1. Declaración del esquema XML

La declaración del esquema XML al inicio del documento es crucial para definir el espacio de nombres (namespace) de XML Schema
que indica que el documento contiene definiciones de esquema escritas conforme a las especificaciones de XML Schema.
<xmlns:xs="http://www.w3.org/2001/XMLSchema">

2. Definición de Elementos Complejos

Los elementos complejos son aquellos que pueden contener otros elementos, atributos o una mezcla de ambos.
La estructura <xs:complexType> permite definir tales elementos, incluyendo la secuencia de subelementos (<xs:sequence>), así como los atributos (<xs:attribute>) que pueden poseer.

3. Definición de Tipos de Datos Simples

XML Schema permite crear tipos de datos personalizados (<xs:simpleType>) a partir de tipos de datos base.
Estos tipos personalizados pueden tener restricciones como longitud mínima y máxima, patrones de caracteres específicos (<xs:pattern>), rangos de valores, entre otros.
Esto es útil para validar el contenido de los elementos o atributos contra requisitos específicos.

4. Uso de Elementos Globales

Los elementos definidos globalmente (<xs:element> fuera de cualquier <xs:complexType> o <xs:simpleType>) son accesibles en cualquier parte del documento XML.
Esto facilita la reutilización de definiciones en diferentes partes del esquema.

5. Definición de Atributos

Similar a los elementos complejos, los atributos (<xs:attribute>) pueden ser definidos dentro de elementos complejos para especificar información adicional que es opcional u obligatoria, dependiendo de cómo se configure.

6. Restricciones y Validaciones en los Tipos de Datos

Las restricciones (<xs:restriction>) permiten definir qué valores son válidos para un tipo de dato específico.

Por ejemplo, se pueden enumerar valores permitidos (<xs:enumeration>), definir rangos de valores numéricos, establecer longitudes específicas para cadenas, entre otros.

7. Definición de Elementos Anidados y Referencias

Los elementos pueden contener otros elementos dentro de ellos, formando una estructura anidada.
Las referencias (ref="nombre_elemento") permiten reutilizar definiciones de elementos existentes, promoviendo la modularidad y evitando la repetición de código.

8. Uso de Elementos de Agrupación

Los grupos de elementos (<xs:group>) sirven para definir conjuntos de elementos que pueden ser reutilizados en diferentes complejos de elementos dentro del esquema, lo que ayuda a mantener la consistencia y reduce la duplicación.

9. Definición de Elementos con Contenido Mixto

El contenido mixto (mixed="true") en un <xs:complexType> permite que un elemento contenga tanto texto como subelementos.
Esto es útil para modelar estructuras de datos que pueden tener un contenido variable o que deben soportar la inserción de datos de texto entre elementos marcados.

Cada uno de estos elementos de código XML Schema desempeña un papel vital en la creación de esquemas XML robustos y precisos.
Al definir la estructura, tipos de datos, y restricciones con XML Schema, se asegura que los documentos XML cumplan con los requisitos especificados,
lo que facilita el intercambio de datos estructurados y su validación automática.

Expresiones Regulares compatibles con XMLSchema
Caracteres

Literal characters: Los caracteres literales se usan tal cual, excepto los especiales que deben ser escapados (\, ^, ., $, |, ?, *, +, (,), [, {).

Escape sequences: Las secuencias de escape permiten incluir caracteres especiales o sets de caracteres especiales como \n (nueva línea), \t (tabulación), \d (dígitos), \D (no dígitos), \w (caracteres de palabra, incluyendo letras, dígitos y el guión bajo), y \W (no caracteres de palabra).

Clases de Caracteres

Clases de caracteres personalizadas: Permiten definir un conjunto específico de caracteres utilizando corchetes. Por ejemplo, [abc] coincide con "a", "b", o "c". Puedes usar rangos dentro de las clases de caracteres como [a-z] para cualquier letra minúscula.

Cuantificadores

Cuantificadores: Los cuantificadores indican el número de veces que debe aparecer el elemento anterior:

? - Cero o una vez.
* - Cero o más veces.
+ - Una o más veces.
{n} - Exactamente n veces.
{n,} - n o más veces.
{n,m} - Entre n y m veces, ambos inclusive.

Anclas

Inicio y fin de cadena: ^ y $ se usan para indicar el inicio y el fin de la cadena respectivamente, aunque su uso en XML Schema es más limitado en comparación con otros sistemas de expresiones regulares debido al contexto en que se validan las cadenas.

Agrupación y Alternancia

Agrupación: Los paréntesis () se utilizan para agrupar partes de la expresión y aplicar cuantificadores a la agrupación completa.

Alternancia: El carácter | se utiliza para la alternancia, permitiendo que la expresión coincida con uno de varios subpatrones posibles.

