TP3 : DTD & XSD

Exercice N°1 :

On souhaite représenter un carnet d'adresses en XML. Pour chaque entrée du carnet, on veut conserver les informations suivantes :

- nom et prénom de la personne
- adresse
- numéro de téléphone (éventuellement plusieurs)
- adresse mail (éventuellement plusieurs)
- date d'anniversaire

Questions :

1. Écrire une DTD pour le carnet d'adresses.

2. Écrire un fichier XML valide pour cette DTD, comportant au moins deux entrées et faisant apparaître toutes les possibilités de la DTD (c'est-à-dire toutes les valeurs possibles pour les attributs et tous les éléments et attributs optionnels).

3. On souhaite donner une version plus professionnelle du carnet d'adresses de l'exercice 1.1. Pour chaque entrée du carnet, on conserve les informations suivantes :

- nom et prénom de la personne
- numéro de téléphone (éventuellement plusieurs)
- adresse mail (éventuellement plusieurs)
- service
- entreprise (avec le site)

L'adresse de chaque contact est celle de son entreprise. Le fichier XML doit donc contenir, pour chaque entreprise, les informations suivantes :

- nom
- une liste de sites, avec pour chaque site :
    - nom du site
    - adresse du site

Pour éviter la redondance dans le fichier, le lien entre une personne et le site de l'entreprise pour laquelle elle travaille est obtenu grâce à une référence croisée.

a. Ecrire une DTD pour la nouvelle version du carnet.

b. Écrire un fichier valide pour cette DTD, comportant au moins une entreprise à deux sites et au moins deux personnes (une pour chaque site).

Les références croisées :

Les types ID, IDREF et IDREFS permettent des références croisées au sein d'un document :

- ID : pour associer un label à un élément un seul attribut ID par élément #REQUIRED ou #IMPLIED contenu unique dans un document
- IDREF et IDREFS : pour faire une référence à un label doit obligatoirement faire référence à un label existant IDREFS : plusieurs labels séparés par des espaces
- le validateur vérifie les références croisées et permet (entre autres) d'éviter la redondance dans un fichier XML

disques.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE disques SYSTEM "DisquesML.dtd">
<disques>
<groupe nom="muse">
<nom>MUSE</nom>
<membre>Matthew Bellamy</membre>
<membre>Dominic Howard</membre>
<membre>Chris Wolstenholme</membre>
</groupe>
<disque>
<interprète nom="muse"/>
<titre>Showbiz</titre>
</disque>
<disque>
<interprète nom="muse"/>
<titre>Origin of symmetry</titre>
</disque>
</disques>
```

DisquesML.dtd

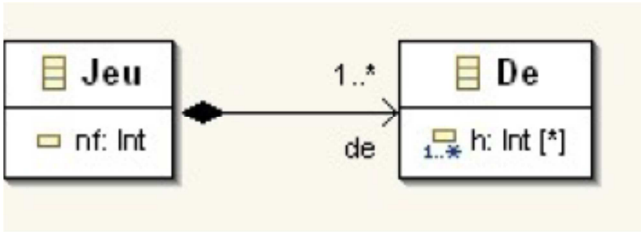
```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!ELEMENT disques (groupe*, disque*)  >
<!ELEMENT groupe (nom,membre+) >
<!ATTLIST groupe nom ID #REQUIRED>
<!ELEMENT nom (#PCDATA) >
<!ELEMENT membre (#PCDATA) >
<!ELEMENT disque (interprète, titre)>
<!ELEMENT interprète EMPTY >
<!ATTLIST interprète nom IDREF #REQUIRED>
<!ELEMENT titre (#PCDATA) >
```

Exercice N°2 :

On demande de modéliser par un schéma XML un jeu de n dés à nf faces.

Jeu de dés

Voici le diagramme de classe de ce modèle :



1. Ecrire le schéma XSD
2. Ecrire un document XML valide

Exercice N°3 :

Une entreprise de vente de matériel informatique désire garder les informations de son stock dans un document XML. Vous disposez des informations suivantes :

- Le stock contient plusieurs produits.
- Chaque produit identifié par un identifiant unique (idprod) est reconnu par sa marque, son modèle et son fournisseur.
- Chaque produit appartient à une catégorie donnée.
- Chaque catégorie identifiée par « idcat » est reconnue par son libellé. Le idcat doit nécessairement commencer par la lettre C suivie de 5 chiffres.
- Chaque fournisseur identifié par « idfour » est reconnu par sa raison sociale, son adresse et son téléphone fixe. L'adresse du fournisseur ne doit pas dépasser les 40 caractères.

1. Ecrire un exemple de document XML répondant aux besoins de cette agence
2. Ecrire un schema XML (XSD) qui valide ce document XML.

Exercice N°4 :

Soit le document company.xml suivant :

