

Les schémas XML

- Un Schéma XML (XML schema Definition XSD) est un langage basé sur XML qui permet la définition de schémas (structure + type de données) des documents XML et facilite la communication entre applications.

- **La différence entre les schémas XML et les DTDs**

DTD	XML schema
N'est pas une syntaxe XML	Basé sur la syntaxe XML
Difficile à étendre	Facilement extensible
Données textuelles non typées	Supporte les types de données
Ne permet pas de spécifier exactement le nombre d'occurrences d'un élément	Permet de spécifier exactement le nombre d'occurrences d'un élément
Ne supporte pas les espaces de noms	Supporte les espaces de noms

Exemple

- **Un simple document XML**

```
<?xml version="1.0"?>
<note>
  <to>Ali</to>
  <from>Seddik</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

- **Un fichier DTD**

```
<!ELEMENT note (to, from, heading, body)>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
```

Un schéma XML (Exemple)

```
<?xml version="1.0"?>
<xs:schema xmlns:xs=http://www.w3.org/2001/XMLSchema
elementFormDefault="qualified">
  <xs:element name="note">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="to" type="xs:string"/>
        <xs:element name="from" type="xs:string"/>
        <xs:element name="heading" type="xs:string"/>
        <xs:element name="body" type="xs:string"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Le référencement d'un schéma dans un document XML (Exemple)

- **Une référence à une DTD**

```
<?xml version="1.0"?>
<!DOCTYPE note SYSTEM "note.dtd">
<note>
  <to>Ali</to>
  <from>Seddik</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

- **Une référence à un schéma XML**

```
<?xml version="1.0"?>
<note xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
  xsi:schemaLocation="note.xsd">
  <to>Ali</to>
  <from>Seddik</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

Structure d'un XML Schema

```
<?xml version="1.0"?>
```

```
<xs:schema xmlns:xs=http://www.w3.org/2001/XMLSchema  
elementFormDefault="qualified">
```

```
...
```

```
...
```

```
</xs:schema>
```

Les éléments simples

- **Qu'est ce qu'un élément simple :**

Un élément simple est un élément XML qui peut contenir du texte seulement. Il ne peut pas contenir d'autres éléments ou d'attributs.

- **Définition d'un élément simple:**

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

- **Les types les plus communs sont:**

xs: string

xs: decimal

xs: int

xs: boolean

xs: date

xs: time

Les éléments simples (exemple)

- Voici quelques éléments XML:

```
<lastname>Maamri</lastname>  
<age>36</age>  
<dateborn>2002-03-27</dateborn>
```

- Et voici les définitions correspondantes (élément simple):

```
<xs:element name="lastname" type="xs:string"/>  
<xs:element name="age" type="xs:integer"/>  
<xs:element name="dateborn" type="xs:date"/>
```

- **Valeur par défaut/fixe d'un élément simple**

```
<xs:element name="color" type="xs:string" default="red"/>  
<xs:element name="company" type="xs:string" fixed="Microsoft"/>
```

Les attributs

- Tous les attributs sont déclarés comme des types simples. Si un élément a des attributs, il est considéré comme étant de type complexe. Mais l'attribut lui-même est toujours déclaré comme un type simple.

- La syntaxe pour définir un attribut est:

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

- **Exemple**

Voici un élément XML avec un attribut:

```
<lastname lang="EN">Maamri</lastname>
```

Et voici la définition de l'attribut correspondant:

```
<xs:attribute name="lang" type="xs:string"/>
```

- **Valeur par défaut/fixe d'un attribut**

```
<xs:attribute name="genre" type="xs:string" default="female"/>
```

```
<xs:attribute name="lang" type="xs:string" fixed="EN"/>
```


Les restrictions

- **Utilité :**

Les restrictions sont utilisées pour définir des valeurs acceptables pour les éléments ou les attributs XML.

- **Types de restrictions**

1. **Restrictions sur les valeurs :**

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

Les restrictions

2. Restrictions sur un ensemble de valeurs

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

3. Restrictions sur une série de valeurs

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

Les restrictions

4. Expressions possibles

- `<xs:pattern value="[A-Z][A-Z][A-Z]"/>`
- `<xs:pattern value="[a-zA-Z][a-zA-Z][a-zA-Z]"/>`
- `<xs:pattern value="[xyz]"/>`
- `<xs:pattern value="[0-9][0-9][0-9][0-9][0-9]"/>`
- `<xs:pattern value="([a-z])*"/>`
- `<xs:pattern value="([a-zA-Z])+"/>`
- `<xs:pattern value="m|f"/>`

Les restrictions

5. Restriction sur le longuer d'un élément

a.

