



UNIDAD 4.

XML: creación, validación y utilización

Contenidos:

1. Introducción, evolución y estado actual
2. Estructura y sintaxis de XML
3. Validación de XML
4. XML aplicado
5. Herramientas de visualización y edición de XML

XML-Schema.



- **Es el lenguaje utilizado para describir la estructura, relaciones y restricciones de los documentos XML.**
- **Mejora la precisión proporcionada por los DTD.**
- **Se definen en ficheros XML como extensión .xsd.**
- **Su uso es menos frecuente salvo en entornos más académicos o especializados.**

XML-Schema.



Las aportaciones más relevantes que XML Schema proporciona son las siguientes:

- **Permite determinar qué elementos y atributos admite un XML.**
- **Permite determinar qué tipos de datos pueden utilizar los elementos y atributos, habilitando la creación de nuevos tipos.**
- **Permite asignar valores por defecto a elementos y atributos.**
- **Permite determinar cardinalidades precisas.**
- **Permite determinar las relaciones entre los elementos del documento XML: qué hijos tiene cada elemento, en qué número y en qué orden.**

XML-Schema. Estructura básica



- Declaración del documento XML.

```
<?xml version="1.0" encoding="UTF-8"?>
```

Declaración del elemento raíz y del espacio de nombres.

```
<xs:schema
```

```
xmlns:xs="http://www.w3.org/2001/XMLSchema">
```

Declaración de los elementos: En esta sección se indican los elementos que componen la estructura del XML, su jerarquía, relación, atributos y demás restricciones.

Declaración de los atributos: Asociados a los elementos, se determina qué atributos y qué valores son admitidos.

XML-Schema. Referencia en mi XML



- Se referencia desde el documento XML a validar:

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<elemento-raiz
```

```
xmlns:xs="http://www.w3.org/2001/XMLSchema-instance"
```

```
xs:noNamespaceSchemaLocation="fichero-esquema.xsd">
```

Conceptos básicos de XML Schema (XSD)

XML Schema define la estructura de un documento XML mediante la especificación de elementos y atributos, sus tipos de datos y restricciones.

Principales etiquetas de XSD:

- **<xs:schema>**: Define el esquema XML.
- **<xs:element>**: Define un elemento en el documento XML.
- **<xs:complexType>**: Define un tipo de datos complejo que puede contener otros elementos.
- **<xs:sequence>**: Especifica el orden en que deben aparecer los elementos dentro de un tipo complejo.
- **<xs:simpleType>**: Define un tipo de datos simple con restricciones.
- **<xs:restriction>**: Restringe los valores permitidos para un tipo de dato.
- **<xs:minInclusive>** y **<xs:maxInclusive>**: Definen los valores mínimo y máximo permitidos.
- **<xs:attribute>**: Define un atributo dentro de un elemento.

XML-Schema. Tipos de datos.

Tipo (utilizando xs como prefijo)	Descripción
xs:string	Cadena de caracteres.
xs:integer	Números enteros.
xs:decimal	Números decimales, utilizando el punto como separador decimal.
xs:boolean	Tipo de dato lógico. Admite los valores «true» y «false».
xs:date	Fechas en formato «AAAA-MM-DD» donde AAAA indica el año, MM el mes y DD el día del mes.
xs:time	Horas en formato «hh:mm:ss» donde hh indica la hora, mm el minuto y ss el segundo.
xs:duration	Un periodo de tiempo en formato «PnYnMnDTnHnMnS», donde P indica un periodo; nY , número de años; nM , número de meses; nD , número de días; T , el comienzo de la sección referente al tiempo; nH , número de horas; nM , número de minutos, y nS , número de segundos.

XML-Schema. Tipos de elementos. Simples



- **La sintaxis básica de un elemento es la siguiente:**

```
<xs:element name=""></xs:element>
```

Siendo element la etiqueta que determina que es un elemento y el atributo name el que determina el nombre de este.

