# XMLSCHEMA

XML Schema instance namespace

## O. TIPOS DE DATOS

## B) STRING

- **string** cadena (ID, IDREF, language, Name, NMTOKEN, etc)

## B) NÚMERICOS

- **byte** Un entero de 8 bits con signo
- **decimal** Un valor decimal
- **int** Un entero de 32 bits con signo
- **integer** Un valor entero
- **long** Un entero de 64 bits con signo
- **negativeInteger** Un entero contiene solo valores negativos (...,-2,-1).
- **nonNegativeInteger** Un entero contiene solamente valores no negativos (0,1,2,...)
- **nonPositiveInteger** Un entero que contiene valores no positivos (...,-2,-1,0).
- **positiveInteger** Un entero contiene valores positivos (1,2,...)
- **short** Un entero de 16 bit con signo
- **unsignedLong** Un entero de 64 bit sin signo
- **unsignedInt** Un entero de 32 bits sin signo
- **unsignedShort** Un entero de 16 bits sin signo
- **unsignedByte** Un entero de 8bits sin signo

## C) DATE

- **date** Define valor de fecha
- **dateTime** Define una fecha y hora
- **duration** Define un intervalo de tiempo. Formato: "PnYnMnDTnHnMnS"
    - P periodo (obligatorio)
    - nY número de años
    - nM número de meses
    - nD número de dias
    - T comienza la sección tiempo
    - nH número de horas
    - nM número de mínutos
    - nS número de segundos Se puede expresar en negativo.
- **gDay** Define dia (DD)
- **gMonth** Define parte de la fecha- mes (MM)
- **gMonthDay** Define parte de la fecha- mes y dia (MM-DD)
- **gYear** Define parte de la fecha- YYYY
- **gYearMonth** Define parte de la fecha- año y mes (YYYY-MM)
- **time** Define un valor de tiempo ### d) Tipos de datos misceláneos
- anyURI
- base64Binary (Base64-encoded binary data)
- boolean Valores true o false, 1 o 0.
- double
- float
- hexBinary (hexadecimal-encoded binary data)
- NOTATION
- QName > ## 1. DEFINICION TIPOS SIMPLES > Aquellos elementos que no tienen otros elementos. Ni atributos. > El elemento xs:element puede tener los siguientes atributos:

siguientes atributos:

- **name** Nombre del elemento
- **type** Tipo de elemento
- **minOccurs** Número mínimo de veces que puede aparecer
- **maxOccurs** Número máximo de veces que puede aparecer. El valor **unbounded** indica infinito.
- **default** Valor por defecto
- **fixed** Valor que se asigna al elemento y que no pue ser modificado

## 1.1 DEFINICIÓN DE UN ELEMENTO SIMPLE

```
<xs:element name="xxx" type="yyy" />
```

siendo **xxx**, el nombre del elemento y **yyy** el tipo de dato.

## 1.2 RESTRICCION:NÚMERICAS

> Aquellos elementos númericos donde podemos indicarle un rango o bien, para números reales número total de digitos y/o los correspondientes a la parte fraccionaria.
>
> - **minInclusive** Mayor o igual al valor
> - **maxInclusive** Menor o igual al valor
> - **minExclusive** Mayor que el valor
> - **maxExclusive** Menor que el valor
> - **totalDigits** Número total de dígitos
> - **fractionDigits** Número de dígitos de la parte fraccionaria

```
<xs:element name="nombre">
    <simpleType>
            <xs:restriction base="xs:integer">
                <xs:minInclusive value="0" />
                <xs:maxInclusive value="100" />
            </xs:restriction>
    </simpleType>
</xs:element>
```

## 1.3 RESTRICCIÓN: enumeration

```
 <xs:element name="siglas">
    <xs:simpleType>
        <xs:restriction base="xs:string">
             <xs:enumeration value="C" />
             <xs:enumeration value="JAVA" />
                 <xs:enumeration value="XML" />
             <xs:enumeration value="HTML" />
             <xs:enumeration value="PHP" />
             <xs:enumeration value="JAVASCRIPT" />
         </xs:restriction>
    </xs:simpleType>
</xs:element>
```

> O bien
>
> ```
>  <xs:element name="siglas">
>     <xs:simpleType>
>         <xs:restriction base="xs:string">
>              <xs:pattern value="C|JAVA|XML|HTML|PHP|JAVASCRIPT"/>
>          </xs:restriction>
>     </xs:simpleType>
> </xs:element>
> ```
>
> ## 1.4 RESTRICCIÓN: longitud de la cadena
>
> ```
> <xs:element name="nombreAsignatura" type="nombreAsignaturaTipo"/>
> <xs:simpleType name="nombreAsignaturaTipo">
>     <xs:restriction base="xs:string">
> ```

```
        <xs:maxLength value="100"/>
        <xs:minLength value="5" />
    </xs:restriction>
</xs:simpleType>
```