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

b.

```
<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>
```

Les éléments complexes

- Un élément complexe est un élément XML qui contient d'autres éléments et / ou des attributs. Il existe quatre types d'éléments complexes:
 - les éléments vides
 - éléments qui contiennent uniquement des éléments autres
 - éléments qui ne contiennent que du texte
 - éléments qui contiennent les deux autres éléments et le texte
- **Note:** Chacun de ces éléments peuvent contenir des attributs.
- **Exemples d'éléments complexes:**

```
<product pid="1345"/>
```

```
<employee>  
  <firstname>John</firstname>  
  <lastname>Smith</lastname>  
</employee>
```

Les éléments vides (Exemple)

- **Code XML :**

```
<product prodid="1345" />
```

Définition 1:

```
<xs:element name="product">  
  <xs:complexType>  
    <xs:attribute name="prodid" type="xs:positiveInteger"/>  
  </xs:complexType>  
</xs:element>
```

Définition 2:

- ```
<xs:complexType name="prodtype">
 <xs:attribute name="prodid" type="xs:positiveInteger"/>
</xs:complexType>
```
- ```
<xs:element name="product" type="prodtype"/>
```

Les éléments complexes (Exemples)

- `<food type="dessert">Ice cream</food>`
- `<employee>`
 - This employee
 - `<firstname>Ali</firstname>`
 - `<lastname>Maamri</lastname>`
 - since
 - `<date>2002-04-23</date>`.
 - He is become a manager`</employee>`

Comment définir un élément complexe

- Soit l'élément complexe suivant:

```
<employee>  
  <firstname>Ali</firstname>  
  <lastname>Maamri</lastname>  
</employee>
```

- Cet élément peut être défini de deux façons:

1. Sans définition de type:

```
<xs:element name="employee">  
  <xs:complexType>  
    <xs:sequence>  
      <xs:element name="firstname" type="xs:string"/>  
      <xs:element name="lastname" type="xs:string"/>  
    </xs:sequence>  
  </xs:complexType>  
</xs:element>
```

- Si vous utilisez la méthode décrite ci-dessus, seul l'élément «employé» peut utiliser le type complexe spécifié.

Comment définir un élément complexe

2. Avec définition de type:

```
<xs:complexType name="personinfo">  
  <xs:sequence>  
    <xs:element name="firstname" type="xs:string"/>  
    <xs:element name="lastname" type="xs:string"/>  
  </xs:sequence>  
</xs:complexType>
```

- Si vous utilisez la méthode décrite ci-dessus, plusieurs éléments peuvent se référer:

```
<xs:element name="employee" type="personinfo"/>  
<xs:element name="student" type="personinfo"/>  
<xs:element name="member" type="personinfo"/>
```

Extension d'un élément complexe

- Vous pouvez également construire un élément complexe en utilisant un élément complexe existant et ajouter quelques éléments, comme ceci:

```
<xs:complexType name="personinfo">
  <xs:sequence>
    <xs:element name="firstname" type="xs:string"/>
    <xs:element name="lastname" type="xs:string"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="fullpersoninfo">
  <xs:complexContent>
    <xs:extension base="personinfo">
      <xs:sequence>
        <xs:element name="address" type="xs:string"/>
        <xs:element name="city" type="xs:string"/>
        <xs:element name="country" type="xs:string"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<xs:element name="employee" type="fullpersoninfo"/>
```

Les éléments de types mixtes

- Un élément complexe de type mixte peut contenir des attributs, des éléments, et de texte.
- **Exemple:**

Code XML

```
<letter>  
  Dear Mr.<name>John Smith</name>.  
  Your order <orderid>1032</orderid>  
  will be shipped on <shipdate>2001-07-13</shipdate>.  
</letter>
```

Les éléments de types mixtes

- **Définition 1:**

```
<xs:element name="letter">  
  <xs:complexType mixed="true">  
    <xs:sequence>  
      <xs:element name="name" type="xs:string"/>  
      <xs:element name="orderid" type="xs:positiveInteger"/>  
      <xs:element name="shipdate" type="xs:date"/>  
    </xs:sequence>  
  </xs:complexType>  
</xs:element>
```

- **Définition 2:**

```
<xs:complexType name="lettertype" mixed="true">  
  <xs:sequence>  
    <xs:element name="name" type="xs:string"/>  
    <xs:element name="orderid" type="xs:positiveInteger"/>  
    <xs:element name="shipdate" type="xs:date"/>  
  </xs:sequence>  
</xs:complexType>
```

- <xs:element name="letter" type="lettertype"/>

Les indicateurs d'ordre

L'indicateur « all »

Cet indicateur spécifie que les éléments enfants peuvent apparaître dans n'importe quel ordre, et que chaque élément enfant ne doit apparaître qu'une seule fois :

```
<xs:element name="person">
  <xs:complexType>
    <xs:all>
      <xs:element name="firstname" type="xs:string"/>
      <xs:element name="lastname" type="xs:string"/>
    </xs:all>
  </xs:complexType>
</xs:element>
```

Remarque : lorsque vous utilisez l'indicateur `<all>` , vous pouvez définir l'indicateur `<minOccurs>` sur 0 ou 1 et l'indicateur `<maxOccurs>` ne peut être réglé que sur 1.

