

# Schéma XML

- Groupe de substitution
- Dérivation de types complexe
- Contrainte d'unicité
- Clefs
- Inclusion de schémas
- Redéfinition de types d'un schéma externe
- Annotations

## Références

<http://www.w3.org/TR/xmlschema-1/> (Structures)

<http://www.w3.org/TR/xmlschema-2/> (Datatypes)

13/11/2013

Mario Jo Bellosta

# Group substitution

Permet de définir un groupe d'éléments substituables

- Deux parties
  - Élément substituable **introduit par substitutionGroup**
  - Élément qui le remplace **introduit par name**

```
<xsd:element name= "elementSubstituable" type="TypeElement"/>  
<xsd:element name="E" substitutionGroup="elementSubstituable" />
```

# Exemple: schema transportation.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">

<xsd:element name="transportation" type="transportationType"/>

<xsd:complexType name="transportationType">
  <xsd:sequence>
    <xsd:element ref="subway"/>
    <xsd:element name="city" type="xsd:NMTOKENS" minOccurs="1" />
    <xsd:element ref="description"
      minOccurs="0" maxOccurs="unbounded" />
  </xsd:sequence>
</xsd:complexType>

<xsd:element name="subway" type="xsd:normalizedString"/>

<xsd:element name="description" type="xsd:normalizedString"/>
</xsd:schema>
```

## Exemple: instance picadilly.xml

```
<transportation xmlns:xsi="http://www.w3.org/2001/XMLSchema-  
instance"
```

```
xsi:noNamespaceSchemaLocation="transportation.xsd">
```

```
<subway>Picadilly circus</subway>
```

```
<city>London</city>
```

```
<description>Located in travelcard zone 1</description>
```

```
<description>
```

```
    on the Picadilly Line between Green Park and Leicester Square
```

```
</description>
```

```
<description>
```

```
    on the Bakerloo line between Charing Cross and Oxford Circus
```

```
</description>
```

```
</transportation>
```

[http://en.wikipedia.org/wiki/Piccadilly\\_Circus\\_tube\\_station](http://en.wikipedia.org/wiki/Piccadilly_Circus_tube_station)

# Exemple: instance covent.xml

```
<transportation xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="transportation.xsd">
```

```
<subway>Covent garden</subway>
```

```
<city>London</city>
```

```
</transportation>
```



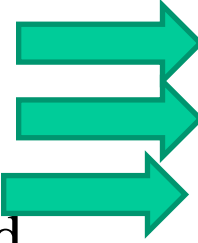
# Métro français

transport

metro

ville

Ajout dans transportation.xsd



transportation

subway

city

```
<xsd:element name="transport" substitutionGroup="transportation" />
```

```
<xsd:element name="metro" substitutionGroup="subway" />
```

```
<xsd:element name="ville" substitutionGroup="city" /> ERREUR
```

Instance bonneNouvelle.xml

```
<transport xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="transportation.xsd">
```

```
  <metro>bonne nouvelle</metro>
```

```
  <city>Paris</city>
```

```
  <description>
```

la station est sur les lignes 8 et 9 en limite du 2e,9e et 10e arrondissements de Paris

```
  </description>
```

```
</transport>
```

# Mixte autorisé

transport



transportation  
subway

Instance mixteCovent1.xml

```
<transport xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="transportation.xsd">
  <subway>covent garden</subway>
  <city>London</city>
</transport>
```

metro



transportation  
subway

Instance mixteCovent2.xml

```
<transportation xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="transportation.xsd">
  <metro>covent garden</metro>
  <city>London</city>
</transportation >
```

# Blocage de la substitution

Un élément interdit la substitution avec un autre élément en ajoutant l'attribut *block*

```
<xsd:element name="name_el" type="type_el" block="substitution"/>
```

## Exemple

```
<xsd:element name="name" type="xsd:string" block="substitution" />  
<xsd:element name="nom" substitutionGroup="name"/> ERREUR!!!!
```



# Spécification du type

Le type de chaque élément dans un groupe de substitution doit être le même, ou dérivé du type à substituer.

```
<xsd:element name="A" type="TypeA"/>
```

```
<xsd:element name="B" substitutionGroup="A" type="TypeB"/>
```



TypeB doit être soit TypeA  
soit un dérivé de TypeA

metroType dérivée de normalizeString

le nom d'une station commence par une majuscule et  
peut être composé de 1 à 4 mots

```
<xsd:element name="metro" substitutionGroup="subway"  
type="metroType"/>
```

```
<xsd:simpleType name=" metroType">
```

```
<xsd:restriction base="xsd:normalizedString">
```

```
<xsd:pattern value="\p{Lu}\p{L}+(\s(\p{L}))+){0,4}"/>
```

```
<xsd:minLength value="2"/>
```

```
</xsd:restriction>
```

```
</xsd:simpleType>
```

# Instance bonneNouvelle.xml

XML validation started.

## Checking

file:/C:/Users/mj/Desktop/WEB\_2012/XML%20schema/ProjetEmployeKey/BonneNouvelle.xml...

## Referenced entity at

"file:/C:/Users/mj/Desktop/WEB\_2012/XML%20schema/ProjetEmployeKey/transportation.xsd".

cvc-pattern-valid : La valeur 'bonne nouvelle' n'est pas un facet valide par rapport au modèle

'\p{Lu}\p{L}+(\s(\p{L}))+\{0,4\}' pour le type 'nomType'. [5]

cvc-type.3.1.3 : La valeur 'bonne nouvelle' de l'élément 'metro' n'est pas valide. [5]

XML validation finished.