### 1.5 RESTRICCIÓN: Patrón de cadena

```
<xs:element name="codigoPostal" type="codigoPostalTipo" />
<xs:simpleType name="codigoPostalTipo">
    <xs:restriction base="xs:integer">
       <xs:pattern value="[0-9][0-9][0-9][0-9][0-9]"/>
    </xs:restriction>
</xs:simpleType>
```

### 1.6 RESTRICCIÓN: Acepta una cadena de un caracter en minúscula

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

### 1.7 RESTRICCION: Acepta una cadena de tres caracteres en mayúsculas

```
<xs:element name="iniciales" type="inicialesTipo"/>
<xs:simpleType name="inicialesTipo">
    <xs:restriction base="xs:string">
        <xs:pattern value="[A-Z][A-Z][A-Z]"/>
    </xs:restriction>
</xs:simpleType>
```

### 1.8 RESTRICCION: Acepta una cadena de tres caracteres en mayúsculas o minúsculas

```
<xs:element name="iniciales" type="inicialesTipo"/>
<xs:simpleType name="inicialesTipo">
    <xs:restriction base="xs:string">
        <xs:pattern value="[A-Za-z][A-Za-z][A-Za-z]"/>
    </xs:restriction>
</xs:simpleType>
```

### 1.9 RESTRICCION: Sobre una serie de valores

El elemento letras acepta **cero o más letras** minúsculas desde la 'a' a 'z'

```
<xs:element name="letras">
    <xs:simpleType>
        <xs:restriccion base="xs:string">
            <xs:pattern value="([a-z])*"/>
        </xs:restriction>
    </xs:simpleType>
</xs:element>
```

El elemento letras el valor acepta **uno o más pares de letras**, considerando que capa par consiste una leteas minúsculas seguido de una letra mayúsculas. Por elemplo esta cadena es aceptada: sToP

```
<xs:element name=""letras">
    <xs:simpleType>
            <xs:restriction base="xs:string">
        <xs:pattern value="([a-z][A-Z])+"/>
        </xs:restriction>
    </xs:simpleType>
</xs:element>
```

El elemento podemos indicar cuantos elementos acepta. Por ejemplo,

son 8 letras o dígitos, las letras mayúsculas o minúsculas o dígito de 0 a 9.

```
<xs:element name=""password">
    <xs:simpleType>
            <xs:restriction base="xs:string">
        <xs:pattern value="([a-zA-Z0-9]){8}"/>
        </xs:restriction>
    </xs:simpleType>
</xs:element>
```

### 1.10 RESTRICCION: Espacio entre caracteres. Valores preserve -> El procesador XML no eliminará ningún espacio en blanco.

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

### 1.11 RESTRICCION: Espacio entre caracteres. Valores replace ->El preprocesador XML replaza todos los caracteres espacio (nuevas lineas, tabuladores, espacios y retornos de carro) con espacio:

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

### 1.12 RESTRICCION: Espacio entre caracteres. Remover todos los espacios (los avances de línea, tabulaciones, espacios, retornos de carro, los espacios iniciales y finales se eliminan, y los espacios se reducen en un solo espacio)

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

### 1.13 RESTRICCION: Longitud de una cadena

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

### 1.13 RESTRICCION: Longitud entre un mínimo y un máximo

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

## 2. DECLARACION ATTRIBUTOS

Los atributos de un elemento son:

- **name** Nombre del atributo
- **type** Tipo de atributo
- **use** required | optional | prohibited
- **default** Valor por defecto
- **fixed** Valor fijo

```
<xs:attribute name="orderDate" type="xs:date"/>
```

# 3. DEFINICION DE TIPOS COMPLEJOS

## 3.1 TIPO COMPLEJO VACIO Y CON UN ATRIBUTO

Dado el elemento

```
<repetidor opcion="yes" />
```

XML Schema para el elemento repetidor

```
<xs:element name="repetidor" type="repetidorTipo" />
<xs:complexType name="repetidorTipo">
       <xs:attribute name="opcion" type="xs:string" />
  </xs:complexType>
```

## 3.2 COMPLEJO CON CONTENIDO SIMPLE

Declaración de un elemento complejo de contenido simple con atributos

Dado el elemento en XML:

```
<asignatura nota="10">Lenguaje de Marcas</asignatura>
```

XML Schema

```
<xs:element name="asignatura" type="asignaturaTipo"/>
<xs:complexType name="asignaturaTipo">
   <xs:simpleContent>
      <xs:extension base="xs:string" >
          <xs:attribute name="nota" type="tnota" />
      </xs:extension>
   </xs:simpleContent>
</xs:complexType>
```