Les indicateurs d'ordre

L'indicateur « choice »

L'indicateur <choix> spécifie qu'un élément enfant ou un autre peut se produire :

```
<xs:element name="person">  
  <xs:complexType>  
    <xs:choice>  
      <xs:element name="employee" type="employee"/>  
      <xs:element name="member" type="member"/>  
    </xs:choice>  
  </xs:complexType>  
</xs:element>
```

Les indicateurs d'ordre

Sequence Indicator

L'indicateur `<sequence>` spécifie que les éléments fils doivent apparaître dans un ordre spécifique :

```
<xs:element name="person">  
  <xs:complexType>  
    <xs:sequence>  
      <xs:element name="firstname" type="xs:string"/>  
      <xs:element name="lastname" type="xs:string"/>  
    </xs:sequence>  
  </xs:complexType>  
</xs:element>
```

Les indicateurs d'occurrence

Les indicateurs d'occurrence servent à définir la fréquence à laquelle un élément peut se produire.

Remarque :

Pour tous les indicateurs "Ordre" et "Groupe" (any, all, choice, sequence, group name, et group reference) la valeur par défaut pour maxOccurs et minOccurs est 1.

```
<xs:element name="child_name" type="xs:string"  
  minOccurs="0" maxOccurs="10"/>
```


Les indicateurs de group

Element Groups

```
<xs:group name="persongroup">  
  <xs:sequence>  
    <xs:element name="firstname" type="xs:string"/>  
    <xs:element name="lastname" type="xs:string"/>  
    <xs:element name="birthday" type="xs:date"/>  
  </xs:sequence>  
</xs:group>
```

```
<xs:complexType name="personinfo">  
  <xs:sequence>  
    <xs:group ref="persongroup"/>  
    <xs:element name="country" type="xs:string"/>  
  </xs:sequence>  
</xs:complexType>
```

```
<xs:element name="person" type="personinfo"/>
```

Les indicateurs de group

Attribute Groups

```
<xs:attributeGroup name="personattrgroup">  
  <xs:attribute name="firstname" type="xs:string"/>  
  <xs:attribute name="lastname" type="xs:string"/>  
  <xs:attribute name="birthday" type="xs:date"/>  
</xs:attributeGroup>
```

```
<xs:element name="person">  
  <xs:complexType>  
    <xs:attributeGroup ref="personattrgroup"/>  
  </xs:complexType>  
</xs:element>
```

Les indicateurs de group

Example:

Code XML:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<persons xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="family.xsd">
  <person>
    <full_name>Hege Refsnes</full_name>
    <child_name>Cecilie</child_name>
  </person>
  <person>
    <full_name>Tove Refsnes</full_name>
    <child_name>Hege</child_name>
    <child_name>Stale</child_name>
    <child_name>Jim</child_name>
    <child_name>Borge</child_name>
  </person>
  <person>
    <full_name>Stale Refsnes</full_name>
  </person>
</persons>
```

Les indicateurs de group

Les indicateurs de group:

Schéma XML:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified">

<xs:element name="persons">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="person" maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="full_name" type="xs:string"/>
            <xs:element name="child_name" type="xs:string"
minOccurs="0" maxOccurs="5"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>

</xs:schema>
```

L'élément <any>

Utilité:

L'élément <any> nous permet d'étendre le document XML avec des éléments non spécifiés par le schéma.

Scéma family.sxd

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xs:schema xmlns:xs=http://www.w3.org/2001/XMLSchema
elementFormDefault="qualified">
  <xs:element name="person">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="firstname" type="xs:string"/>
        <xs:element name="lastname" type="xs:string"/>
        <xs:any minOccurs="0"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
```

L'élément <any>

Schéma children.sxd

```
<?xml version="1.0" encoding="ISO-8859-1"?>  
<xs:schema xmlns:xs=http://www.w3.org/2001/XMLSchema  
elementFormDefault="qualified">
```

```
<xs:element name="children">  
  <xs:complexType>  
    <xs:sequence>  
      <xs:element name="childname" type="xs:string"  
        maxOccurs="unbounded"/>  
    </xs:sequence>  
  </xs:complexType>  
</xs:element>  
  
</xs:schema>
```

L'élément <any>

Le fichier family.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

```
<persons xmlns="http://www.microsoft.com"  
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
xsi:SchemaLocation="family.xsd children.xsd">
```

```
<person>  
  <firstname>Hege</firstname>  
  <lastname>Refsnes</lastname>  
  <children>  
    <childname>Cecilie</childname>  
  </children>  
</person>
```