```
<transport xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="transportation.xsd">
  <metro>bonne nouvelle</metro>
  <city>Paris</city>
  <description>la station est sur les lignes 8 et 9 en limite du 2e,9e et
10e arrondissements de Paris</description>
</transport>
```

# Instance bonneNouvelle2.xml

XML validation started.

Checking file:/C:/Users/mj/Desktop/WEB\_2012/XML%20schema/ProjetEmployeKey/BonneNouvelle2.xml...

Referenced entity at "file:/C:/Users/mj/Desktop/WEB\_2012/XML%20schema/ProjetEmployeKey/transportation.xsd".

XML validation finished.

```
<transport xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="transportation.xsd">
  <metro>Bonne nouvelle</metro>
  <city>Paris</city>
  <description>
    la station est sur les lignes 8 et 9 en limite du 2e,9e et 10e
    arrondissements de Paris
  </description>
</transport>
```

# Dérivation de type complexe

- Par extension
  - Ajout d'éléments
  - Ajout d'attributs
- Par restriction
  - Suppression d'éléments
  - Restriction du domaines de valeurs d'un élément
  - Diminution d'occurrence d'éléments

# Extension d'un type complexe

```
<xsd:complexType name="SubType">  
  <xsd:complexContent>  
    <xsd:extension base="SuperType" >  
      <xsd:(sequence|choice|all)>  
        <!-- éléments ajoutés dans ce sous type -->  
      </xsd:(sequence|choice|all)>  
      <!-- attributs ajoutés dans ce sous type -->  
    </xsd:extension>  
  </xsd:complexContent>  
</xsd:complexType>
```

# Transport peut avoir un élément ligne et attribut monuments

```
<xsd:element name= "transport" substitutionGroup="transportation" type="transportType"/>
  <xsd:complexType name= "transportType">
    <xsd:complexContent>
      <xsd:extension base="transportationType">
        <xsd:sequence >
          <xsd:element name="ligne" type="ligneType" minOccurs="1" maxOccurs="unbounded"/>
        </xsd:sequence>
        <xsd:attribute name= "monuments" type="xsd:NMTOKENS" use="required"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:simpleType name="ligneType">
    <xsd:restriction base="xsd:normalizedString">
      <xsd:pattern value="\p{N}(\s(BIS|bis))?" />
      <xsd:pattern value="\p{L}+" />
      <xsd:pattern value="RER\s\p{L}" />
    </xsd:restriction>
  </xsd:simpleType>
```

# Instance bonneNouvelle.xml

```
<transport xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="transportation.xsd"
  monuments="Grand Rex. Theatre du Gymnase. Max Linder."> >
  <metro>Bonne nouvelle</metro>
  <city>Paris</city>
  <description>
    la station est sur les lignes 8 et 9 en limite du 2e,9e et 10e
    arrondissements de Paris
  </description>
  <ligne>9</ligne>
  <ligne>8</ligne>
</transport>
```



# Dérivation par restriction

## Principe

1. Doit répéter tous les composants (element,group) du type de base inclus dans le type dérivé
2. La déclaration d'attributs est directement héritée et ne doit pas être répétée

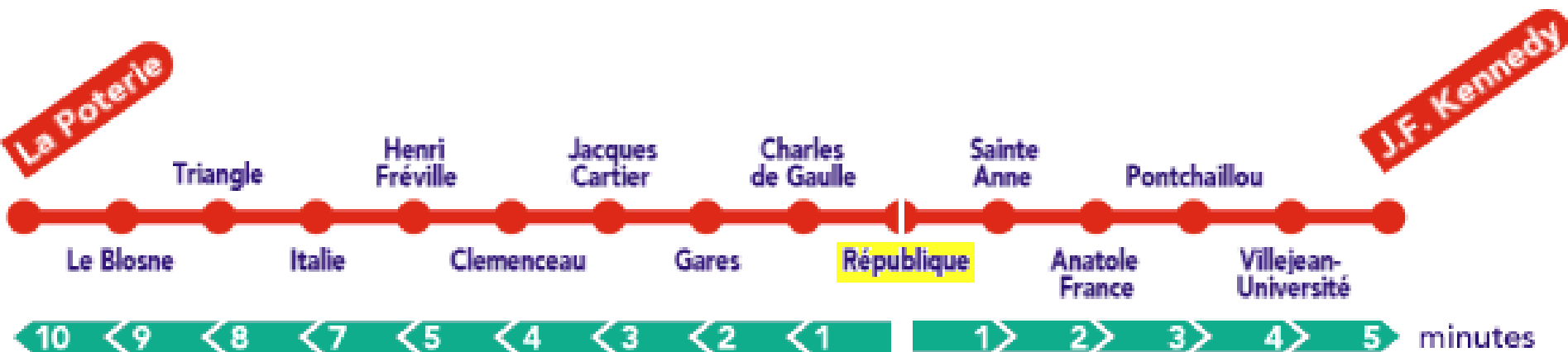
```
<xsd:complexType name="SubType">  
  <xsd:complexContent>  
    <xsd:restriction base="SuperType" >  
      <xsd:sequence>  
        <!-- éléments qui restent dans ce sous type -->  
      </xsd:sequence>  
    </xsd:extension>  
  </xsd:complexContent>  
</xsd:complexType>
```

# Transport avec le nom du métro