## 3.3 COMPLEJO: SECUENCIA CON ATRIBUTOS

Declaración de un tipo complejo formado por una *secuencia de elementos*

Dado el elemento XML:

```
<ordenPago fecha="2018-02-08">
    <venderA>
            <calle>Calle Joaquin Sabina, 18</calle>
            <localidad codigo="49007">Benavente</localidad>
            <provincia>Zamora</provincia>
     </venderA>
     <factura>A99</factura>
     <comentario/>
</ordenPago>
```

DTD

```
<!ELEMENT OrdenPagoTipo(vendedorTo, facturaTo, comentario,
termino, fecha)>
```

XML Schema

```
<xs:element name="ordenPago" minOccurs="unbounded">
    <xs:complexType >
        <xs:sequence>
        <xs:element name="venderA">
```

```
        <xs:element name="VenderA" >
        <xs:complexType>
            <xs:sequence>
            <xs:element name="calle" type="xs:string"/>
            <xs:element name="localidad">
                <xs:complexType>
                    <xs:simpleContent>
                        <xs:extension base="xs:string">
                        <xs:attribute name="codigo"
type="xs:string" />
                        </xs:extension>
                    </xs:simpleContent>
                </xs:complexType>
            </xs:element>
            <xs:element name="provincia" type="xs:string" />
            </xs:sequence>
        </xs:complexType>
        </xs:element>
        <xs:element name="factura">
            <xs:simpleType >
            <xs:restriction base="xs:string">
                <xs:pattern value="[A-Z][0-9][0-9]"/>
            </xs:restriction>
            </xs:simpleType>
        </xs:element>
        <xs:element name="comentario" type="xs:string"
minOccurs="0"/>
        </xs:sequence>
        <xs:attribute name="fecha" type="xs:date"/>
    </xs:complexType>
</xs:element>
```

## 3.4 COMPLEJO: SELECCIÓN DE ELEMENTOS

Declaración de un tipo complejo formado por una *selección de elementos:* Dado el elemento XML:

```
   <formadepago>
       <contado>100</contado>
   </formadepago>
```

Dado el elemento DTD:

```
  <!ELEMENT formadepago
(contado|contrareembolso|visa|paypal|master)>
```

XML Schema

```
<xs:element name="formadepago">
    <xs:complexType>
        <xs:choice>
            <xs:element name="contado" type="xs:float" />
            <xs:element name="contrareembolso" type="xs:float" />
            <xs:element name="visa" type="xs:float" />
            <xs:element name="paypal" type="xs:float" />
            <xs:element name="master" type="xs:float" />
        </xs:choice>
      <xs:complexType>
</xs:element>
```

## 3.5 COMPLEJO: Opcionalidad de elemento (min 0 y max 1) y en cualquier orden

Declaración de un tipo complejo formado por elementos opcionales y sin importar el orden Dado el elemento XML:

```
   <lista40 fecha="2018-02-22">
       <cancion1>MOONLIGHTNING</cancion1>
       <cancion3>SUMMER SON</cancion3>
       <cancion4>PRIVATE DANCER</cancion4>
   </lista40>
```

</lista40>

Dado el elemento DTD:

```
  <!ELEMENT lista40 (cancion1?|cancion2?|cancion3?|cancion4?
|cancion5?)>
  <!ATTLIST lista40 fecha CDATA #REQUIRED>
```

XML Schema

```
<xs:element name="cancion1" type="xs:string"/>
<xs:element name="cancion2" type="xs:string"/>
<xs:element name="cancion3" type="xs:string"/>
<xs:element name="cancion4" type="xs:string"/>
<xs:element name="cancion5" type="xs:string"/>
<xs:element name="fecha" type="xs:date"/>
<xs:element name="lista40">
    <xs:complexType>
        <xs:all>
            <xs:element ref="cancion1"/>
            <xs:element ref="cancion2"/>
            <xs:element ref="cancion3"/>
            <xs:element ref="cancion4"/>
            <xs:element ref="cancion5"/>
        </xs:all>
        <xs:attribute ref="fecha"/>
      </xs:complexType>
</xs:element>
```

## 3.6 COMPLEJO: Grupo de elementos

XML Schema

```
<xs:group name="nombreGrupo">
    <xs:sequence>
        <xs:element name="elemento11" type="xs:string"/>
        <xs:element name="elemento12" type="xs:string"/>
        <xs:element name="elemento13" type="xs:string"/>
    </xs:sequence>
</xs:group>
<xs:element name="elemento1">
    <xs:complexType>
        <xs:group ref="nombreGrupo"/>
    </xs:complexType>
</xs:element>
```