No es suficiente que el XML Schema esté bien formado:
Un documento XML bien formado es un documento que cumple las reglas sintácticas de XML como:
– Empezar con la declaración XML.
– Tener **un único elemento raíz**.
– Las etiquetas de **apertura** deben tener su correspondientes etiquetas de **cierre**.
– Los elementos son "**case-sensitive**".
– Todos los elementos tienen que estar adecuadamente anidados.
– Los **atributos deben estar entre comillas**.
– Se deben utilizar las **entidades** para utilizar **caracteres especiales**.
A pesar de que los documentos estén bien formados pueden seguir teniendo errores.

Ejemplo XML:
```
<?xml version="1.0"?>
<nota>
<a>Juan</a>
<de>Susana</de>
<cabecera>Recordatorio</cabecera>
<cuerpo>¡Recuerda que tenemos reunión!</cuerpo>
</nota>
```

Ejemplo XML Schema:
```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.w3schools.com"
xmlns="http://www.w3schools.com"
elementFormDefault="qualified">

<xs:element name="nota">
        <xs:complexType>
                <xs:sequence>
                        <xs:element name="a" type="xs:string"/>
                        <xs:element name="de" type="xs:string"/>
                        <xs:element name="cabecera" type="xs:string"/>
                        <xs:element name="cuerpo" type="xs:string"/>
                </xs:sequence>
        </xs:complexType>
</xs:element>
</xs:schema>
```

Referencia a un XML Schema:
Cómo se referencia un archivo XML Schema desde un documento XML:

```
<?xml version="1.0"?>
<nota
xmlns="http://www.w3schools.com"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.w3schools.com note.xsd">

<a>Juan</a>
<de>Susana</de>
<cabecera>Recordatorio</cabecera>
<cuerpo>¡Recuerda que tenemos reunión!</cuerpo>
</nota>
```

XML Schema. Elemento raíz:

El elemento <schema> es el elemento raíz (elemento documento) de todo XML Schema:

<?xml version="1.0"?>
<xs:schema *xmlns:xs="http://www.w3.org/2001/XMLSchema"*
targetNamespace="http://www.w3schools.com"
xmlns="http://www.w3schools.com"
elementFormDefault="qualified">
...
...
</xs:schema>

XML Schema. Elementos simples I:
Un **elemento simple** es un elemento XML que sólo puede contener texto. **No puede contener ni elementos ni atributos.**
Sin embargo, la restricción "sólo texto" no es del todo cierta. El texto puede contener muchos tipos de datos. Puede ser de uno de los tipos incluidos en la definición
de XML Schema (boolean, string, date, etc.), o puede ser de un tipo definido por el usuario.
Además, se pueden añadir restricciones (facets) a los tipos de datos, para limitar así su contenido, o para requerir que ciertos datos sean conformes a un patrón específico.

XML Schema. Elementos simples II
La sintaxis de un elemento simple es:
<xs:element name="xxx" type="yyy"/>
donde xxx es el nombre del elemento e yyy es el tipo de datos del elemento.
Los tipos de datos más comunes en XML Schema son:
– xs:string
– xs:decimal
– xs:integer
– xs:boolean
– xs:date

XML Schema. Elementos simples III
Ejemplo de XML:
<nombre>Jana</nombre>
<edad>36</edad>
<fechanacimiento>1970-03-27</fechanacimiento>
Ejemplo de XML Schema:
<xs:element name="nombre" type="xs:string"/>
<xs:element name="edad" type="xs:integer"/>
<xs:element name="fechanacimiento" type="xs:date"/>

XML Schema. Elementos simples IV
Se pueden especificar valores por defecto o valores fijos para los elementos simples.
Se asignará automáticamente el valor por defecto si no se especifica otro valor para el elemento.
<xs:element name="color" type="xs:string" ***default****="rojo"/>*

Se asignará automáticamente un valor para el elemento, pero en este caso, dicho valor no puede ser modificado.
<xs:element name="color" type="xs:string" ***fixed****="rojo"/>*

XML Schema. Atributos I
Los elementos simples no pueden tener atributos.
Un elemento con atributos será considerado un tipo complejo (complex type).
Sin embargo, los atributos se definen como un tipo simple.
*<xs:****attribute*** *name="xxx" type="yyy"/>*

Donde xxx es el nombre del elemento e yyy es el tipo de datos del atributo.

XML Schema. Atributos II

Código en XML:

*<apellido **idioma**="EN">Smith</apellido>*

Código en XML Schema:

*<xs:attribute name="**idioma**" type="xs:string"/>*

XML Schema. Atributos III

Se pueden especificar valores por defecto o valores fijos para los atributos.
Se asignará automáticamente el valor por defecto si no se especifica otro valor para el atributo.

*<xs:attribute name="idioma" type="xs:string" **default**="EN"/>*

Se asignará automáticamente un valor para el atributo, pero en este caso, dicho valor no puede ser modificado.

*<xs:attribute name="idioma" type="xs:string" **fixed**="EN"/>*

Por defecto los atributos son opcionales, pero se puede hacer que éstos sean requeridos.