```
<xsd:element name="simpleTransport"
  substitutionGroup="transportation" type="SimpleTransportType"/>
<xsd:complexType name="SimpleTransportType">
  <xsd:complexContent>
    <xsd:restriction base="transportationType">
      <xsd:sequence>
        <xsd:element ref="metro"/>
        <xsd:element name="city" type="xsd:NMTOKENS" minOccurs="1" />
      </xsd:sequence>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>
```

# Instance: rennes.xml

```
<simpleTransport xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="transportation.xsd">
  <metro>Republique</metro>
  <city>Rennes</city>
</simpleTransport>
```



# Dérivation par restriction

## Contrainte

### Attention

Un élément du super type dont l'occurrence minimal est mise à 1 est **obligatoirement** mis dans les types restricts

```
<xsd:complexType name="SubType">
  <xsd:complexContent>
    <xsd:restriction base="SuperType" >
      <xsd:sequence>
        <!-- éléments qui restent dans ce sous type -->
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

# Transport avec le nom du métro

## ERREUR

XML validation started.

C:/Users/mj/Desktop/WEB\_2012/XML schema/ProjetEmployeKey/transport.xsd:66,0

ERROR: cos-particle-restrict.2 : Restriction de particule interdite : 'choice:all,sequence,elt'

C:/Users/mj/Desktop/WEB\_2012/XML schema/ProjetEmployeKey/transport.xsd:66,0

ERROR: derivation-ok-restriction.5.4.2 : Erreur dans le type 'SimpleTransportType'. La particule du type n'est pas une restriction valide de la particule du type de base

2 Error(s), 0 Warning(s).

XML validation finished.<xsd:element name="simpleTransport" substitutionGroup="transportation" type="SimpleTransportType"/>

```
<xsd:complexType name="SimpleTransportType">
  <xsd:complexContent>
    <xsd:restriction base="transportationType">
      <xsd:sequence>
        <xsd:element ref="metro"/>
      </xsd:sequence>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>
```

# BookType et MagazineType Dérivent de PublicationType

PublicationType  
(Title, Author, Date)

BookType  
(Title, Author, Date, ISBN, Publisher)

MagazineType  
(Title, Date)

```
<xsd:complexType name="PublicationType">  
  <xsd:sequence>  
    <xsd:element name="Title" type="xsd:string"/>  
    <xsd:element name="Author" type="xsd:string" minOccurs="0" maxOccurs="unbounded"/>  
    <xsd:element name="Date" type="xsd:gYear"/>  
  </xsd:sequence>  
</xsd:complexType>
```

# Éléments BookStore et Publication

```
<xsd:element name="Publication" type="PublicationType"/>
```

```
<xsd:element name="BookStore">
```

```
  <xsd:complexType>
```

```
    <xsd:sequence>
```

```
      <xsd:element ref="Publication" maxOccurs="unbounded"/>
```

```
    </xsd:sequence>
```

```
  </xsd:complexType>
```

```
</xsd:element>
```

# Élément Book

```
<xsd:element name="Book" substitutionGroup="Publication"  
              type="BookType"/>
```

```
<xsd:complexType name="BookType">  
  <xsd:complexContent>  
    <xsd:extension base="PublicationType" >  
      <xsd:sequence>  
        <xsd:element name="ISBN" type="xsd:string"/>  
        <xsd:element name="Publisher" type="xsd:string"/>  
      </xsd:sequence>  
    </xsd:extension>  
  </xsd:complexContent>  
</xsd:complexType>
```



# Élément Magazine

```
<xsd:element name="Magazine" substitutionGroup="Publication"  
type="MagazineType" />
```

```
<xsd:complexType name="MagazineType">  
  <xsd:complexContent>  
    <xsd:restriction base="PublicationType">  
      <xsd:sequence>  
        <xsd:element name="Title" type="xsd:string"/>  
        <xsd:element name="Date" type="xsd:gYear"/>  
      </xsd:sequence>  
    </xsd:restriction>  
  </xsd:complexContent>  
</xsd:complexType>
```

```
<BookStore xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="bookstore.xsd">
```

```
<Book>
```

```
<Title>Persuasion</Title>
```

```
<Author>Jane Austen</Author>
```

```
<Date>1967</Date>
```

```
<ISBN>0-441-34319-4</ISBN>
```

```
<Publisher>Gallimard</Publisher>
```

```
</Book>
```

```
<Magazine>
```

```
<Title>Magic Revue Pop Moderne</Title>
```

```
<Date>2013</Date>
```

```
</Magazine>
```

```
<Book>
```

```
<Title>Meurtre sur la route de Bethléem</Title>
```

```
<Author>Batya Gour</Author>
```

```
<Date>2003</Date>
```

```
<ISBN>2-07-030898-7</ISBN>
```

```
<Publisher>Gallimard Foliopolicier</Publisher>
```

```
</Book>
```

```
</BookStore>
```

<BookStore> peut  
contenir des éléments  
qui se sont substitués  
à Publication!

