Bases de Données II, Mons, 10 juin 2013

Cahier fermé. Durée : 3 heures

Nom et prénom
Année

Question 1 La figure 1 montre un schéma XML pour une bibliothèque. Concevez une DTD qui accepte exactement les mêmes documents XML que ceux qui sont acceptés par le schéma XML.

.../9

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="fullname">
        <xs:complexType>
            <xs:all> <xs:element name="first" type="xs:string" minOccurs="0"/>
                     <xs:element name="middle" type="xs:string" minOccurs="0"/>
                     <xs:element name="last" type="xs:string"/>
            </xs:all>
        </xs:complexType>
    </xs:element>
    <xs:element name="simplename" type="xs:string"/>
    <xs:element name="born" type="xs:integer"/>
    <xs:group name="name">
        <xs:choice> <xs:element ref="simplename"/>
                    <xs:element ref="fullname"/>
        </xs:choice>
    </xs:group>
    <xs:simpleType name="gender">
        <xs:restriction base="xs:string"> <xs:enumeration value="M"/>
                                          <xs:enumeration value="F"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:element name="author">
        <xs:complexType>
            <xs:sequence> <xs:group ref="name"/>
                          <xs:element ref="born" minOccurs="1"/>
                          <xs:element name="dead" type="xs:integer" minOccurs="0"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="character">
        <xs:complexType>
            <xs:sequence> <xs:group ref="name"/>
                          <xs:element ref="born"/>
            </xs:sequence>
            <xs:attribute name="gender" type="gender"/>
        </xs:complexType>
    </xs:element>

    <xs:element name="library">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="book" maxOccurs="unbounded">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="title" type="xs:string"/>
                            <xs:element ref="author" minOccurs="1" maxOccurs="unbounded"/>
                            <xs:element ref="character" minOccurs="0" maxOccurs="unbounded"/>
                        </xs:sequence>
                        <xs:attribute name="available" type="xs:string" use="required"/>
                    </xs:complexType>
                </xs:element>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
</xs:schema>
```

FIGURE 1 – XML Schema

Bases de Données II, Mons, 10 juin 2013

Nom et prénom
Année

La figure 3 montre une base de données XML pour stocker les podiums des courses cyclistes annuelles. Chaque coureur est identifié par un code unique (attribut `id`). Pour enregistrer les podiums, on utilise le *document order* d'XML ; par exemple, le gagneur du Tour des Flandres en 2011 était Nick Nuyens, avant Sylvain Chavanel (deuxième) et Fabian Cancellara (troisième).

Les noms des courses ne contiennent pas de blancs ou symboles spéciaux à l'intérieur.

La figure 2 montre le DTD.

```
<!-- This file is called courses.dtd -->
<!ELEMENT CYCLISME (COUREURS, COURSES)>
<!ELEMENT COUREURS (COUREUR*)>
<!ELEMENT COURSES (COURSE*)>
<!ELEMENT COURSE (PODIUM*)>
<!ELEMENT PODIUM (COUREUR*)>
<!ELEMENT COUREUR (#PCDATA)>
<!ATTLIST COUREUR id CDATA #REQUIRED>
<!ATTLIST COUREUR naissance CDATA #IMPLIED>
<!ATTLIST COUREUR nat CDATA #IMPLIED>
<!ATTLIST COURSE nom CDATA #REQUIRED>
<!ATTLIST PODIUM annee CDATA #REQUIRED>
```

FIGURE 2 – DTD.

