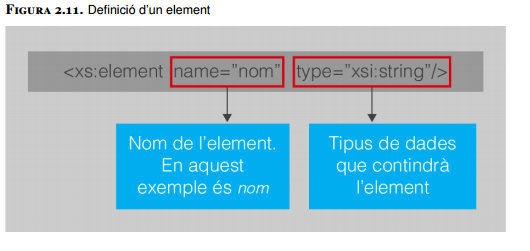
**Apuntes XSD**

* Se creó para sustituir DTD.
* Está escrito en XML y por lo tanto no hace falta aprenderse otro lenguaje nuevo de esquemas XML.
* Tiene su propio sistema de datos.
* Soporta espacio de nombres.
* Permite ser reutilizado y sigue el modelo de programación con herencia de objeto y sustitución de tipos.

1. **Asociar un esquema a un fichero XML**

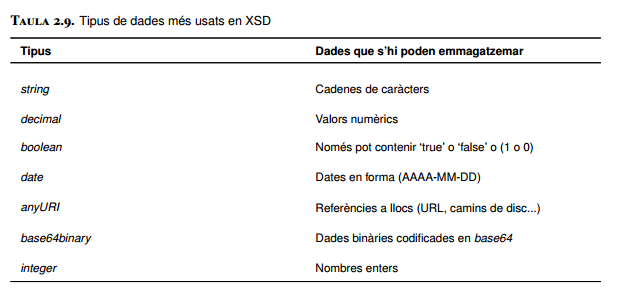
<realestate xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="realestate.xsd"> 🡪 Siempre va al principio y se utiliza para definir dónde está el fichero xsd.

1. **Definir un fichero de SCHEMA**

* XSD se divide en 2 grupos de datos:
  + Elementos de **TIPO SIMPLE:** Elementos sin atributos, sólo datos
  + Elementos de **TIPO COMPLEJO:** Elementos que pueden tener atributos, no tener contenido o contener elementos.

**ELEMENTOS CON CONTENIDO DE TIPO SIMPLE**

* **TIPO SIMPLE:** No tienen otros elementos ni atributos. El type=”xs: ” puede ser…



**XSD**

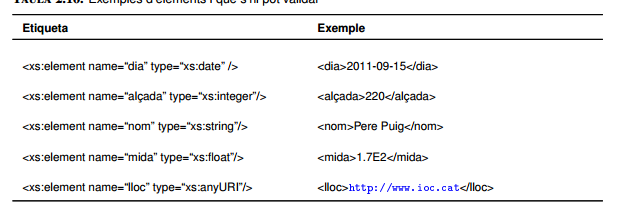
<xs:element name=”posicio” type=”xs:integer” /> | \*Se autocierra

</xs:schema>

**XML**

<posicio>Primer</posicio>

\*Solo conseguirá validar un elemento si su contenido es un nombre entero.



Si nos fijamos la etiqueta **dia** es del tipo **date**.

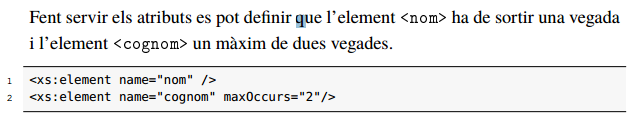
LETRAS 🡪 **XS:STRING**

NUMEROS 🡪 **XS:INTEGER**

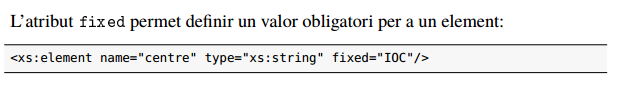
DECIMALES 🡪**XS:FLOAT**

URL 🡪 **XS:ANYURI**

* **minOccurs:** Permite definir **cuantas veces** saldrá este elemento como mínimo. Añadir un valor **‘0’** indica que este elemento **no** podrá salir.
* **maxOccurs:** Sirve para definir **cuantas veces** saldrá como máximo este elemento. **Unbounded** implica que **no** hay límite.

Saldrá como máximo 2 veces.