# Élément d'unicité

Spécifie les éléments ou les attributs d'un élément qui doivent être uniques

## Notation

```
<xsd:element name="ElementComplexe" type="TypeElementComplexe">  
<xsd:unique name="nomContrainte" >  
<xsd:selector xpath="chemin-contrainte-unicite" />  
<xsd:field xpath="chemin-valeur-unique" />  
</xsd:unique >  
</xsd:element>
```

# Éléments sélector et field

- Element selector

Contient une expression *Xpath* spécifiant l'ensemble des éléments où la valeur spécifiée doit être unique

- Element field

Contient une expression *Xpath* spécifiant les valeurs qui doivent être uniques pour l'ensemble des éléments spécifiés par l'élément *selector*.

# Élément PlanTransport

Dans l'élément PlanTransport, l'élément *metro* de l'élément *transport* doit avoir une valeur unique

```
<xsd:element name="PlanTransport"
  type="PlanTransportType">
  <xsd:unique name="metroUnic">
    <xsd:selector xpath="./transport"/>
    <xsd:field xpath="metro"/>
  </xsd:unique>
</xsd:element>
```

# type PlanTransportType

```
<xsd:complexType name="PlanTransportType">
  <xsd:sequence minOccurs="0" maxOccurs="unbounded">
    <xsd:element ref="transport" minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>
<xsd:element name="transport" type="transportType"/>

<xsd:complexType name="transportType">
  <xsd:sequence>
    <xsd:element name="metro" type="metroType" nillable="true"/>
    <xsd:element name="ville" type="xsd:NMTOKENS" minOccurs="1" />
    <xsd:element name="description" type="xsd:normalizedString"
      minOccurs="0" maxOccurs="unbounded" />
    <xsd:element name="ligne" type="ligneType" minOccurs="1"
      maxOccurs="unbounded"/>
  </xsd:sequence>
  <xsd:attribute name="monuments" type="xsd:NMTOKENS" use="required"/>
</xsd:complexType>
```

# Vérification de l'unicité

XML validation started.

Checking file:/C:/Users/mj/Desktop/WEB\_2012/XML%20schema/ProjetEmployeKey/ParisPlanTransport.xml...

Referenced entity at

"file:/C:/Users/mj/Desktop/WEB\_2012/XML%20schema/ProjetEmployeKey/PlanTransport.xsd".

Valeur unique en double [Bonne nouvelle] déclarée pour la contrainte d'identité de l'élément "PlanTransport". [13]

XML validation finished.

```
<PlanTransport xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="PlanTransport.xsd">
  <transport monuments="Grand Rex">
    <metro>Bonne nouvelle</metro>
    <ville>Paris</ville>
    <description>la station est sur les lignes 8 et 9 en limite du 2e,9e et 10e arrondissements de
Paris</description>
    <ligne>9</ligne>
    <ligne>8</ligne>
  </transport>
  <transport monuments="Grand Rex">
    <metro>Bonne nouvelle</metro>
    <ville>Paris</ville>
    <description>la station est sur les lignes 8 et 9 en limite du 2e,9e et 10e arrondissements de
Paris</description>
    <ligne>9</ligne>
    <ligne>8</ligne>
  </transport>
</PlanTransport>
```

# Clés

- Spécifie qu'un attribut ou un élément doit être une clé, c'est à dire unique, non nulle et toujours présente.
- La clé doit être définie dans un `< element>` (appelé portée)
- La clé doit être à la fin de `<element>` (après le modèle du contenu et les déclarations d'attributs)



# Clés

## Notation

```
<xsd:element name="ElementPortee"  
              type="TypeElementPortee">  
  <xsd:Key name="nomKey" >  
    <xsd:selector xpath="chemin-élément-avec-clé" />  
    <xsd:field xpath="chemin-clé" />  
  </xsd:key >  
  ...  
</xsd:element>
```

# Clés Application Bookstore

## Gestion des achats

```
<xsd:element name="BookStore" type="BookStoreType"/>
<xsd:complexType name="BookStoreType">
  <xsd:sequence>
    <xsd:element name="Book" type="BookType" maxOccurs="unbounded"/>
    <xsd:element name="items" type="ItemsType"/>
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="ItemsType">
  <xsd:choice>
    <xsd:element name="item" type="ItemType" maxOccurs="unbounded"/>
  </xsd:choice>
</xsd:complexType>
```

# Types ItemType et BookType

```
<xsd:complexType name="ItemType">
  <xsd:all>
    <xsd:element name="price" type="xsd:double"/>
    <xsd:element name="nombre" type="xsd:integer"/>
  </xsd:all>
  <xsd:attribute name="book" type="xsd:string" use="required"/>
</xsd:complexType>
```

```
<xsd:complexType name="BookType">
  <xsd:sequence>
    <xsd:element name="Title" type="xsd:string"/>
    <xsd:element name="Author" type="xsd:string" minOccurs="0"
      maxOccurs="unbounded"/>
    <xsd:element name="Date" type="xsd:gYear"/>
    <xsd:element name="ISBN" type="xsd:string"/>
    <xsd:element name="Publisher" type="xsd:string"/>
  </xsd:sequence>
</xsd:complexType>
```

