Sujet 2. Conditions d'examen

- Durée 1H30
- Tous documents électroniques déposés sur votre ordinateur.
- Accès interdit à Internet
- Rendu de l'examen sur clé USB (l'administration vous fournira la clé USB à la fin de l'examen)

Exercice 1 : Bases du langage XSL

On vous donne ci-dessous un fichier XML qui décrit de façon simplifiée le schéma d'une base de données. Ce fichier a été emprunté à Mr Daniel Lucazeau sur le site sam-mag.com.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<database name="infos">
   <tables>
      <table name="codeDepartement">
         <descRecord>
            <field name="IDDep" type="string"/>
            <field name="NOM" type="string"/>
         </descRecord>
         <records>
            <record>
               <field name="IDDep">01</field>
               <field name="NOM">Ain</field>
            </record>
         </records>
      </table>
      <table name="codeRegion">
         <descRecord>
            <field name="IDRegion" type="int"/>
            <field name="nom" type="string"/>
          </descRecord>
          <records>
            <record>
               <field name="IDRegion">1</field>
               <field name="nom">Alsace</field>
            </record>
         </records>
      </table>
   </tables>
</database>
```

Question. Ecrire une feuille XSL qui permet d'afficher dans un fichier HTML les caractéristiques d'une table dont le nom sera passé en paramètre dans la ligne de commande. Si la table choisie est ***codeDepartement*** la présentation sera comme suit :

Table : codeDepartement – nombre d'enregistrements :2

Champ	type
IDRegion	String
nom	string

Note : A min avis il devrait plutôt y avoir IDDep au lieu de IDRegion pour cet exemple.

Le tableau ci-dessus est celui que vous devez obtenir ave l'exemple. Votre code XSL doit bien sûr s'appliquer à tout arbre XML ayant la même structure que l'exemple.

Réponse : B arême 5 points

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!-- entête commune à tous les fichiers XMl -->
<!-- Début de la feuille XSL -->
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="html" indent="yes" encoding="utf-8" />
<xsl:param name="nomTable"/>
<!-- le patron du noeud database -->
<xsl:template match="database">
<html>
        <head>
        </head>
        <body>
         <xsl:text> Ici exercice 1. Vous demandez la table </xsl:text><xsl:value-of select="$nomTable"/>
<br/>  <xsl:apply-templates select="tables/table[@name=$nomTable]"/>
        </body>
</html>
</xsl:template>
<xsl:template match="table">
 <xsl:text>Table :</xsl:text> <xsl:value-of select="@name"/>
 <xsl:text> -- nombre d'enregistrements : </xsl:text>
 <xsl:value-of select="count(descRecord/field)"/>
        <table border="1px">
                        <tr>
                        <td>Champ</td>
                        <td>type</td>
                </tr>
                <xsl:for-each select="descRecord/field">
                        <tr>
```

```
                    <td> <xsl:value-of select="@name"/></td>

                    <td><xsl:value-of select="@type"/></td>

                </tr>

            </xsl:for-each>

    </table>

</xsl:template>

<!-- fin de la feuille xsl-->

</xsl:stylesheet>
```

Exercice 2 : Navigation et filtre avec XSL

On reprend le fichier XML de l'exercice ci-dessus.

Question. Ecrire la feuille XSL qui permet d'afficher dans un fichier texte le contenu des 10 premières lignes (au maximum) de la table (des tables) contenant le plus grand nombre d'enregistrements. S'il y a plusieurs tables qui ont le plus grand nombre d'enregistrements on n'affiche que la première dans le schéma. L'affichage doit se présenter comme suit :

Nom de la table
valeur du champ1, … , valeur du champs n
….

Réponse : Barême 5 points

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!-- entête commune à tous les fichiers XMl -->

<!-- Début de la feuille XSL -->
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="html" indent="yes" encoding="utf-8" />

<!-- le patron du noeud database -->
<xsl:template match="database">
<html>
        <head>
        </head>
        <body>

<br/>
```

```
<xsl:for-each select="tables/table">
   <xsl:sort select="count(descRecord/field)"/>
      <xsl:if test="position()=last()">
         <xsl:value-of select="@name"/>
         <xsl:apply-templates select="."/>
      </xsl:if>
</xsl:for-each>

</body>
</html>
</xsl:template>

<xsl:template match="table">

                        <xsl:for-each select="records/record[position()&lt;10]">
                              <xsl:for-each select="field">
                              <xsl:value-of select="."/>
<xsl:if test="not( position()=last() )">
<xsl:text> , </xsl:text>
</xsl:if>
                        </xsl:for-each>
<br/>
                        </xsl:for-each>