<xs:attribute name="idioma" type="xs:string" use="required"/>

XML Schema. Restricciones I

Las restricciones en XML Schema se denominan "FACETS"

```
<xs:element name="edad">
        <xs:simpleType>
                <xs:restriction base="xs:integer">
                        <xs:minInclusive value="0"/>
                        <xs:maxInclusive value="120"/>
                </xs:restriction>
        </xs:simpleType>
</xs:element>
```

XML Schema. Restricciones II

Para limitar el contenido de un elemento XML a un conjunto de valores, se utilizará la **restricción enumeration**.

```
<xs:element name="coche">
        <xs:simpleType>
                <xs:restriction base="xs:string">
                        <xs:enumeration value="Audi"/>
                        <xs:enumeration value="Golf"/>
                        <xs:enumeration value="BMW"/>
                </xs:restriction>
        </xs:simpleType>
</xs:element>
```

Otra **alternativa**:

```
<xs:element name= "coche" type="TipoCoche"/>

<xs:simpleType name="TipoCoche">
        <xs:restriction base="xs:string">
                <xs:enumeration value="Audi"/>
                <xs:enumeration value="Golf"/>
                <xs:enumeration value="BMW"/>
        </xs:restriction>
</xs:simpleType>
```

XML Schema. Restricciones IV

Para limitar que el contenido de un elemento XML solo pueda tener una serie de números o letras se deberá utilizar la **restricción pattern**.

```
<xs:element name="letra">
    <xs:simpleType>
        <xs:restriction base="xs:string">
            <xs:pattern value="[a-z]"/>
        </xs:restriction>
    </xs:simpleType>
</xs:element>
```

XML Schema. Restricciones V

#Los únicos valores permitidos son tres letras de la 'a' a la 'z' escritas en mayúscula.

```
<xs:pattern value="[A-Z][A-Z][A-Z]"/>
```

#Los únicos valores permitidos son tres letras de la 'a' a la 'z' escritas en mayúscula o en minúscula:

```
<xs:pattern value="[a-zA-Z][a-zA-Z][a-zA-Z]"/>
```

#El único valor permitido es una de las siguientes letras: 'x', 'y', o 'z':

```
<xs:pattern value="[xyz]"/>
```

#Los únicos valores permitidos son cinco dígitos comprendidos entre '0' y '9':

```
<xs:pattern value="[0-9][0-9][0-9][0-9][0-9]"/>
```

XML Schema. Restricciones VI

#Los únicos valores permitidos son cero o más ocurrencias de letras en minúscula desde la 'a' a la 'z':

```
<xs:pattern value="([a-z])*"/>
```

#Los únicos valores permitidos son una o más ocurrencias de dos letras. En cada par de letras, la primera irá en minúscula y la segunda en mayúscula. Por ejemplo, "sToP" sería validado por el patrón, pero no "Stop", "STOP" ó "stop":

```
<xs:pattern value="([a-z][A-Z])+"/>
```

Los únicos valores permitidos son exactamente ocho caracteres, donde esos caracteres pueden ser letras minúsculas, letras mayúsculas o un dígito del 0 al 9:

```
<xs:pattern value="[a-zA-Z0-9]{8}"/>
```

XML Schema. Restricciones VII

Para especificar como se van a tratar los caracteres en blanco, se utilizará la **restricción whiteSpace.**

```
<xs:element name="direccion">
    <xs:simpleType>
        <xs:restriction base="xs:string">
            <xs:whiteSpace value="preserve"/>
        </xs:restriction>
    </xs:simpleType>
</xs:element>
```

XML Schema. Restricciones VIII

```
<xs:element name="direccion">
    <xs:simpleType>
        <xs:restriction base="xs:string">
            <xs:whiteSpace value="replace"/>
        </xs:restriction>
    </xs:simpleType>
</xs:element>
```

Si se desea que XML elimine todos los espacios en blanco; LF, CR, tabulaciones, los espacios de principio y de final de cadena, y los espacios intermedios (dejando para este caso sólo uno)
*<xs:**whiteSpace value="collapse**"/>*

XML Schema. Restricciones IX

```
<xs:element name="password">
        <xs:simpleType>
                <xs:restriction base="xs:string">
                        <xs:length value="8"/>
                </xs:restriction>
        </xs:simpleType>
</xs:element>

<xs:element name="password">
        <xs:simpleType>
                <xs:restriction base="xs:string">
                        <xs:minLength value="5"/>
                        <xs:maxLength value="8"/>
                </xs:restriction>
        </xs:simpleType>
</xs:element>
```

XML Schema. Restricciones X
(Restricción : Descripción)
Enumeration: Define una lista de valores permitidos.
FractionDigits: Especifica el número máximo de decimales permitidos. Debe ser mayor o igual que cero.
Length: Especifica el número exacto de caracteres o ítems permitidos. Debe ser mayor o igual que cero.
MaxExclusive: Especifica el límite superior para un valor numérico (el valor debe ser inferior al número especificado).
MaxInclusive: Especifica el límite superior para un valor numérico (el valor debe ser inferior o igual al número especificado).
MaxLength: Especifica el número máximo permitido de caracteres o de ítems. Debe ser mayor o igual que cero.
MinExclusive: Especifica el límite inferior para un valor numérico (el valor debe ser superior al número especificado).
MinInclusive: Especifica el limite inferior para un valor numérico (el valor debe ser superior o igual al número especificado).
MinLength: Especifica el numero mínimo permitido de caracteres o de ítems. Debe ser mayor o igual que cero.
Pattern: Define la secuencia exacta de caracteres permitidos.
TotalDigits: Especifica el número exacto de dígitos permitidos. Debe ser mayor que cero.
WhiteSpace: Especifica como se manejan los espacios en blanco (LF, CR, tabulaciones, espacios).