```
<CYCLISME>
<COUREURS>
   <COUREUR id="tb" naissance="1980" nat="B">Tom Boonen</COUREUR>
   <COUREUR id="pg" naissance="1982" nat="B">Philippe Gilbert</COUREUR>
   <COUREUR id="nn" naissance="1980" nat="B">Nick Nuyens</COUREUR>
   <COUREUR id="sc" naissance="1979" nat="F">Sylvain Chavanel</COUREUR>
   <COUREUR id="fc" naissance="1981" nat="CH">Fabian Cancellara</COUREUR>
   <COUREUR id="fp" naissance="1981" nat="I">Filippo Pozzato</COUREUR>
   <COUREUR id="ab" naissance="1979" nat="I">Alessandro Ballan</COUREUR>
   <COUREUR id="jv" naissance="1981" nat="B">Johan Vansummeren</COUREUR>
   <COUREUR id="mt" naissance="1977" nat="NL">Maarten Tjallingii</COUREUR>
   <COUREUR id="st" naissance="1984" nat="F">Sebastien Turgot</COUREUR>
   <COUREUR id="vn" naissance="1984" nat="I">Vincenzo Nibali</COUREUR>
   <COUREUR id="sg" naissance="1980" nat="AUS">Simon Gerrans</COUREUR>
   <COUREUR id="ib" naissance="1977" nat="I">Ivan Basso</COUREUR>
</COUREURS>
<COURSES>
<COURSE nom="TourDesFlandres">
   <PODIUM annee="2011">
      <COUREUR id="nn"/><COUREUR id="sc"/><COUREUR id="fc"/>
   </PODIUM>
   <PODIUM annee="2012">
      <COUREUR id="tb"/><COUREUR id="fp"/><COUREUR id="ab"/>
   </PODIUM>
</COURSE>
<COURSE nom="ParisRoubaix">
   <PODIUM annee="2011">
      <COUREUR id="jv"/><COUREUR id="fc"/><COUREUR id="mt"/>
   </PODIUM>
   <PODIUM annee="2012">
      <COUREUR id="tb"/><COUREUR id="st"/><COUREUR id="ab"/>
   </PODIUM>
</COURSE>
<COURSE nom="MilanSanRemo">
   <PODIUM annee="2012">
      <COUREUR id="sg"/><COUREUR id="fc"/><COUREUR id="vn"/>
   </PODIUM>
</COURSE>
</COURSES>
</CYCLISME>
```

FIGURE 3 – Fichier XML avec des informations sur les podiums des courses cyclistes.

Question 2 Écrivez une expression XPath (aussi simple que possible) qui rend chaque nœud de type `texte` dont la valeur est le nom d'un coureur ayant terminé deuxième dans Paris-Roubaix. Pour le document de la figure 3, la réponse consiste en `Fabian Cancellara` et `Sebastien Turgot`.

.../3

Question 3 Écrivez une expression XPath (aussi simple que possible) qui rend chaque nœud de type `texte` dont la valeur est le nom d'un coureur ayant atteint un podium à la fois en 2011 et en 2012. Pour le document de la figure 3, le seul coureur dans la réponse est `Fabian Cancellara`.

.../3

Question 4 Écrivez une expression XPath (aussi simple que possible) qui rend chaque nœud de type `attribute` dont la valeur est une année où un Italien a atteint le podium de Paris-Roubaix. Pour le document de la figure 3, la seule réponse est `annee="2012"`.

.../3

Question 5 Écrivez une expression XPath (aussi simple que possible) qui rend chaque nœud de type `texte` dont la valeur est le nom d'un coureur italien ayant déjà atteint un podium. La liste doit être sans doublons. Pour le document de la figure 3, la réponse consiste en `Filippo Pozzato`, `Alessandro Ballan` et `Vincenzo Nibali`.

.../3

Question 6 Écrivez un programme XSLT qui génère un document XML affichant les podiums de façon conviviale et en anglais, comme suit. La position des blancs et retours à la ligne n'a pas d'importance. Le programme ne peut pas contenir des `xsl:for-each` or `xsl:if`.