# Instance- bookstore.xml

```
<BookStore xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="bookstoreKey.xsd">
  <Book>
    <Title>Persuasion</Title>
    <Author>Jane Austen</Author>
    <Date>1967</Date>
    <ISBN>0-441-34319-4</ISBN>
    <Publisher>Gallimard</Publisher>
  </Book>
  <Book>
    <Title>Meurtre sur la route de Bethléem</Title>
    <Author>Batya Gour</Author>
    <Date>2003</Date>
    <ISBN>2-07-030898-7</ISBN>
    <Publisher>Gallimard Foliopolicier</Publisher>
  </Book>
  <items>
    <item book="2-07-030898-7">
      <number>2</number>
      <price>20</price>
    </item>
    <item book="2-07-030898-7">
      <number>5</number>
      <price>50</price>
    </item>
  </items>
</BookStore>
```

# Exemple définition de clé

Dans l'élément *BookStore*, l'élément *ISBN* de l'élément *Book* est une clé

```
<xsd:element name="BookStore" type="BookStoreType">  
  <xsd:key name="bookKey">  
    <xsd:selector xpath="./Book" />  
    <xsd:field xpath="ISBN"/>  
  </xsd:key>  
</xsd:element>
```

# Référence à un élément clé

Spécifie qu'un attribut ou une valeur d'élément référence la clef de l'élément spécifié

## Notation

```
<xsd:element name="ElementPortee"
              type="TypeElementPortee">
  <xsd:key name="nomKey">
    <xsd:selector xpath="chemin-element-avec-clé" />
    <xsd:field xpath="chemin-clé" />
  </xsd:key>
  <xsd:keyref name="nomKeyRef"
              refer="nomKey">
    <xsd:selector xpath="chemin-element-reference-clé" />
    <xsd:field xpath="chemin-reference"/>
  </xsd:keyref>
</xsd:element>
```

## Exemple de référence à un élément clé

Spécifie que dans l'élément **BookStore**, tout élément d'*items* ayant un attribut *number* référence la clé d'un livre

```
<xsd:element name="BookStore" type="BookStoreType">  
  <xsd:key name="bookKey">  
    <xsd:selector xpath="./Book" />  
    <xsd:field xpath="ISBN"/>  
  </xsd:key>  
  
  <xsd:keyref name="bookKeyRef" refer="bookKey">  
    <xsd:selector xpath="./items/*"/>  
    <xsd:field xpath="@number"/>  
  </xsd:keyref>  
</xsd:element>
```

# Elément <redefine>

redéfinit 0 à n composants d'un schéma référencé dans schemaLocation

```
<redefine schemaLocation="URL to schema document">  
    [simpleType or complexType or attributeGroup or group]*  
</redefine>
```



# Redéfinition de BookType

- Inclue les composants de LibraryBook.xsd
- Redéfinie *BookType* (de LibraryBook.xsd) en y ajoutant avec l'élément *Summary*.

```
<xsd:redefine schemaLocation="LibraryBook.xsd">  
  <xsd:complexType name="BookType">  
    <xsd:complexContent>  
      <xsd:extension base="BookType">  
        <xsd:sequence>  
          <xsd:element name="Summary" type="xsd:string"/>  
        </xsd:sequence>  
      </xsd:extension>  
    </xsd:complexContent>  
  </xsd:complexType>  
</xsd:redefine>
```

# Associer un document instance à plusieurs schémas

- Un document instance peut être composé d'éléments de plusieurs schémas
- Validation
  - ensemble du document
  - un élément

```

<?xml version="1.0"?>
<Library xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation=
    "http://www.book.org Book.xsd "
    http://www.employee.org Employee.xsd">
  <Books>
    <Book xmlns="http://www.book.org">
      <Title>Meurtre sur la route de Bethléem</Title>
      <Author>Bayta Gour</Author>
      <Date>2003</Date>
      <ISBN>2-07-0308985-7</ISBN>
      <Publisher>Gallimard FolioPolicier</Publisher>
    </Book>
  </Books>
  <Employees>
    <Employee xmlns="http://www.employee.org">
      <Name>Lola Montes</Name>
      <SSN>147-54-7698</SSN>
    </Employee>
  </Employees>
</Library>

```

Validation par  
rapport  
à deux schémas

Eléments <Library>,  
<Books>,  
et <Employees>  
Ne sont définis dans  
aucun schéma!

1. Validation de  
chaque  
élément  
Book par rapport à  
Book.xsd.
2. Validation de chaque  
élément Employee  
par rapport à  
Employee.xsd.
3. Eléments <Library>,  
<Books>,  
et <Employees>  
ne sont pas validés

# Associer un schéma à plusieurs schémas dans un même espace de noms

Element *include* permet d'accéder à des composants d'autres schémas  
Tous les schémas inclus doivent avoir le même espace de noms que l'espace de noms courant



A.xsd

```
<xsd:schema ...>  
  <xsd:include schemaLocation="A.xsd"/>  
  ...  
</xsd:schema>
```

...