También se pueden dar valores a los elementos con atributos **FIXED, DEFAULT y NULLABLE.**



<centre>IOC</centre>

**DEFAULT 🡪 Deja un valor por defecto**

**LA ESTRUCTURA ES LA MISMA**

1 <xs:element name=”persona”> 🡪 Nombre del elemento/etiqueta

2 <xs:simpleType> 🡪 Tipo de XSD SIMPLE

3

4 </xs:simpleType>

5 </xs:element>

1 <xs:element name=”partidos”> 🡪 Nombre del elemento/etiqueta

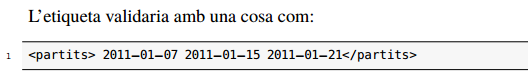
2 <xs:simpleType> 🡪 Tipo de XSD SIMPLE

3 <xs:lista itemType=”xs:date”/> 🡪 **lista**

4 </xs:simpleType>

5 </xs:element>

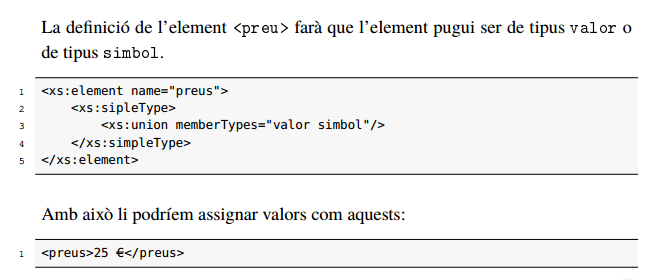
**Lista permitirá definir un elemento que puede contener listas de valores. LISTA DE DATOS.**



Los elementos **simpleType** también se pueden definir con un nombre fuera de los **elementos y** posteriormente utilizarlos como tipo de datos personal.

1. <xs:simpleType name=”dies”>
   1. <xs:lista itemType=”xs:date” />
2. </xs:simpleType>
3. <xs:element name=”partits” type=”dies” /> 🡪 fuera del elemento

**Union** permite que se puedan **mezclar tipos diferentes** en el contenido **de un elemento.**



**RESTRICTION** define restricciones para que se puedan aceptar algunos determinados valores.

<xs:simpleType name=”any\_naixement”> 🡪

<xs:restriction base=”xs:integer”> 🡪 Tipo entero del tipo RESTRICTION

<xs:maxInclusive value=”2011” /> 🡪 Valor máximo que puede llegar

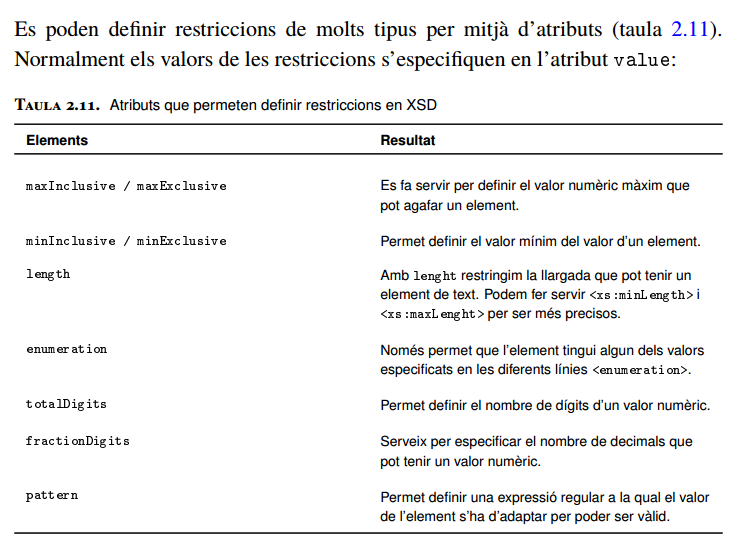
<xs:minInclusive value=”1850” /> 🡪 Valor mínimo

<xs:restriction>

</xs.simpleType>

<xs:element name=”naixement” type=”any\_naixement” />

1. Primero definir el elemento fuera y después con una simpleType o complexType crear sub elementos.

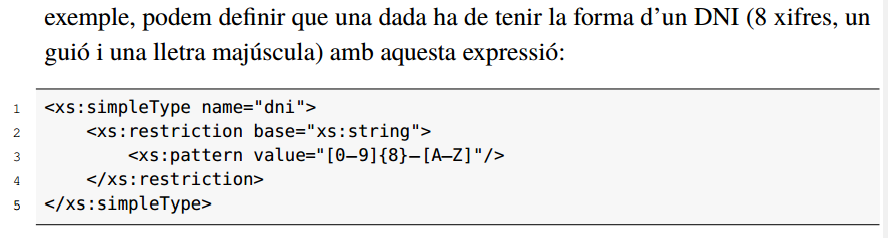


**Enumeration**

**<xs:enumeration value=”A” />**

**ATRIBUTO PATTERN 🡪** Restricciones de expresiones regulares con condiciones.

|  |  |
| --- | --- |
| **Símbolo** | **Equivalencia** |
| . | Cualquier carácter |
| \d | Cualqueir dígito |
| \D | Cualquier carácter no digito |
| \s | Caracteres no imprimibles |
| \S | Cualquier carácter imprimible |
| x\* | Saldrá 0 o más veces |
| x+ | Saldrá 1 o más veces |
| x? | Puede salir o no |
| [abc] | Puede ser ‘a’ ‘b’ o ‘c’ |
| [0-9] | Del 0 al 9, extremos incluidos |
| x{5} | Habrá 5 veces el valor |
| x{5,} | Saldrá 5 o más veces |
| x{5,8} | Habrá entre 5 y 8 veces |