Ejemplos de Ejercicios:

Ejecicio impresora mio, con comentarios

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified">
    <xs:element name="impresoras">
        <xs:complexType> <!--creamos un tipo coplejo, impresoras, que es el padre
de los siguientes complejos -->
            <xs:sequence>
                <xs:element name="impresora" maxOccurs="unbounded"> <!--ponemos el
maximo de ocurrencias en infinito-->
                    <xs:complexType> <!--nuevo grupo de tipos complejos-->
                        <xs:sequence>
                            <xs:element name="marca" type="xs:string"/> <!--marca
y modelo entran como tipo de dato string-->
                            <xs:element name="modelo" type="xs:string"/>
                            <xs:element name="peso"> <!--peso tiene restriccion de
que sea positivo y decimal con 2 cifras, se valida abajo-->
                                <xs:simpleType>
                                    <xs:restriction base="xs:decimal">
                                        <xs:fractionDigits value="2"/>
                                    </xs:restriction>
                                </xs:simpleType>
                            </xs:element>
                            <xs:element name="tamaño" type="xs:string"
maxOccurs="unbounded"/> <!--nuevos elementos hijos de impresora, tamano y
cartucho. Tamano va como
                            unbounded porque una impresora puede tener varios
tamanos de impresion-->
                            <xs:element name="cartucho"> <!--cartucho tiene una
restriccion de carateres. debe empezar por "C-", continuar con 3 numeros y 1 o 2
letras-->
                                <xs:simpleType>
                                    <xs:restriction base="xs:string">
                                        <xs:pattern value="C-[0-9]{3}[A-Z]{1,2}"/>
                                    </xs:restriction>
                                </xs:simpleType>
                            </xs:element>
                            <xs:element name="enred" minOccurs="0"/> <!--elemento
en red, es opcional y puede ir como elemento vacio-->
                        </xs:sequence>
```

```
                        <xs:attribute name="numSerie" type="xs:string"
use="required"/> <!--NumeroSerie y tipo de impresora como elementos obligatorios,
el tipo tiene una restriccion de datos que solo pueden ser los indicados-->
                        <xs:attribute name="tipo" use="required">
                            <xs:simpleType>
                                <xs:restriction base="xs:string">
                                    <xs:enumeration value="matricial"/>
                                    <xs:enumeration value="láser"/>
                                    <xs:enumeration value="tinta"/>
                                </xs:restriction>
                            </xs:simpleType>
                        </xs:attribute>
                        <xs:attribute name="compra" type="xs:positiveInteger"
use="optional"/> <!-- compra que solo puede ser un integer positivo y es
opcional-->
                    </xs:complexType>
                </xs:element>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
</xs:schema>

Ejercicio impresora de la profesora, sin comentarios
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified">
  <xs:simpleType name="tCartucho">
    <xs:restriction base="xs:string">
      <xs:pattern value="C-[0-9]{3}[A-Z]{1,2}"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:element name="impresoras">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="impresora" minOccurs="1" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element type="xs:string" name="marca"/>
              <xs:element type="xs:string" name="modelo"/>
              <xs:element name="peso">
                <xs:simpleType>
                  <xs:restriction base="xs:decimal">
                    <xs:minExclusive value="0"/>
                    <xs:fractionDigits value="2"/>
```

```
                </xs:restriction>
              </xs:simpleType>
            </xs:element>
            <xs:element type="xs:string" name="tamaño" minOccurs="1"
maxOccurs="unbounded"/>
            <xs:element type="tCartucho" name="cartucho"/>
            <xs:element name="enred" minOccurs="0">
              <xs:complexType>
            </xs:complexType>
            </xs:element>
          </xs:sequence>
          <xs:attribute name="numSerie" type="xs:ID" use="required"/>
          <xs:attribute name="tipo" use="required">
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:enumeration value="matricial"/>
                <xs:enumeration value="láser"/>
                <xs:enumeration value="tinta"/>
              </xs:restriction>
            </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="compra" type="xs:positiveInteger"/>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
 </xs:element>
</xs:schema>
```

Un ejemplo de un xml con el xsd vinculado seria