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

Exemple include

```
<xsd:include schemaLocation="LibraryBook.xsd"/>
```

```
<xsd:element name="Library">
```

```
  <xsd:complexType>
```

```
    <xsd:sequence>
```

```
      <xsd:element name="Books">
```

```
        <xsd:complexType>
```

```
          <xsd:sequence>
```

```
            <xsd:element ref="Book" maxOccurs="unbounded"/>
```

```
          </xsd:sequence>
```

```
        </xsd:complexType>
```

```
      </xsd:element>
```

```
    </xsd:sequence>
```

```
  </xsd:complexType>
```

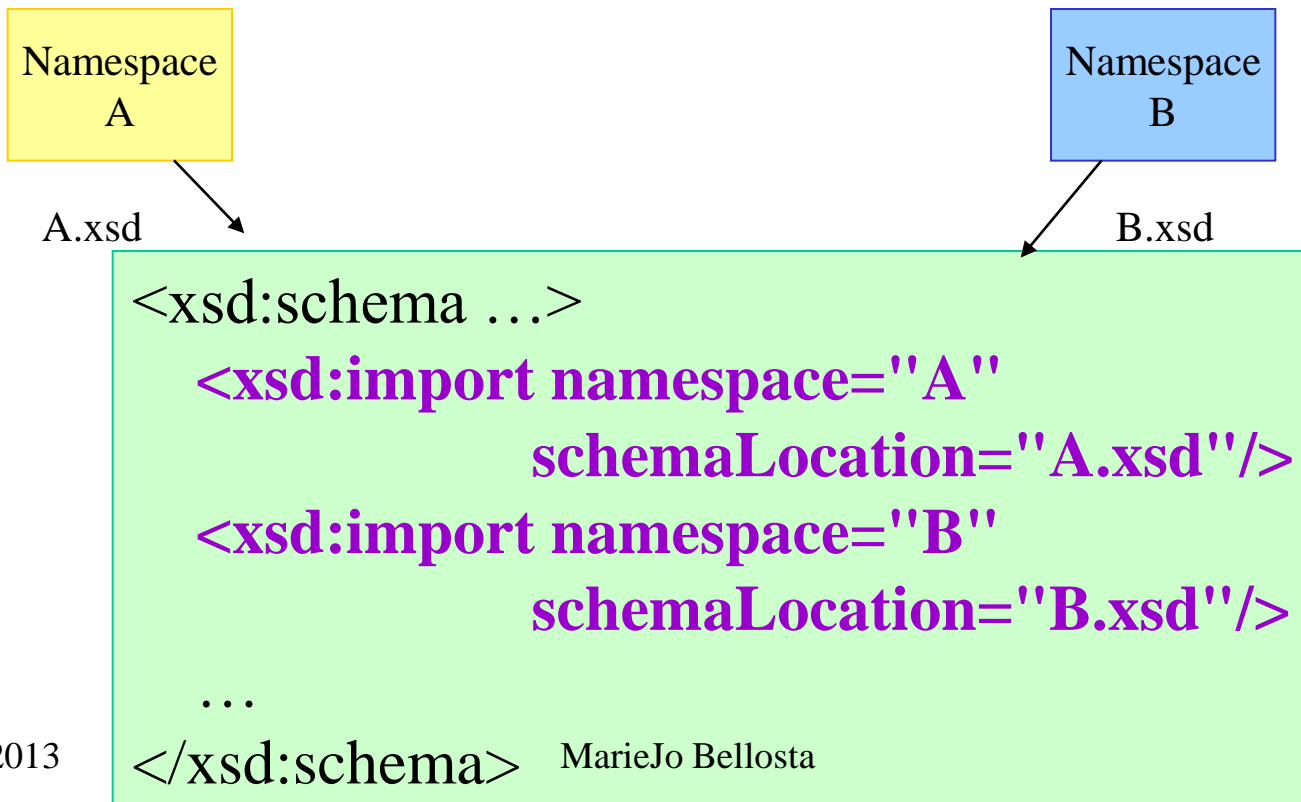
```
</xsd:element>
```

```
</xsd:schema>
```

13/11/2013

# Associer un schéma à plusieurs schémas dans des espaces de noms différents

Elément *import* permet d'accéder à des éléments et types de différents espaces de noms



```
<?xml version="1.0"?>
<xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://www.nikon.com"
  xmlns="http://www.nikon.com"
  elementFormDefault="qualified">
  <xsd:complexType name="body_type">
    <xsd:sequence>
      <xsd:element name="description"
        type="xsd:string"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:schema>
```

## Nikon.xsd

```
<?xml version="1.0"?>
<xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://www.olympus.com"
  xmlns="http://www.olympus.com"
  elementFormDefault="qualified">
  <xsd:complexType name="lens_type">
    <xsd:sequence>
      <xsd:element name="zoom"
        type="xsd:string"/>
      <xsd:element name="f-stop"
        type="xsd:string"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:schema>
```

## Olympus.xsd

```
<?xml version="1.0"?>
<xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://www.pentax.com"
  xmlns="http://www.pentax.com"
  elementFormDefault="qualified">
  <xsd:complexType name="manual_adapter_type">
    <xsd:sequence>
      <xsd:element name="speed"
        type="xsd:string"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:schema>
```