XML Schema. Elementos complejos I
Un elemento complejo es un elemento XML que puede tener más elementos y/o atributos.
Hay cuatro clases de elementos complejos:
- Elementos vacíos.
- Elementos que contienen otros elementos.
- Elementos que contienen sólo texto.
- Elementos que contienen ambos: otros elementos y texto.

Nota: Cada uno de estos elementos pueden contener atributos.

XML Schema. Elementos complejos II

Elemento XML complejo **vacío**:
*<producto pid="1345"**/>***

Elemento XML complejo que contiene **sólo otros elementos** :
<empleado>
<nombre>Juan</nombre>
<apellidos>García López</apellidos>
</empleado>

Elemento XML complejo que contiene **sólo texto**:
<comida tipoC="postre">Helado</comida>

Elemento XML complejo que contiene **ambos (texto y otros elementos**):
<carta>
Estimado Sr.<nombre>Juan García</nombre>.
Su pedido <pedidoid>1032</pedidoid>
será enviado el <fechaenv>2007-03-25</fechaenv>.
</carta>

XML Schema. Elementos complejos III / IV
Se puede definir un elemento complejo en XML Schema de dos formas distintas :
1. El **elemento** "empleado" se puede **declarar directamente nombrando el elemento** :
<xs:element name="empleado">
<xs:complexType>
<xs:sequence>
<xs:element name="nombre" type="xs:string"/>
<xs:element name="apellidos" type="xs:string"/>
</xs:sequence>
</xs:complexType>
</xs:element>

2. El **elemento** "empleado" **puede tener un tipo de atributo que hace referencia al nombre del tipo complejo a usar.**
*<xs:element name="empleado" **type="InfoPersona"/>***
*<xs:**complexType name="InfoPerson a"**>*
<xs:sequence>
<xs:element name="nombre" type="xs:string"/>
<xs:element name="apellidos" type="xs:string"/>
</xs:sequence>
</xs:complexType>

Usando este método, **muchos elementos** se pueden **referir**
a la **misma definición de tipo complejo** :
<xs:element name="empleado" type="InfoPersona"/>
<xs:element name="alumno" type="InfoPersona"/>
<xs:element name="Socio" type="InfoPersona"/>
<xs:complexType name="personinfo">
...
</xs:complexType>

XML Schema. Elementos complejos V
Además, se puede describir un **elemento complejo** basándose en otro elemento complejo añadiendo más elementos:
*<xs:element name="empleado" **type="InfoPersonaT"/>***
<xs:complexType name="InfoPersona">
<xs:sequence>
<xs:element name="nombre" type="xs:string"/>

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

<xs:complexType name=" InfoPersonaT">
        <xs:complexContent>
            <xs:extension base=" InfoPersona">
                <xs:sequence>
                    <xs:element name="direccion" type="xs:string"/>
                    <xs:element name="ciudad" type="xs:string"/>
                    <xs:element name="pais" type="xs:string"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
</xs:complexType>
```

XML Schema. Elementos complejos vacíos

Definición en XML Schema:

```
<xs:element name="producto">
        <xs:complexType>
            <xs:attribute name="pid" type="xs:positiveInteger"/>
        </xs:complexType>
</xs:element>
<xs:element name="producto" type="TipoProd"/>

<xs:complexType name="TipoProd">
        <xs:attribute name="pid" type="xs:positiveInteger"/>
</xs:complexType>
```

XML Schema. Indicadores para elementos complejos

Permiten realizar el control de qué elementos van a ser utilizados en un documento XML.

Hay siete tipos de indicadores:

– Indicadores de orden:

- Sequence

– Indicadores de ocurrencia:

- maxOccurs
- minOccurs

XML Schema. Indicadores de orden

Indicador **<xs:sequence>**

– Deben aparecer los hijos en la secuencia descrita y solo una vez.

XML Schema. Indicadores de ocurrencia

Indicador **<maxOccurs>:** El número máximo de veces que se puede dar un elemento.

Indicador **<minOccurs>:** El número mínimo de veces que un elemento puede aparecer.

```
<xs:element name="person">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="nombre_completo" type="xs:string"/>
                <xs:element name="nombre_hijo" type="xs:string"
                                        maxOccurs="10" minOccurs="0"/>
            </xs:sequence>
        </xs:complexType>
</xs:element>
```

Por defecto, <minOccurs> = “1”