```xml
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE impresoras>
<impresoras xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="Tmartinez_martinez_daniel_impresoras.xsd">
<!--Vinculamos el archivo xsd correspondiente para la validacion-->
  <impresora numSerie="i245" tipo="láser" compra="2010">
    <marca>Epson</marca>
      <modelo>EPL300</modelo>
      <peso>4.52</peso>
      <tamaño>A4</tamaño>
      <tamaño>A5</tamaño>
      <cartucho>C-123BV</cartucho>
      <enred/>
  </impresora>
  <impresora numSerie="i246" tipo="matricial">
    <marca>HP</marca>
    <modelo>LaserJet 2410</modelo>
    <peso>3.2</peso>
    <tamaño>A4</tamaño>
    <cartucho>C-456P</cartucho>
  </impresora>
</impresoras>
```

DTD

El Document Type Definition (DTD) es una de las formas de definir la estructura, el contenido y la gramática de documentos XML. Aunque con la introducción de XML Schema, que ofrece más funcionalidades y mayor flexibilidad, el uso de DTD ha disminuido, sigue siendo relevante por su simplicidad y por ser parte de la especificación original de XML. Aquí te dejo una visión general sobre DTD:

Conceptos Básicos

Definición: Un DTD define los elementos legales y la estructura de un documento XML.
Tipos: Puede ser interno (incluido en el mismo documento XML) o externo (referenciado como un archivo DTD externo).
Sintaxis: Utiliza una sintaxis no XML para definir elementos, atributos, entidades, y más.

Componentes de un DTD

Elementos: Definen los nombres de los elementos y el modelo de contenido (qué elementos pueden contener y en qué orden).
<!ELEMENT elemento (subelemento1, subelemento2)>

Atributos: Especifican los atributos de los elementos, incluyendo el tipo de dato y si son obligatorios u opcionales.
<!ATTLIST elemento atributo1 CDATA #REQUIRED>

Entidades: Permiten definir abreviaturas para el uso frecuente de cadenas de texto o caracteres especiales.
<!ENTITY nombre "Texto o carácter especial">

Notaciones: Se usan para describir el formato de datos no XML, como puede ser en archivos multimedia o binarios.
<!NOTATION nombre SYSTEM "url">

Ventajas y Limitaciones
Ventajas:
Simplicidad: Más fácil de escribir y entender que otras definiciones de tipo de documento como XML Schema.
Compatibilidad: Soportado por todas las herramientas que trabajan con XML.

Limitaciones:
Tipos de datos limitados: No permite definir tipos de datos complejos más allá de CDATA (texto arbitrario).
Sin soporte para namespaces: No puede manejar elementos que pertenecen a diferentes namespaces.
Validación básica: Ofrece menos control sobre la validación de los datos contenidos en los elementos y atributos.
No extensible: A diferencia de XML Schema, DTD no permite la creación de tipos de datos personalizados ni la extensión de tipos existentes.

Uso de DTD
A pesar de sus limitaciones, DTD puede ser útil en situaciones donde se requiere una definición de documento simple y no se necesitan las características avanzadas proporcionadas por XML Schema. Es particularmente útil para definir documentos XML sencillos donde la validación de tipos de datos complejos no es crítica.

```
Ejemplo de DTD
    Un DTD interno simple podría verse así:

    <!DOCTYPE raiz [
    <!ELEMENT raiz (elemento1, elemento2)>
    <!ELEMENT elemento1 (#PCDATA)>
    <!ELEMENT elemento2 (#PCDATA)>
    <!ATTLIST elemento2 atributo1 CDATA #REQUIRED>
    ]>
    <raiz>
        <elemento1>Texto de ejemplo</elemento1>
        <elemento2 atributo1="valor">Otro texto</elemento2>
```