```
<?xml version="1.0" ?>
<CYCLING>
  <TourDesFlandres>
     <PODIUM year="2011"><WINNER nat="B">Nick Nuyens</WINNER>
                         <SECOND nat="F">Sylvain Chavanel</SECOND>
                         <THIRD nat="CH">Fabian Cancellara</THIRD>
     </PODIUM>
     <PODIUM year="2012"><WINNER nat="B">Tom Boonen</WINNER>
                         <SECOND nat="I">Filippo Pozzato</SECOND>
                         <THIRD nat="I">Alessandro Ballan</THIRD>
     </PODIUM>
  </TourDesFlandres>
  <ParisRoubaix>
     <PODIUM year="2011"><WINNER nat="B">Johan Vansummeren</WINNER>
                         <SECOND nat="CH">Fabian Cancellara</SECOND>
                         <THIRD nat="NL">Maarten Tjallingii</THIRD>
     </PODIUM>
     <PODIUM year="2012"><WINNER nat="B">Tom Boonen</WINNER>
                         <SECOND nat="F">Sebastien Turgot</SECOND>
                         <THIRD nat="I">Alessandro Ballan</THIRD>
     </PODIUM>
  </ParisRoubaix>
  <MilanSanRemo>
     <PODIUM year="2012"><WINNER nat="AUS">Simon Gerrans</WINNER>
                         <SECOND nat="CH">Fabian Cancellara</SECOND>
                         <THIRD nat="I">Vincenzo Nibali</THIRD>
     </PODIUM>
  </MilanSanRemo>
</CYCLING>
```

. . ./12

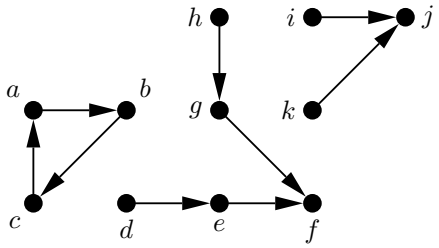
Question 7 Voici une requête qui est l'union de deux requêtes conjonctives. Simplifiez cette requête en s'appuyant sur les théorèmes vus au cours ou expliquez pourquoi aucune simplification n'est possibile.

$$\begin{cases} Answer(x) & \leftarrow & L(\text{“Jean”}, z), L(\text{“Ed”}, z), S(y, z), V(x, y) \\ Answer(u) & \leftarrow & L(\text{“Jean”}, z_1), L(w, z_2), S(y, z_1), S(y, z_2), V(u, y) \end{cases}$$

.../6

Question 8 Un graphe dirigé est encodé en utilisant le prédicat E pour les arêtes. On peut supposer que chaque nœud a une arête entrante ou sortante, et qu'aucun nœud n'a deux (ou plusieurs) arêtes sortantes.

Par exemple, le graphe ci-après est encodé par $\{E(a,b),\ E(b,c),\ E(c,a),\ E(h,g),\ E(g,f),\ E(d,e),\ E(e,f),\ E(i,j),\ E(k,j)\}$.



Un nœud sans arête sortante est appelé *un puit*. Dans l'exemple, les puits sont j et f. Écrivez un programme en datalog$^\neg$ (i.e., datalog avec négation) pour le prédicat idb $Equidistant(x,y)$ qui signifie que x et y sont deux nœuds ayant la même distance à un même puit. La distance entre un nœud x et un puit p est le nombre d'arêtes à traverser pour aller de x à p. S'il n'existe pas de chemin de x à p, cette distance n'est pas définie.

Pour l'exemple, le résultat contient huit atomes : $\{Equidistant(g,e), Equidistant(e,g), Equidistant(d,h), Equidistant(h,d), Equidistant(i,k), Equidistant(k,i)\}$.

$Equidistant(d,h)$ est dans le résultat car les nœuds d et h sont tous les deux à une distance 2 du puit f.

. . . /6