```
<Company>
  <Address type="US-Address">
    <Name>Main office</Name>
    <Street>Grosvenor Bvld.</Street>
    <City>Los Angeles</City>
    <State>California</State>
    <Zip>3141</Zip>
  </Address>
  <Division>
    <Division-Name>Sales</Division-Name>
    <Location>Washington</Location>
    <Person Manager="true" Degree="MA">
      <First>Allison</First>
      <Last>Andersen</Last>
      <PhoneExt>111</PhoneExt>
      <EMail>Andersen@work.com</EMail>
    </Person>
    <Person Manager="false" Degree="BA">
      <First>Bruce</First>
      <Last>Burrwinkle</Last>
      <PhoneExt>222</PhoneExt>
      <EMail>Burrwinkle@work.com</EMail>
    </Person>
  </Division>
  <Division>
    ……
  </Division>
</Company>
```

Questions :

1. Ecrire la DTD correspondante à ce document XML
2. Ecrire XSD correspondant à ce document XML
3. Ecrire le document XSL présentant les données de ce fichier XML sous la forme suivante :

Données de la société (Company)

Pour chaque département (Division), il faut créer un tableau des employés.

Ex :

Sales (Washington)

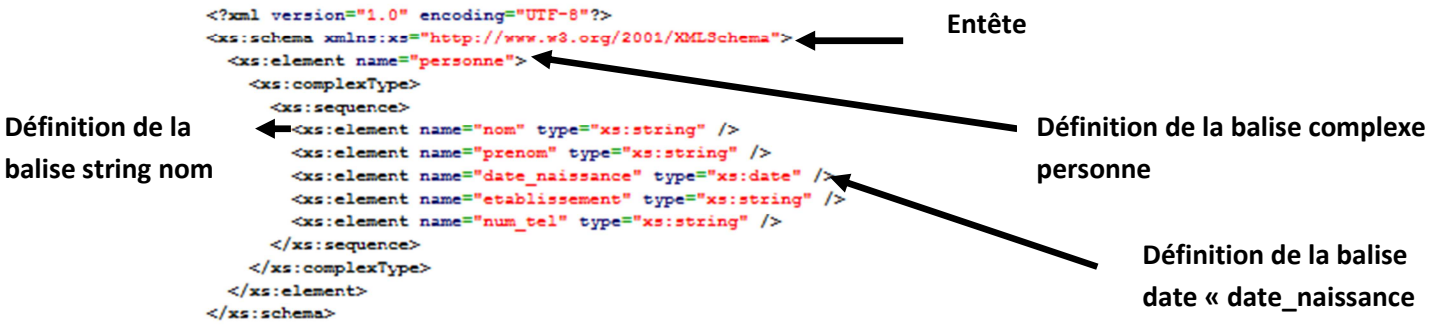
First	Last	Manager	Degree	Phone	E-mail

Annexe

Fichier XSD :

XML Schema est un langage de description de format de document XML permettant de définir la structure d'un document XML. La connaissance de la structure d'un document XML permet de vérifier la validité de ce document. Un fichier de description de structure (XML Schema Definition ou XSD) est lui-même un document XML. Ce langage de description de contenu de documents XML est lui-même défini par un schéma, dont les balises de définition s'auto-définissent (c'est un exemple de définition récursive).

Un exemple de fichier XSD:




```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="personne">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="nom" type="xs:string" />
        <xs:element name="prenom" type="xs:string" />
        <xs:element name="date_naissance" type="xs:date" />
        <xs:element name="etablissement" type="xs:string" />
        <xs:element name="num_tel" type="xs:string" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Suivi d'un fichier XML valide:

```
<?xml version="1.0" encoding="UTF-8"?>
<personne xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="personne.xsd">
  <nom>MBODJ</nom>
  <prenom>Babacar</prenom>
  <date_naissance>1996-10-06</date_naissance>
  <etablissement>NIIT</etablissement>
  <num_tel>764704140</num_tel>
</personne>
```

XML-Schema permet d'utiliser des données :

- de type prédéfini (string, int…)
- de type complexe
- dont le type est une restriction de type
- dont le type est une extension de type

Types prédéfinis :

- byte, unsignedByte, hexBinary, integer, positiveInteger, negativeInteger, int, unsignedInt, long, unsignedLong, short, unsignedShort, decimal, float, double…
- string, NormalizedString, token
- boolean, anyURI, language
- time, dateTime, duration, date, gMonth, gYear, gYearMonth, gDay, gMonthDay
- ID, IDREF, IDREFS, ENTITY, ENTITIES, NOTATIN, NMTOKEN, NMTOKENS

Exemple : <xsd:element name="comment" type="xsd:string"/>

Types complexes :

Exemple : le type de données *TypeAdresse* se compose de 6 éléments *Numero, Rue1, Rue2, Ville, CP et Pays* :

```
<xs:complexType name="TypeAdresse">
  <xs:sequence>
    <xs:element name="Numero"  type="xs:positiveInteger" />
    <xs:element name="Rue1"  type="xs:string" />
    <xs:element name="Rue2"  type="xs:string" />
    <xs:element name="Ville"  type="xs:string" />
    <xs:element name="Numero"  type="xs:positiveInteger" />
    <xs:element name="CP"  type="xs:decimal" />
    <xs:element name="Pays"  type="xs:string" fixed="France" />
  </xs:sequence>
</xs:complexType>
```

Restriction de type existant :
Exemple : le type de données string comprend 6 attributs optionnels : pattern, enumeration, length, minlength, maxlength, whitespace. Si on désire représenter un choix Oui/Non (restriction sur l'attribut enumeration) :

```
<xs:simpleType name="choixOuiNon">
  <xs:restriction base="xs:string">
  <xs:enumeration value="oui"/>
  <xs:enumeration value="non"/>
  </xs:restriction>
</xs:simpleType>
<xs:element name="choix" type="choixOuiNon"/>
```

Extension / dérivation de type existant
Exemple : si l'on souhaite créer un type Personne contenant en plus du nom et du prénom, un élément de type Adresse (extension du type Adresse vu précédemment) :

```
<xs:complexType name="Personne">
  <xs:complexContent>
    <xs:extension  base="adresse">
    <xs:sequence>
    <xs:element name="Nom" type="xs:string"/>
    <xs:element name="Prenom" type="xs:string"/>
    </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Déclaration des attributs :
Exemple de l'élément montant dans un document XML :
<montant devise = "EURO" horsTaxe="TRUE" />
L'attribue devise est optionnel et a comme valeur pr défaut « EURO ».
L'attribut hors taxe est obligatoire et a comme valeur par défaut « TRUE »
Modélisation en XML-Schema

```
<xs:element name="element">
<xs:complexType>
<xs:attribute name="devise" type="xs:string" use="optional" value="EURO"/>
<xs:attribute name="horsTaxe" type="xs:boolean" use="required" value="TRUE"/
</xs:complexType>
</xs:element>
```

Possibilité de référencer un élément :
<element ref="Nom" />