```
<xs:element name="email" type="xs:string"/>
```

```
<xs:element name="edad" type="xs:integer"/>
```

XML-Schema. Tipos de elementos. Simples



Atributos para la definición del elemento (NO SON ATRIBUTOS)

- **type.** Tipo de dato.
- **default.** Permite asignar un valor por defecto.
- **fixed.** Determina el valor del atributo en caso de que este exista.
- **minOccurs.** Número mínimo de ocurrencias del elemento. El valor por defecto es 1. Si se quiere que el número mínimo de ocurrencias sea ilimitado el valor que debe tomar es <<unbounded>>.
- **maxOccurs.** Número máximo de ocurrencias del elemento. El valor por defecto es 1. Si se quiere que el número máximo sea ilimitado, el valor que debe tomar es «unbounded».

XML-Schema. Tipos de elementos. Complejos



Los elementos de tipo complejo pueden actuar de contenedor. Se declaran utilizando la etiqueta `complexType` y pueden:

- **Contener a otros elementos, pudiendo contener opcionalmente atributos.**
- **Estar vacíos, pudiendo contener opcionalmente atributos.**
- **Tener contenido mixto: otros elementos y texto, pudiendo contener opcionalmente atributos.**

XML-Schema. Tipos de elementos. Complejos



La sintaxis básica es la siguiente:

```
<xs:element name="" type="">  
  <xs:complexType/>  
</xs:complexType>  
</xs:element>
```

XML-Schema. Sub-elementos.

Son contenidos por elementos. Tipos:

- **xs:sequence.** Secuencia de elementos obligatorios. (Deben aparecer todos y en orden)
- **xs:choise.** Secuencias de elementos de los que debe aparecer uno.
- **xs:all.** Secuencia de elementos opcionales. (No tienen porqué aparecer y no hay orden)

Ejemplo:

```
<xs:element name="direccion">
  <xs:complexType>
    <xs:sequence>
      <xs:choice> //Choice no puede ir dentro de xs:all solamente sequence
        <xs:element name="particular" type="xs:string"/>
        <xs:element name="profesional" type="xs:string"/>
      </xs:choice>
      <xs:element name="nombre" type="xs:string"/>
      <xs:element name="edad" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

XML-Schema. Sub-elementos.

Este fragmento de documento XML sería válido según la regla anterior:

```
<direccion>
  <particular>Calle San Miguel, 52</particular>
  <!--<profesional>Calle Lubet, 345</profesional> -->
  <nombre>Ana</nombre>
  <edad>40</edad>
</direccion>
```

Solamente puede estar la etiqueta particular o profesional, nunca ambas-

Ejemplo 1

Archivo XML (mensaje.xml)

```
<?xml version="1.0" encoding="UTF-8"?>
<aviso xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="mensaje.xsd">
    <de>David</de>
    <para>Rosalía</para>
    <mensaje>Mañana nos vemos a las 10.</mensaje>
</aviso>
```

Archivo XSD (mensaje.xsd)

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="aviso">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="de" type="xs:string"/>
                <xs:element name="para" type="xs:string"/>
                <xs:element name="mensaje" type="xs:string"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
</xs:schema>
```

Ejemplo 2: Validación de Datos de Contacto

Archivo XML (contacto.xml)

```
<?xml version="1.0" encoding="UTF-8"?>
<contacto xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="contacto.xsd">
  <nombre>Juan Pérez</nombre>
  <edad> 34</edad>
  <email>juan.perez@example.com</email>
  <telefono>+34123456789</telefono>
</contacto>
```

Archivo XSD (contacto.xsd)

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="contacto">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="nombre" type="xs:string"/>
        <xs:element name="edad" type="xs:integer"/>
        <xs:element name="email" type="xs:string"/>
        <xs:element name="telefono" type="xs:string"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```


Ejemplo 3: Validación de Empleados

Archivo XML (empresa.xml)