```
<person>  
  <firstname>Stale</firstname>  
  <lastname>Refsnes</lastname>  
</person>
```

```
</persons>
```

L'élément <anyAttribute>

Utilité:

L'élément <anyAttribute> permet d'étendre le document XML avec attributs non spécifiés par le schéma.

Scéma family.sxd

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xs:schema xmlns:xs=http://www.w3.org/2001/XMLSchema
elementFormDefault="qualified">
  <xs:element name="person">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="firstname" type="xs:string"/>
        <xs:element name="lastname" type="xs:string"/>
      </xs:sequence>
      <xs:anyAttribute/>
    </xs:complexType>
  </xs:element>
```


L'élément <anyAttribute>

Scéma attribute.sxd

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xs:schema xmlns:xs=http://www.w3.org/2001/XMLSchema
elementFormDefault="qualified">

  <xs:attribute name="gender">
    <xs:simpleType>

      <xs:restriction base="xs:string">
        <xs:pattern value="male|female"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>

</xs:schema>
```

L'élément <anyAttribute>

Le fichier family.xml

Le fichier XML ci-dessous (appelé "Myfamily.xml"), utilise des composants de deux schémas différents ; "family.xsd" et "attribute.xsd" :

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

```
<persons xmlns="http://www.microsoft.com"  
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
xsi:SchemaLocation="family.xsd attribute.xsd">
```

```
<person gender="female">  
  <firstname>Hege</firstname>  
  <lastname>Refsnes</lastname>  
</person>
```

```
<person gender="male">  
  <firstname>Stale</firstname>  
  <lastname>Refsnes</lastname>  
</person>
```

```
</persons>
```

Les schémas XML

11. Exercice:

Enoncé:

Donner un schéma xml pour le fichier " shiporder.xml " suivant:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<shiporder orderid="889923" >
  <orderperson>John Smith</orderperson>
  <shipto>
    <name>Ola Nordmann</name>
    <address>Langgt 23</address>
    <city>4000 Stavanger</city>
    <country>Norway</country>
  </shipto>
  <item>
    <title>Empire Burlesque</title>
    <note>Special Edition</note>
    <quantity>1</quantity>
    <price>10.90</price>
  </item>
  <item>
    <title>Hide your heart</title>
    <quantity>1</quantity>
    <price>9.90</price>
  </item>
</shiporder>
```

Les schémas XML

11. Exercice:

Solution 1:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="shiporder">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="orderperson" type="xs:string"/>
        <xs:element name="shipto">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="name" type="xs:string"/>
              <xs:element name="address" type="xs:string"/>
              <xs:element name="city" type="xs:string"/>
              <xs:element name="country" type="xs:string"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Les schémas XML

11. Exercice:

Solution 1:

```
<xs:element name="item" maxOccurs="unbounded">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="title" type="xs:string"/>
      <xs:element name="note" type="xs:string" minOccurs="0"/>
      <xs:element name="quantity" type="xs:positiveInteger"/>
      <xs:element name="price" type="xs:decimal"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
</xs:sequence>
<xs:attribute name="orderid" type="xs:string" use="required"/>
</xs:complexType>
</xs:element>

</xs:schema>
```

Les schémas XML

11. Exercice:

Solution 2:

The previous design method is very simple, but can be difficult to read and maintain when documents are complex.

The next design method is based on defining all elements and attributes first, and then referring to them using the ref attribute.

Here is the new design of the schema file ("shiporder.xsd"):

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <!-- definition of simple elements -->
  <xs:element name="orderperson" type="xs:string"/>
  <xs:element name="name" type="xs:string"/>
  <xs:element name="address" type="xs:string"/>
  <xs:element name="city" type="xs:string"/>
  <xs:element name="country" type="xs:string"/>
  <xs:element name="title" type="xs:string"/>
  <xs:element name="note" type="xs:string"/>
  <xs:element name="quantity" type="xs:positiveInteger"/>
  <xs:element name="price" type="xs:decimal"/>

  <!-- definition of attributes -->
  <xs:attribute name="orderid" type="xs:string"/>
```

Les schémas XML

11. Exercice:

Solution 2:

```
<!-- definition of complex elements -->
<xs:element name="shipto">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="name"/>
      <xs:element ref="address"/>
      <xs:element ref="city"/>
      <xs:element ref="country"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

<xs:element name="item">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="title"/>
      <xs:element ref="note" minOccurs="0"/>
      <xs:element ref="quantity"/>
      <xs:element ref="price"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Les schémas XML

11. Exercice:

Solution 2:

```
<xs:element name="shiporder">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="orderperson"/>
      <xs:element ref="shipto"/>
      <xs:element ref="item" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute ref="orderid" use="required"/>
  </xs:complexType>
</xs:element>

</xs:schema>
```