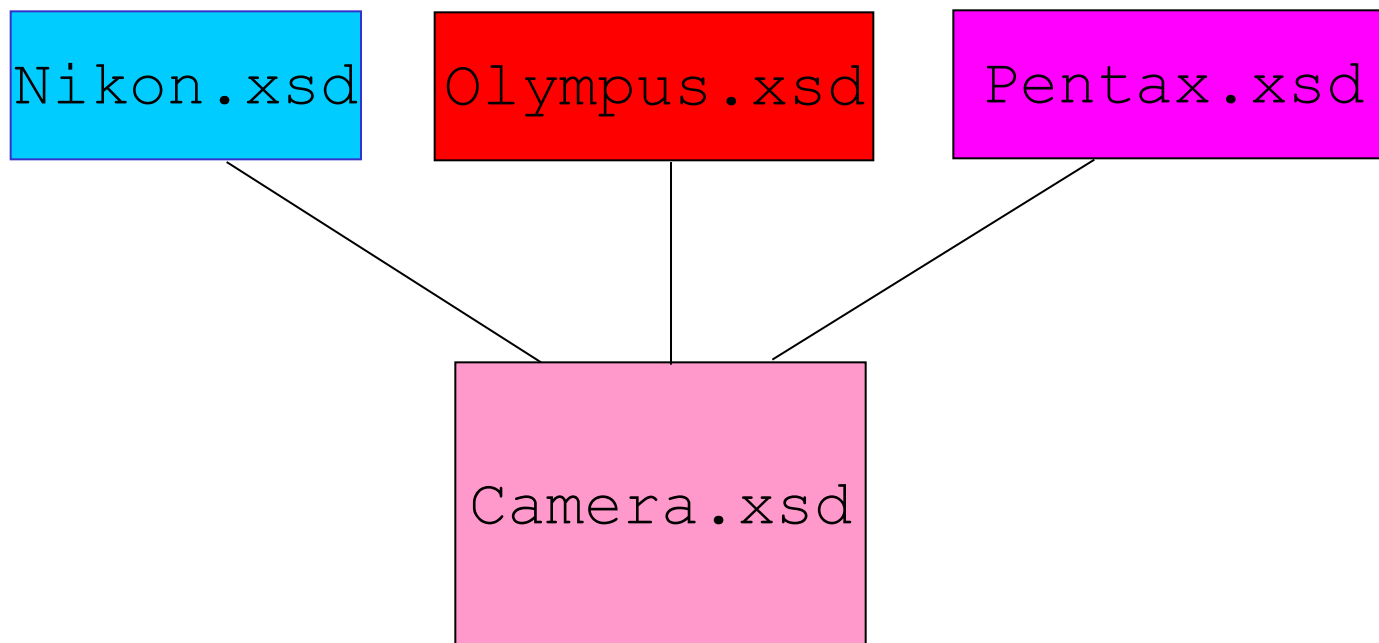
## Pentax.xsd

# Exemple élément *import*

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd=http://www.w3.org/2001/XMLSchema
  xmlns="http://www.camera.org"
  targetNamespace="http://www.camera.org"
  xmlns:nikon="http://www.nikon.com"
  xmlns:olympus="http://www.olympus.com"
  xmlns:pentax="http://www.pentax.com"
  elementFormDefault="qualified">
  <xsd:import namespace="http://www.nikon.com"    schemaLocation="Nikon.xsd"/>
  <xsd:import namespace="http://www.olympus.com"  schemaLocation="Olympus.xsd"/>
  <xsd:import namespace="http://www.pentax.com"    schemaLocation="Pentax.xsd"/>
  <xsd:element name="camera">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="body" type="nikon:body_type"/>
        <xsd:element name="lens" type="olympus:lens_type"/>
        <xsd:element name="manual_adapter" type="pentax>manual_adapter_type"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```



# Validation d'un document à partir de 3 schémas?



# elementFormDefault="qualified"

```
<?xml version="1.0"?>
<c:camera xmlns:c="http://www.camera.org"
  xmlns:nikon="http://www.nikon.com"
  xmlns:olympus="http://www.olympus.com"
  xmlns:pentax="http://www.pentax.com"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation=
    "http://www.camera.org
    Camera.xsd">
  <c:body>
    <nikon:description>Ergonomically designed casing for easy
      handling</nikon:description>
  </c:body>
  <c:lens>
    <olympus:zoom>300mm</olympus:zoom>
    <olympus:f-stop>1.2</olympus:f-stop>
  </c:lens>
  <c>manual_adapter>
    <pentax:speed>1/10,000 sec to 100 sec</pentax:speed>
  </c>manual_adapter>
</c:camera>
```