```
<?xml version="1.0" encoding="UTF-8"?>
<empresa nombreEmpresa="TechCorp"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="empresa.xsd">
  <empleado id="1">
    <nombre>John</nombre>
    <apellido>Doe</apellido>
    <fechaNacimiento>1990-05-15</fechaNacimiento>
    <salario>50000.00</salario>
    <activo>true</activo>
    <horaEntrada>09:00:00</horaEntrada>
  </empleado>
  <empleado id="2">
    <nombre>Jane</nombre>
    <apellido>Smith</apellido>
    <fechaNacimiento>1985-08-20</fechaNacimiento>
    <salario>60000.50</salario>
    <activo>false</activo>
    <horaEntrada>08:30:00</horaEntrada>
  </empleado>
</empresa>
```

Ejemplo 3: Validación de Empleados

Archivo XSD (empresa.xsd)

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="empresa">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="empleado" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="nombre" type="xs:string"/>
              <xs:element name="apellido" type="xs:string"/>
              <xs:element name="fechaNacimiento" type="xs:date"/>
              <xs:element name="salario" type="xs:decimal"/>
              <xs:element name="activo" type="xs:boolean"/>
              <xs:element name="horaEntrada" type="xs:time"/>
            </xs:sequence>
            <xs:attribute name="id" type="xs:integer" use="required"/>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="nombreEmpresa" type="xs:string" use="required"/>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

XML-Schema. Tipos de elementos. Ejemplo XML y XSD

```
<?xml version="1.0" encoding="UTF-8"?>
<factura
  xmlns:xs="http://www.w3.org/2001/XMLSchema-instance"
  xs:noNamespaceSchemaLocation="factura.xsd">
  <numero>1</numero>
  <cliente>
    <nombre>Ediciones Paraninfo S.A.</nombre>
    <CIF>A81461477</CIF>
  </cliente>
  <concepto>
    <descripcion>Imprenta</descripcion>
    <importe>1000</importe>
  </concepto>
  <concepto>
    <descripcion>Maquetación</descripcion>
    <importe>500.50</importe>
  </concepto>
  <impuesto nombre="IVA">PENDIENTE</impuesto>
  <totalFactura euros="1500.50"/>
</factura>
```

XML-Schema. Tipos de elementos. Ejemplo XML y XSD

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="factura">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="numero" type="xs:string"/>
        <xs:element name="cliente" type="cliente"/>
        <xs:element name="concepto" type="concepto" minOccurs="0"
maxOccurs="unbounded" />
        <xs:element name="impuesto">
          <xs:complexType mixed="true">
            <xs:attribute name="nombre" type="xs:string"/>
          </xs:complexType>
        </xs:element>
        <xs:element name="totalFactura" type="totalFactura"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="cliente">
    <xs:sequence>
      <xs:element name="nombre"/>
      <xs:element name="CIF"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="concepto">
    <xs:sequence>
      <xs:element name="descripcion"/>
      <xs:element name="importe"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="totalFactura">
    <xs:attribute name="euros" type="xs:decimal"/>
  </xs:complexType>
</xs:schema>
```

XML-Schema. Tipos de elementos. Ejemplo XML y XSD

En este bloque se declara un elemento global complejo vacío. El elemento **totalFactura** únicamente tiene un atributo **euros** de tipo **xs:decimal**:

```
<xs:complexType name="totalFactura">  
  <xs:attribute name="euros" type="xs:decimal"/>  
</xs:complexType>
```

A continuación, se muestra el código XML válido para el bloque XML Schema anterior:

```
<totalFactura euros="1500.50"/>
```

XML-Schema. Tipos de elementos. Ejemplo XML y XSD

En este bloque se declara un elemento complejo global contenedor de otros elementos. El elemento cliente contiene dos elementos nombre y CIF:

```
<xs:complexType name="cliente">
  <xs:sequence>
    <xs:element name="nombre"/>
    <xs:element name="CIF"/>
  </xs:sequence>
</xs:complexType>
```

A continuación, se muestra el código XML válido para el bloque XML Schema anterior:

```
<cliente>
  <nombre>Ediciones Paraninfo S.A.</nombre>
  <CIF>A81461477</CIF>
</cliente>
```

XML-Schema. Tipos de elementos. Ejemplo XML y XSD

En este bloque se declara un elemento local complejo mixto. El elemento impuesto contiene un atributo nombre de tipo xs:string y, opcionalmente, un bloque de texto.

```
<xs:element name="impuesto">
  <xs:complexType mixed="true">
    <xs:attribute name="nombre" type="xs:string"/>
  </xs:complexType>
</xs:element>
```

A continuación, se muestra el código XML válido para el bloque XML Schema anterior:

```
<impuesto nombre="IVA">PENDIENTE</impuesto>
```