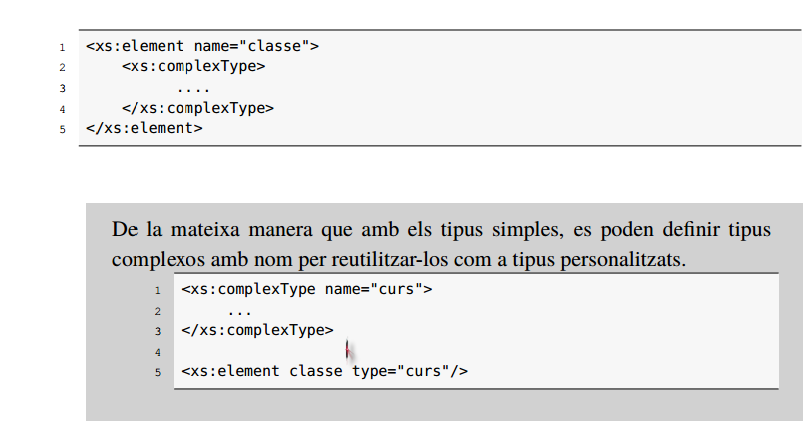


**ELEMENTOS CON CONTENIDO DE TIPO COMPLEJO**

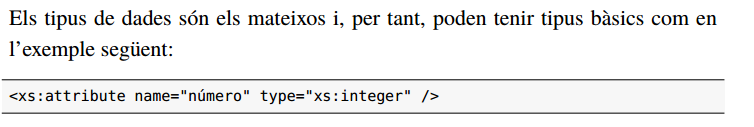
Elementos que contienen otros elementos o no tienen contenido.

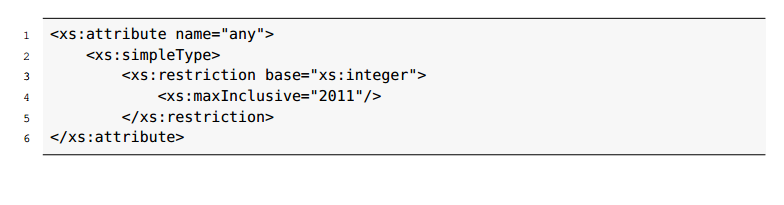
1. Tienen más datos. Son como los del tipo simple, pero con **atributos.**
2. Contenido con más elementos.
3. Los elementos vacíos.
4. Elementos con contenido mezclado.

**<xs:complexType> 🡪 Para definir un XSD del tipo complejo**



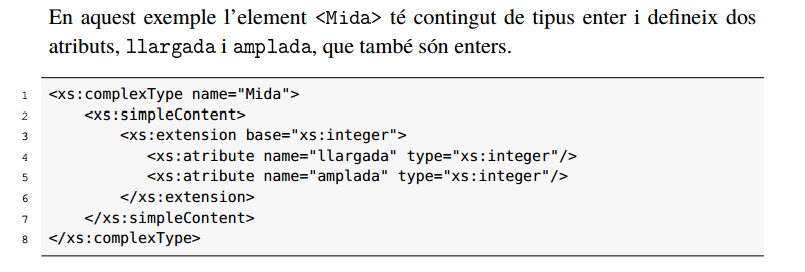
**Pueden utilizar atributos:**



<attribute> tiene una serie de caracteristicas:

|  |  |
| --- | --- |
| **Atributo** | **Uso** |
| Use | Obligatorio (required) | Ocional (optional) |
| Default | Valor por defecto |
| Fixed | Valor obligatorio |
| Form | Permite definir si el atributo puede ir con alias en el espacio de nombres (qualifies) o no (unqualified) |

**‘simpleContent’**

Va en complexType y será un simpleContent 🡪 Permite definir restricciones o extensiones de elementos. 

**EL ORDEN ES IMPORTANTE**