# elementFormDefault="unqualified"

Schéma source transparents à l'instance document

élément <description> ← schéma Nikon,

éléments <zoom> et <f-stop> ← schéma Olympus,

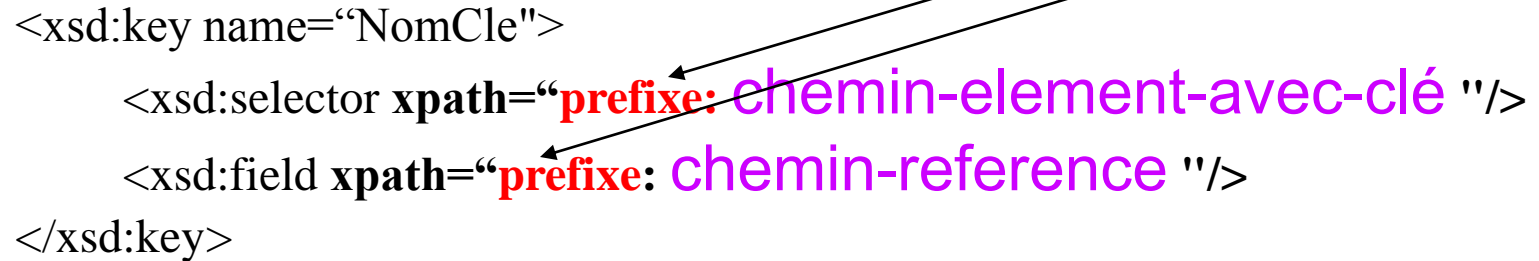
élément <speed> ← schéma Pentax

```
<?xml version="1.0"?>
<my:camera xmlns:my="http://www.camera.org"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation=
    "http://www.camera.org
    Camera.xsd">
  <body>
    <description>Ergonomically designed casing for easy handling</description>
  </body>
  <lens>
    <zoom>300mm</zoom>
    <f-stop>1.2</f-stop>
  </lens>
  <manual_adapter>
    <speed>1/10,000 sec to 100 sec</speed>
  </manual_adapter>
</my:camera>
```

# Clé ou unicité dans un espace de nom qualifié

Les références XPATH doivent être qualifiées:

```
<xsd:key name="NomCle">  
  <xsd:selector xpath="prefix: chemin-element-avec-clé "/>  
  <xsd:field xpath="prefix: chemin-reference "/>  
</xsd:key>
```



## Contrainte Xpath

Impératif même si l'espace de nom cible est l'espace de noms par défaut

elementFormDefault="unqualified" => l'expression xPath ne sera pas qualifiée

# Définir le préfixe de l'espace de nom par défaut

```
<xsd:schema  
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"  
    targetNamespace="http://www.books.org"  
    xmlns="http://www.books.org"  
    xmlns:bk="http://www.books.org"  
    elementFormDefault="qualified">
```

Ici bk est le préfixe du schéma <http://www.books.org>

# Définition de la clé en utilisant le préfixe bk dans l'expression de chemin xPath

```
<xsd:element name="BookStore">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="Book" type="BookType" maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:key name="PK">
    <xsd:selector xpath="./bk:Book"/>
    <xsd:field xpath="bk:ISBN"/>
  </xsd:key>
</xsd:element>
<xsd:complexType name=" BookType ">
  <xsd:sequence>
    <xsd:element name="Title" type="xsd:string"/>
    <xsd:element name="Author" type="xsd:string"/>
    <xsd:element name="Date" type="xsd:string"/>
    <xsd:element name="ISBN" type="xsd:string"/>
    <xsd:element name="Publisher" type="xsd:string"/>
  </xsd:sequence>
</xsd:complexType>
```

# Annoter un schéma

- Élément `<annotation>` est utilisé pour documenter un schéma à la fois pour les humains et les programmes
  - `<xsd:documentation>` pour un commentaires aux utilisateurs
  - `<xsd:appinfo>` pour un commentaire aux programmes
    - Le contenu est XML bien formé
- Annotations n'ont aucun effet sur la validation du schéma

## Exemple

```
<xsd:annotation>
```

```
<xsd:documentation>
```

The following constraint is not expressible with XML Schema:

The value of element A should be greater than the value of element B.

So, we need to use a separate tool (e.g., Schematron) to check this constraint.

We will express this constraint in the appinfo section (below).

```
</xsd:documentation>
```

```
<xsd:appinfo>
```

```
<assert test="A > B">A should be greater than B</assert>
```

```
</xsd:appinfo>
```

```
</xsd:annotation>
```

# Où peut-on mettre des annotations?

- avant ou après un composant global
- Seulement au début d'un composant non-global



```

...
<xsd:element name="BookStore">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="Book" maxOccurs="unbounded">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="Title" type="xsd:string"/>
            <xsd:element name="Author" type="xsd:string"/>
            <xsd:element name="Date" type="xsd:string"/>
            <xsd:element name="ISBN" type="xsd:string"/>
            <xsd:element name="Publisher" type="xsd:string"/>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
...

```

Annotations  
uniquement  
à ces endroits

# Annotation d'un élément

Annotation directe dans la déclaration de l'élément *Date*

```
...  
<xsd:sequence>  
  <xsd:element name="Title" type="xsd:string"/>  
  <xsd:element name="Author" type="xsd:string"/>  
  <xsd:element name="Date" type="xsd:string">  
    <xsd:annotation>  
      <xsd:documentation>  
        Ceci montre comment commenter l'élément  
      </xsd:documentation>  
    </xsd:annotation>  
  </xsd:element>  
  <xsd:element name="ISBN" type="xsd:string"/>  
  <xsd:element name="Publisher" type="xsd:string"/>  
</xsd:sequence>
```

# Attributs pour l 'élément *documentation*

- **source**: comprend un identifiant de ressource pour un fichier qui contient des informations supplémentaire
- **xml:lang**: spécifie le langage dans lequel est exprimé la documentation

## Exemple

```
<xsd:documentation source="http://www.xfront.com/BookReview.txt"  
                  xml:lang="FR"/>
```

# Attribut pour l'élément *appinfo*

- **source**: comprend un identifiant de ressource pour un fichier qui contient des informations supplémentaires

```
<xsd:appinfo source="http://www.xfront.com/Assertions.xml"/>
```