```
    </raiz>
```

En este ejemplo, se define un documento con un elemento raíz que contiene dos elementos, elemento1 y elemento2, donde elemento2 tiene un atributo requerido llamado atributo1.
Conclusión

En DTD (Document Type Definition), hay varios tipos de declaraciones que permiten definir y estructurar un documento XML. Estas declaraciones son esenciales para establecer las reglas que debe seguir el contenido de un documento XML. Los principales tipos de declaraciones en DTD incluyen:
1. Declaraciones de Elementos

Las declaraciones de elementos definen los nombres de los elementos y especifican el modelo de contenido de cada elemento, es decir, qué elementos pueden contener y en qué orden. La sintaxis básica es:

<!ELEMENT nombre_del_elemento (modelo_de_contenido)>

nombre_del_elemento: Es el nombre del elemento que se está definiendo.
modelo_de_contenido: Especifica los hijos que el elemento puede contener, así como la secuencia y la cantidad de veces que pueden aparecer.

2. Declaraciones de Atributos
Las declaraciones de atributos definen los atributos de los elementos, incluyendo el tipo de datos del atributo y si es obligatorio u opcional.

<!ATTLIST nombre_del_elemento nombre_del_atributo tipo_de_dato default_decl>

nombre_del_elemento: El elemento al cual el atributo pertenece.
nombre_del_atributo: El nombre del atributo.
tipo_de_dato: El tipo de dato del atributo, como CDATA (caracteres), ID, IDREF, NMTOKEN, etc.
default_decl: Indica si el atributo es requerido (#REQUIRED), opcional (#IMPLIED), tiene un valor predeterminado o es fijo (#FIXED).

3. Declaraciones de Entidades

Las entidades son una forma de definir abreviaturas para el uso frecuente de cadenas de texto, como caracteres especiales o bloques de texto que se repiten.

<!ENTITY nombre_de_la_entidad "valor">

nombre_de_la_entidad: El nombre asignado a la entidad.
valor: El valor de la entidad, que puede ser texto o incluso otra entidad.

4. Declaraciones de Notaciones

Las notaciones se utilizan para describir el formato de datos no XML, permitiendo referenciar formatos de datos externos.

<!NOTATION nombre_de_la_notacion PUBLIC "Identificador">

nombre_de_la_notacion: El nombre de la notación.
Identificador: Un identificador que especifica el formato de datos.

5. Declaraciones de Parámetro de Entidades

Similar a las entidades generales, pero están diseñadas para ser utilizadas dentro del DTD.

<!ENTITY % nombre_de_la_entidad_parametro "valor">

Estas entidades de parámetro pueden ser usadas para incluir fragmentos dentro del DTD o para definir condiciones condicionales.
Uso en Documentos XML

Estas declaraciones se combinan para formar un DTD, que puede ser interno (incluido dentro del mismo documento XML) o externo (referenciado desde el documento XML).
Un DTD provee un marco para validar la estructura y el contenido de un documento XML, asegurando que este cumpla con los criterios especificados para elementos, atributos, entidades y notaciones. Aunque los DTDs son menos flexibles y expresivos comparados con XML Schema, siguen siendo una herramienta útil para definiciones de documento sencillas y rápidas.

```
Ejemplo de DTD mio, con comentarios
<!DOCTYPE torneo [
    <!ELEMENT torneo (participante+)> <!--Elemento torneo con minimo 1
participante obligatorio-->
    <!ATTLIST torneo <!--Lista de atributos del elemento torneo-->
            edicion CDATA #REQUIRED
            anteriorGanador IDREF #REQUIRED> <!--El IDREF para poder hacer
referencia al ID del ganador en el interno del documento-->
    <!ELEMENT participante (nombre,edad,pais,cabezaDeSerie?)> <!--Elemento
participante con todos sus campos obligatorios y cabezaDeSerie opcional-->
    <!ATTLIST participante  <!--Lista de atributos del elemento participante, con
su ID y IDEREF para hacer referencia al ID de la pareja en el documento interno-->
            idP ID #REQUIRED
            pareja IDREF #REQUIRED>
    <!ELEMENT nombre (#PCDATA)>  <!-- Elemento nombre con PCDATA para
caracteres-->
    <!ELEMENT edad (#PCDATA)>   <!--elemento edad con PCDATA para caracteres-->
    <!ELEMENT pais (#PCDATA)>   <!--elemento pais con PCDATA para caracteres-->
```

```
    <!ELEMENT cabezaDeSerie EMPTY> <!--Elemento cabezaDeSerie con EMPTY porque no necesita tener contenido, con que aparezca en el participante es suficiente-->
]>
```

Ejemplo de la profesora, sin comentarios

```
<!ELEMENT torneo (participante+)>
<!ATTLIST torneo edicion CDATA #REQUIRED anteriorGanador CDATA #REQUIRED>
<!ELEMENT participante (nombre, edad, pais, cabezaDeSerie?)>
<!ELEMENT nombre (#PCDATA)>
<!ELEMENT edad (#PCDATA)>
<!ELEMENT pais (#PCDATA)>
<!ELEMENT cabezaDeSerie EMPTY>
<!ATTLIST participante idP ID #REQUIRED pareja IDREF #REQUIRED>
```

un ejemplo de un xml con el dtd vinculado seria

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<!DOCTYPE torneo SYSTEM "martinez_martinez_daniel_torneo.dtd"> <!--Tenemos que poner standalone no y poner el link al documento dtd-->
<torneo edicion="1998" anteriorGanador="j01">
  <participante idP="j01" pareja="j02">
    <nombre>Manuel Pérez</nombre>
    <edad>23</edad>
    <pais> España</pais>
    <cabezaDeSerie/>
  </participante>
  <participante idP="j02" pareja="j01">
    <nombre>Manuel Gómez</nombre>
    <edad>25</edad>
    <pais>España</pais>
  </participante>
  <participante idP="j03" pareja="j04">
    <nombre>Ana Puertas</nombre>
    <edad>22</edad>
    <pais> E5spaña</pais>
    <cabezaDeSerie/>
  </participante>
  <participante idP="j04" pareja="j03">
    <nombre>Paco Fraile</nombre>
    <edad>45</edad>
    <pais>España</pais>
  </participante>
</torneo>
```