XML-Schema. Atributos. Características



- Deben aparecer inmediatamente después de la declaración del elemento al que hacen referencia.
- El atributo **name** indica el nombre del atributo.
- El atributo **type** indica el tipo del atributo. Deben ser de tipo simple. El tipo por defecto es **anySimpleType**, un tipo de dato genérico que admite cualquier tipo simple.
- El atributo **use** hace referencia a su obligatoriedad, admitiendo los valores **required** (obligatorio), **optional** (opcional, valor por defecto) y **prohibited** (el atributo no se puede utilizar en el elemento).
- El atributo **default** permite asignar un valor por defecto.
- El atributo **fixed** determina el valor del atributo en caso de que este exista.
- No tienen orden de aplicación.
- No tienen cardinalidad.
- No pueden tener hijos.
- Los tipos de valores de los atributos se pueden restringir con las **restricciones**.

XML-Schema. Atributos.

Sintaxis básica: `<xs:attribute name="" type="" use="" />`

Ejemplo: declaración de dos atributos (nombre de tipo string y urgencia de tipo integer) pertenecientes al elemento destinatario.

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="destinatarios">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="destinatario" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="nombreCompleto" type="xs:string" />
              <xs:element name="telefono" type="xs:string" />
            </xs:sequence>
            <xs:attribute name="nombreCorto" type="xs:string" use="required" />
            <xs:attribute name="urgencia" type="xs:integer" use="optional" />
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```


XML-Schema. Facetas o Restricciones

Permiten detallar las restricciones de los elementos o atributos.

Faceta	Descripción
xs:length	Determina una longitud fija.
xs:minLength	Establece una longitud mínima.
xs:maxLength	Fija una longitud máxima.
xs:totalDigits	Determina el máximo número de dígitos que puede tener un número.
xs:fractionDigits	Establece el máximo número de decimales que puede tener un número.
xs:minExclusive	Determina que el valor debe ser mayor que el valor indicado.
xs:maxExclusive	Establece que el valor debe ser menor que el valor indicado.
xs:minInclusive	Fija que el valor debe ser mayor o igual que el valor indicado.
xs:maxInclusive	Determina que el valor debe ser menor o igual que el valor indicado.
xs:enumeration	Establece una lista de valores posibles.
xs:whiteSpace	Determina cómo tratar los espacios en blanco, las tabulaciones y los saltos de línea.
xs:pattern	Fija un patrón de caracteres permitidos.

XML-Schema. Facetas o Restricciones: ejemplo max min

Ejemplo de restricción de mínimo y máximo:



```
<xs:element name="diaSemana">
  <xs:simpleType>
    <xs:restriction base="xs:integer">
      <xs:minInclusive value="1"/>
      <xs:maxInclusive value="7"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

Un posible elemento correcto en el XML sería el siguiente:

```
<diaSemana>7</diaSemana>
```

XML-Schema. Facetas o Restricciones: ejemplo enumeración

Ejemplo de restricción de enumeración:



```
<xs:element name="diaSemana">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:enumeration value="lunes"/>
      <xs:enumeration value="martes"/>
      <xs:enumeration value="miércoles"/>
      <xs:enumeration value="jueves"/>
      <xs:enumeration value="viernes"/>
      <xs:enumeration value="sábado"/>
      <xs:enumeration value="domingo"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

Un posible elemento correcto en el XML sería el siguiente:

```
<diaSemana>sábado</diaSemana>
```

XML-Schema. Facetas o Restricciones: ejemplo patrón

Ejemplo de restricción de patrón:

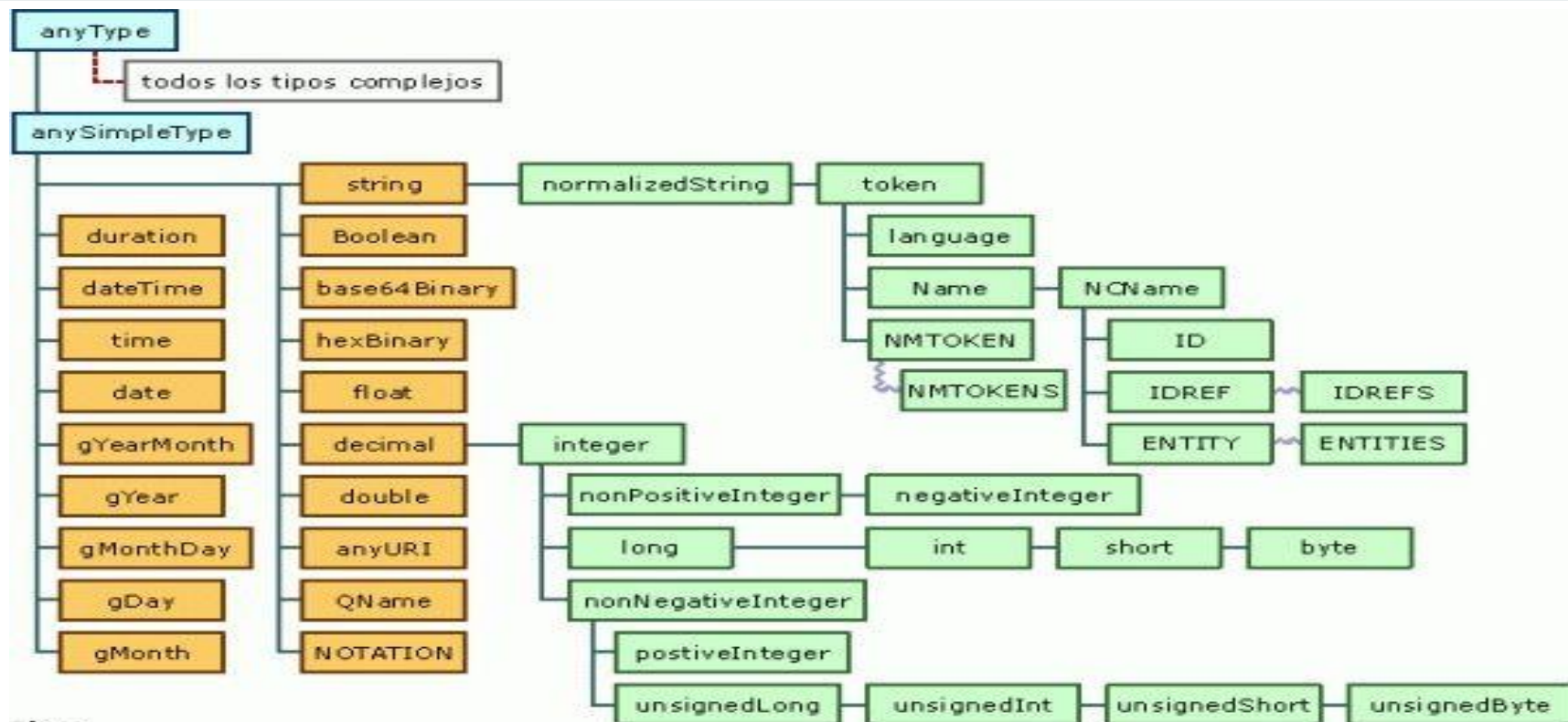
```
<xs:element name="clave">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="[a-zA-Z]{4}[0-9]{2}" />
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

La expresión indicada en el value significa que se requieren cuatro letras, ya sean estas minúsculas o mayúsculas seguidas de dos dígitos.

Un posible elemento correcto en el XML sería el siguiente:

```
<clave>PanI91</clave>
```


XML-Schema. Tipos de datos.



Clave



XML-Schema. Comentarios.



- Se indican con el elemento `<annotation>`.
- Se puede agregar en cualquier lugar del documento.
- Puede ser contenedor de los elementos `xs:documentation` y `xs:appinfo`, además de permitir cualquier otro tipo de contenido, incluido HTML.

