LIFBDW2 – BASES DE DONNÉES AVANCÉES
TD10 – XML, XPath

Licence informatique – Automne 2019–2020

Les questions marquées du symbole (†) sont à préparer pour la séance

Exercice 1 : Espaces de nommage XML

1. Pour chaque élément du document XML ci-dessous, donner son espace de nommage.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<categorie xmlns="http://macollection.com">
  <nom >
    Categorie <em xmlns="http://www.w3.org/1999/xhtml">A</em></nom>
  <categorie>
    <nom >B</nom>
    <categorie xmlns:xhtml="http://www.w3.org/1999/xhtml">
      <nom ><xhtml:em >C</xhtml:em></nom>
    </categorie>
  </categorie>
  <coll:categorie xmlns="http://www.w3.org/1999/xhtml"
          xmlns:coll="http://macollection.com">
    <coll:nom ><b >D</b></coll:nom>
    <categorie >
      <coll:nom >E</coll:nom>
    </categorie>
  </coll:categorie>
</categorie>
```

Exercice 2 : Correspondance avec une DTD

1. Vérifier si le document XML donné en annexe A correspond bien à la définition donnée et le corriger si besoin.

Exercice 3 : Requêtes XPath

On considère le document XML donné en annexe A. Évaluer les requêtes XPath suivantes, à partir de la racine puis du premier noeud `artist` du document. Détailler bien chaque étape, en indiquant le numéro de ligne correspondant au début de chaque noeud sélectionné.

1. `/collection/artist`
2. `collection/artist`
3. `self::node()//artist[album/track/@seconds >= 152]` [1]
4. `album/track/title[starts-with(child::text(),'Serious')]`

1. La commande `self::node()` permet de s'assurer que l'on ne remonte *pas* jusqu'à la racine pour trouver les artistes.

Exercice 4 : Requêtes XPath

Donner une expression XPath des requêtes suivantes sur le document XML donné en annexe A.

1. La première piste de chaque album.
2. L'ensemble des titres de pistes
3. Les pistes dont la durée est supérieure 170.
4. Les albums dont on ne connait pas le nom

A Collection XML

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE collection
[
<!ELEMENT collection (artist*)>
<!ELEMENT artist (name,album+)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT album (title?,track+)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT track (title)>
<!ATTLIST track seconds CDATA #REQUIRED>
]>
<collection>
  <artist>
    <name>Cool guy</name>
    <album>
      <track seconds="183">
        <title>This is cool</title>
      </track>
      <track seconds="451">
        <title>Cool song</title>
      </track>
    </album>
    <album>
      <title>Yet Another</title>
      <track>
        <title>YA Song</title>
      </track>
    </album>
  </artist>
  <artist>
    <album>
      <title>Serious</title>
      <track seconds="150">
        <title>Serious part 1</title>
      </track>
      <track seconds="152">
        <title>Serious part 2</title>
      </track>
      <track seconds="152">
        <title>Serious part 3</title>
      </track>
      <track seconds="153">
        <title>Serious part 4</title>
      </track>
    </album>
  </artist>
</collection>
```

Corrections

Solution de l'exercice 1

1. En notant ++ pour "http://macollection.com" et ** pour "http://www.w3.org/1999/xhtml"

```
<?xml version="1.0" encoding="iso-8859-1"?>
<categorie++ xmlns="http://macollection.com">
  <nom++ >
    Categorie <em** xmlns="http://www.w3.org/1999/xhtml">A</em></nom>
  <categorie>++
    <nom++ >B</nom>
    <categorie++ xmlns:xhtml="http://www.w3.org/1999/xhtml">
      <nom++ ><xhtml:em** >C</xhtml:em></nom>
    </categorie>
  </categorie>
  <coll:categorie++ xmlns="http://www.w3.org/1999/xhtml"
          xmlns:coll="http://macollection.com">
    <coll:nom++ ><b** >D</b></coll:nom>
    <categorie** >
      <coll:nom++ >E</coll:nom>
    </categorie>
  </coll:categorie>
</categorie>
```

Solution de l'exercice 2

1. Erreurs
   — l.25 : ajouter un attribut `seconds`
   — l.30 : ajouter un élément `name`

Solution de l'exercice 3

Convention : noeud repéré par le numéro de ligne. Si c'est un attribut, on précise en plus (att), si c'est un noeud texte, on précise en plus (txt)

Note : la fonction starts-with, qui est pré-définie dans XQuery, renvoie vrai son premier argument est une chaîne de caractère qui débute par son second argument.

1. 1. A partir de la racine :
      — `/` $\rightarrow \{1\}$
      — `/collection` $\rightarrow \{12\}$
      — `/collection/artist` $\rightarrow \{13, 30\}$
   2. Idem à partir du 1er noeud `artist` car `/` au début retourne à la racine.
2. 1. A partir de la racine :
      — `collection` $\rightarrow \{12\}$
      — `collection/artist` $\rightarrow \{13, 30\}$
   2. A partir du 1er noeud `artist` :
      — `collection` $\rightarrow \emptyset$
3. 1. A partir de la racine :
      — `self::node()` $\rightarrow \{1\}$
      — `self::node()//artist` $\rightarrow \{13, 30\}$

— 13 : `album/track/@seconds` → {16(`att`), 19(`att`), 25(`att`)}, l.25 l'attribut `seconds` vaut plus de 152 → true.
— 30 : `album/track/@seconds` → {33(`att`), 36(`att`), 39(`att`), 42(`att`)}, l.39 l'attribut `seconds` vaut plus de 152 → true.
— `self::node()//artist[album/track/@seconds >= 152]` → {13, 30}

2. A partir du premier noeud `artist` :
— `self::node()` → {13}
— `self::node()//artist` → ∅

4. 1. A partir de la racine :
— `album` → ∅

2. A partir du premier noeud `artist` :
— `album` → {15, 22}
— `album/track` → {16, 19, 25}
— `album/track/title` → {17, 20, 26}
— 17 : `child::text()` → {17(`txt`)} → false.
— 20 : `child::text()` → {20(`txt`)} → false.
— 26 : `child::text()` → {26(`txt`)} → false.
— `album/track/title[starts-with(child::text(),'Serious']` → ∅

Solution de l'exercice 3

1. `//album/track[1]`
2. `//track/title`
3. `//track[@seconds >=170]`
4. `//album[not(title)]`