</xsl:template>
<!-- fin de la feuille xsl-->
</xsl:stylesheet>
```

Exercice 3 : XSD et DTD

1. XSD :
   Thème cinéma
   Donner l'extrait de schéma XSD qui permet de représenter une équipe de rugby qui est constituée :
   - d'un club : attribut texte
   - d'une division : attribut parmi : Top 14, Pro 12, Fédérale 1, Fédérale 2, Fédérale 3, Honneur, Promotion d'Honneur, Première série, Deuxième série, Troisième série, Quatrième série
   - d'au moins un entraineur
   - d'au moins 15 joueurs

   Les entraineurs et les joueurs sont constitués :
   - d'un nom : attribut texte
   - d'un prénom : attribut texte

Réponse : Barême 3 points

```
<xsd:element name="film" type="equipeRugby"/>
```

```
<xsd:complexeType name="equipeRugby">

<xsd:attribute name="club" type="xsd:string"/>
<xsd:attribute name="division" type="typeDiv"/>

<xsd:sequence>
<xsd:element name="entraineur" type="typeIndividu" minOccurs="1"
maxOccurs="unbounded"/>
<xsd:element name="joueur" type="typeIndividu" minOccurs="15"
maxOccurs="unbounded"/>
</xsd:sequence>
</xsl:complexeType>


<xsd:simpleType name="typeDiv">
  <xs:restriction base="xs:string">
      <xs:enumeration value="Top 14"/>
      <xs:enumeration value="Pro D2"/>
      <xs:enumeration value="Fédérale 1"/>
       <xs:enumeration value="Fédérale 2"/>
        <xs:enumeration value="Fédérale 3"/>
       <xs:enumeration value="Honneur"/>
      <xs:enumeration value="Promotion d'Honneur"/>
      <xs:enumeration value="Première série"/>
      <xs:enumeration value="Deuxième série"/>
      <xs:enumeration value="Troisième série"/>
      <xs:enumeration value="Quatrième série"/>
    </xs:restriction>
</xsd:simpleType >

<xsd:complexeType name="typeIndividu">
  <xsd:attribute name="nom" type="xsd:string"/>
<xsd:attribute name="prenom" type="xsd:string"/>
</xsd:simpleType>
```

2. DTD

On considère 3 possibilités de définitions de l'élément e :

1. <!ELEMENT e(a*|b*)*>
2. <!ELEMENT e(a+|b+)>
3. <!ELEMENT e(a,b)*>

Les éléments a et b sont définis comme

<!ELEMENT a EMPTY>
<!ELEMENT b EMPTY>

- Pour chaque élément, cocher la (ou les) DTD(s) qui le valide(nt) :

1. <e><a/><b/><a/><b/><a/><b/><a/><b/></e> o 1 o 2 o 3
2. <e><a/><b/><a/><a/><b/><a/><b/><b/><a/><b/></e> o 1 o 2 o 3
3. <e><a/><a/><a/></e> o 1 o 2 o 3
4. <e/> o 1 o 2 o 3

Réponse : Barême 2 points

1. <e><a/><b/><a/><b/><a/>/><a/><b/><a/><b/></e> • 1 o 2 o 3
2. <e><a/><b/><a/><a/><b/><a/><b/><b/><a/><b/></e> • 1 o 2 o 3
3. <e><a/><a/><a/></e> • 1 • 2 o 3
4. <e/> • 1 o 2 • 3

Exercice 4 : Récursivité avec XSL

Soit un ensemble E de n éléments. On désire le décomposer en deux sous-ensembles les plus égaux possibles en taille : E(n/2) pour le premier et n-E(n/2) pour le deuxième. Puis chaque sous-ensemble est décomposé de la même façon et ainsi de suite, jusqu'à obtenir des singletons.

On crée de plus une notion de niveau, sachant que le niveau de départ est 1. Le niveau des 2 sous-ensembles est 2 etc.

On utilisera l'algorithme suivant :

```
procedure decoupage(niveau Entier, taille Entier)
début
    Ecrire('niveau : ', niveau, 'taille : ', taille)
    Si taille >= 2 Alors
        decoupage(niveau+1, E(taille/2))
        decoupage (niveau+1, taille-E(taille/2))
    Fin Si
```

Traduire cet algorithme en XSL avec un template nommé.

Réponse : Barême : 5 points

```
<xsl:template name="decoupage">
    <xsl:param name="niveau"/>
    <xsl:param name="taille"/>
```

```
        <xsl:text> Niveau: </xsl:text> <xsl:value-of select="$niveau"/><xsl:text> Taille: </xsl:text> <xsl:value-
of select="$taille"/>
<br/>
<xsl:choose>
        <xsl:when test="$taille != 1">

                <xsl:call-template name="decoupage">
                        <xsl:with-param name="niveau" select="$niveau+1"/>
                         <xsl:with-param name="taille" select="floor(number($taille) div 2)"/>
                </xsl:call-template>

      <xsl:call-template name="decoupage">
                        <xsl:with-param name="niveau" select="$niveau+1"/>
                        <xsl:with-param name="taille" select="number($taille)-floor(number($taille) div 2)"/>
                </xsl:call-template>
        </xsl:when>

   <xsl:otherwise>
      <xsl:text>Fin</xsl:text><br/>
   </xsl:otherwise>



</xsl:choose>
</xsl:template>
```