```
<xs:annotation>
  <xs:appinfo>Validador de mensajes</xs:appinfo>
  <xs:documentation xml:lang="es">
    Este esquema valida mensajes privados seguros.
  </xs:documentation>
</xs:annotation>
```

Actividad resuelta

Validar utilizando XML Schema un documento XML que contiene un registro de temperatura con la siguiente estructura:

- El elemento principal contiene un registro de temperaturas con el nombre de la provincia, la fecha del registro, la temperatura mínima y máxima y una relación de incidencias.
- La fecha de registro debe ser una fecha válida.
- La temperatura mínima no debe ser inferior a -50 °C.
- La temperatura máxima no debe ser superior a 50 °C.
- El número de incidencias no tiene límite.
- Las incidencias tienen un nombre que puede ser "frío", "nieve", "lluvia" o "calor". Las incidencias tienen una severidad que puede ser "alta", "media" o "baja".

Actividad resuelta

```
<?xml version="1.0" encoding="UTF-8"?>
<registro_temperatura
  xmlns:xs="http://www.w3.org/2001/XMLSchema-instance"
  xs:noNamespaceSchemaLocation="actividad_resuelta_4_3.xsd">
  <provincia>Zamora</provincia>
  <fecha>2022-01-07</fecha>
  <temperatura_minima>-5</temperatura_minima>
  <temperatura_maxima>4</temperatura_maxima>
  <incidencias>
    <incidencia nombre="frio" severidad="alta"/>
    <incidencia nombre="nieve" severidad="alta"/>
    <incidencia nombre="lluvia" severidad="media"/>
  </incidencias>
</registro_temperatura>
```

Actividad resuelta: solución

Solución

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:simpleType name="nombre_incidencia">
    <xs:restriction base="xs:string">
      <xs:enumeration value="frio" />
      <xs:enumeration value="nieve" />
      <xs:enumeration value="lluvia" />
      <xs:enumeration value="calor" />
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="severidad_incidencia">
    <xs:restriction base="xs:string">
      <xs:enumeration value="baja" />
      <xs:enumeration value="media" />
      <xs:enumeration value="alta" />
    </xs:restriction>
  </xs:simpleType>
```

```
<xs:element name="registro_temperatura">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="provincia" type="xs:string"/>
      <xs:element name="fecha" type="xs:date"/>
      <xs:element name="temperatura_minima">
        <xs:simpleType>
          <xs:restriction base="xs:integer">
            <xs:minInclusive value="-50"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:element>
      <xs:element name="temperatura_maxima">
        <xs:simpleType>
          <xs:restriction base="xs:integer">
            <xs:maxInclusive value="50"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:element>
      <xs:element name="incidencias">
        <xs:complexType>
          <xs:sequence>
```

```
<xs:element name="incidencia" maxOccurs="unbounded">  
    <xs:complexType>  
        <xs:attribute name="nombre" type="nombre_"  
incidencia"/>  
        <xs:attribute name="severidad" type="severidad_  
incidencia"/>  
    </xs:complexType>  
</xs:element>  
</xs:sequence>  
</xs:complexType>  
</xs:element>  
</xs:sequence>  
</xs:complexType>  
</xs:element>  
</xs:sequence>  
</xs:complexType>  
</xs:element>  
</xs:schema>
```

Actividad propuesta

Validación de expedientes académicos con XML Schema

Crea un validador utilizando XML Schema para un XML que almacena los expedientes académicos de un conjunto de estudiantes.

La estructura del documento XML es la siguiente:

- Un documento contiene un conjunto de expedientes.
- Cada expediente contiene el nombre de la titulación, el nombre del estudiante y un conjunto de módulos por cada estudiante.
- Cada módulo tiene un atributo con el nombre, otro atributo que indica si el módulo ya ha sido aprobado y un tercer atributo que informa del curso al que pertenece el módulo y que puede tomar los valores "Primero o "Segundo".


```
<?xml version="1.0" encoding="UTF-8"?>
<expedientes>
  <expediente>
    <titulacion>DAM</titulacion>
    <estudiante>Ainhoa Gárate Lizarraga</estudiante>
    <modulos>
      <modulo nombre="Sistemas Informáticos" aprobado="true"
curso="Primero"/>
      <modulo nombre="Lenguajes de Marcas y Sistemas de Gestión de la
Información" aprobado="true" curso="Primero"/>
      <modulo nombre="Programación multimedia y dispositivos móviles"
aprobado="false" curso="Segundo"/>
    </modulos>
  </expediente>
  <expediente>
    <titulacion>DAW</titulacion>
    <estudiante>Ferran Soler Puig</estudiante>
    <modulos>
      <modulo nombre="Programación" aprobado="true" curso="Primero"/>
      <modulo nombre="Desarrollo web en entorno cliente" aprobado="false"
curso="Segundo"/>
    </modulos>
  </expediente>
</expedientes>
```

