia</parada>
            <llegada></llegada>
            <salida>18:00</salida>
        </escala>
        <escala dia="2">
            <parada>Navegación</parada>
            <llegada></llegada>
            <salida></salida>
        </escala>
        .....
    </escalas>
</crucero>
```

Estructura de la etiqueta *crucero*.

Cada elemento *crucero* encierra un registro que contiene toda la información necesaria para elaborar un documento descriptivo del crucero.

Resumen

Has terminado la lección, repasemos los puntos más importantes que hemos tratado.

- Una **base de datos XML** está constituida por uno o varios documentos XML que siguen una estructura lógica similar a la estructura de filas y columnas de una base de datos tradicional, pero con varios niveles. Esta estructura debe ser validada utilizando las tecnologías disponibles a este efecto.
- Para que un archivo XML pueda ser considerado base de datos, o parte de una base de datos, debe cumplir unos **criterios de validación** que pueden establecerse con una de las siguientes técnicas: *Document Type Definition (DTD)* o *XML Schema Definition (XSD)*.
- El concepto de base de datos XML está directamente relacionado con las llamadas bases de datos documentales. Una **base de datos documental** es un conjunto de información estructurada en registros, donde cada registro constituye una unidad autónoma de información que puede estar, a su vez, estructurada en diferentes niveles, constituyendo una estructura jerárquica o en árbol.
- En las bases de datos documentales, cada registro se corresponde con un documento completo de cualquier tipo, un albarán, una factura, la ficha de un libro, etc.



PROEDUCA