Comme expliqué au cours, datalog$^\neg$ permet d'exprimer l'inégalité $\neq$.

$$
\begin{aligned}
&Noeud(x) \leftarrow E(x,y)\\
&Noeud(y) \leftarrow E(x,y)\\
&NonPuit(x) \leftarrow E(x,y)\\
&Puit(x) \leftarrow Noeud(x), \neg NonPuit(x)\\
&Eqdis(x,y) \leftarrow E(x,p), E(y,p), Puit(p)\\
&Eqdis(x,y) \leftarrow E(x,u), E(y,w), Eqdis(u,w)\\
&Equidistant(x,y) \leftarrow Eqdis(x,y), x \neq y
\end{aligned}
$$

La dernière règle a été ajoutée pour éviter des faits de forme $Equidistant(g,g)$. Noter :

- Pour le graphe $\{E(c,b), E(d,b), E(b,a)\}$, le programme calcule $Puit(a)$, $Eqdis(b,b)$, $Eqdis(c,c)$, $Eqdis(c,d)$, $Eqdis(d,c)$, $Eqdis(d,d)$. On trouve $Equidistant(c,d)$ et $Equidistant(d,c)$.
- Évidemment, il est faux d'écrire $Puit(x) \leftarrow E(y,x), \neg E(x,z)$.

Question 9 On considère les trois relations edb suivantes :
– $\mathit{Frequente}(x, y)$ signifie que x est un buveur qui fréquente le bar y. Par exemple, $\mathit{Frequente}(\text{Ed}, \text{Calypso})$.
– $\mathit{Sert}(y, z)$ signifie que y un bar qui sert la bière z. Par exemple, $\mathit{Sert}(\text{Calypso}, \text{Orval})$.
– $\mathit{Aime}(x, z)$ signifie que x est un buveur qui aime la bière z. Par exemple, $\mathit{Aime}(\text{Ed}, \text{Orval})$.
On suppose que chaque buveur aime au moins une bière et fréquente au moins un bar, que chaque bar sert au moins une bière et que chaque bière est servie dans au moins un bar. Exprimer en datalog$^\neg$ le prédicat idb $\mathit{Exigeant}(x)$ qui signifie que x est un buveur qui ne fréquente que des bars qui servent au moins une bière qu'il aime.

…/6

$\mathit{Frequentable}(y, x)$ est vrai si y est un bar qui sert au moins une bière que le buveur x aime.

$$\mathit{Frequentable}(y, x) \leftarrow \mathit{Sert}(y, z), \mathit{Aime}(x, z)$$
$$\mathit{NonExigeant}(x) \leftarrow \mathit{Frequente}(x, y), \neg \mathit{Frequentable}(y, x)$$
$$\mathit{Exigeant}(x) \leftarrow \mathit{Frequente}(x, y), \neg \mathit{NonExigeant}(x)$$

Question 10 Détaillez pourquoi le prédicat *Exigeant* de la question 9 ne peut pas être calculé en datalog sans négation.

.../3

Question 11 Dessiner le *dependency graph* pour le programme datalog$^\neg$ suivant. Donner le résultat de ce programme pour la base de données $\{E(a,b), E(b,c), E(c,a), E(c,c), E(a,d), E(d,a)\}$. Détailler les calculs.

$$\begin{cases} S(x) & \leftarrow & E(x,x) \\ T(x,z) & \leftarrow & T(x,y), E(y,z) \\ R(x,z) & \leftarrow & R(x,y), E(y,z), \neg S(z) \\ R(x,y) & \leftarrow & E(x,y), \neg S(x), \neg S(y) \\ T(y,x) & \leftarrow & E(y,x) \\ Answer(x) & \leftarrow & T(x,x), \neg R(x,x) \end{cases}$$

.../6

Question 12 Est-ce que le schéma $\mathbf{S} = \{R[A, B, C], S[B, C, D], T[B, F], U[C, D, E], V[D, E, G]\}$ a un *full reducer* ?
- ☐ oui
- ☐ non
- ☐ je ne sais pas

Détaillez le raisonnement qui mène à cette réponse.

.../6