Otras alternativas de validación:



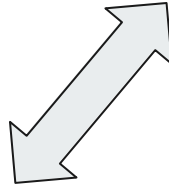
- **RELAX NG** (REgular LAnguage for XML Next Generation) fue definido por OASIS, una organización sin ánimo de lucro que trabaja en estándares de seguridad, tecnologías relacionadas con los datos o computación en la nube, además de otras disciplinas. Dispone de una sintaxis XML y de otra más compacta no XML.
- **Schematron**, por su parte, está basado en reglas y se expresa en XML. Es más potente que XML Schema y DTD, ya que contempla algunas restricciones que estos no permiten.

XML-Schema VS DTD

Book.xsd

Book.dtd

```
<!ELEMENT BookStore (Book+)>
<!ELEMENT Book (Title, Author, Date, ISBN, Publisher)>
<!ELEMENT Title (#PCDATA)>
<!ELEMENT Author (#PCDATA)>
<!ELEMENT Date (#PCDATA)>
<!ELEMENT ISBN (#PCDATA)>
<!ELEMENT Publisher (#PCDATA)>
```



```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://www.books.org"
  xmlns="http://www.books.org"
  elementFormDefault="qualified">
  <xsd:element name="BookStore">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref="Book" minOccurs="1" maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="Book">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref="Title" minOccurs="1" maxOccurs="1"/>
        <xsd:element ref="Author" minOccurs="1" maxOccurs="1"/>
        <xsd:element ref="Date" minOccurs="1" maxOccurs="1"/>
        <xsd:element ref="ISBN" minOccurs="1" maxOccurs="1"/>
        <xsd:element ref="Publisher" minOccurs="1" maxOccurs="1"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="Title" type="xsd:string"/>
  <xsd:element name="Author" type="xsd:string"/>
  <xsd:element name="Date" type="xsd:string"/>
  <xsd:element name="ISBN" type="xsd:string"/>
  <xsd:element name="Publisher" type="xsd:string"/>
</xsd:schema>
```



```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://www.books.org"
  xmlns="http://www.books.org"
  elementFormDefault="qualified">
```

```
  <xsd:element name="BookStore">
```

```
    <xsd:complexType>
```

```
      <xsd:sequence>
```

```
        <xsd:element ref="Book" minOccurs="1" maxOccurs="unbounded"/>
```

```
      </xsd:sequence>
```

```
    </xsd:complexType>
```

```
  </xsd:element>
```

```
  <xsd:element name="Book">
```

```
    <xsd:complexType>
```

```
      <xsd:sequence>
```

```
        <xsd:element ref="Title" minOccurs="1" maxOccurs="1"/>
```

```
        <xsd:element ref="Author" minOccurs="1" maxOccurs="1"/>
```

```
        <xsd:element ref="Date" minOccurs="1" maxOccurs="1"/>
```

```
        <xsd:element ref="ISBN" minOccurs="1" maxOccurs="1"/>
```

```
        <xsd:element ref="Publisher" minOccurs="1" maxOccurs="1"/>
```

```
      </xsd:sequence>
```

```
    </xsd:complexType>
```

```
  </xsd:element>
```

```
  <xsd:element name="Title" type="xsd:string"/>
```

```
  <xsd:element name="Author" type="xsd:string"/>
```

```
  <xsd:element name="Date" type="xsd:string"/>
```

```
  <xsd:element name="ISBN" type="xsd:string"/>
```

```
  <xsd:element name="Publisher" type="xsd:string"/>
```

```
</xsd:schema>
```

<!ELEMENT BookStore (Book+)>

<!ELEMENT Book (Title, Author, Date,
ISBN, Publisher)>

<!ELEMENT Title (#PCDATA)>

<!ELEMENT Author (#PCDATA)>

<!ELEMENT Date (#PCDATA)>

<!ELEMENT ISBN (#PCDATA)>

<!ELEMENT Publisher (#PCDATA)>