Département Informatique et Statistique, ICOM, Université Lumière Lyon 2
M1 Informatique – Year 2015-2016
Semi-structured data & XML – Exam
J. Darmont (*http://eric.univ-lyon2.fr/~jdarmont/*), 26/04/2016

Duration: 2 hours – Documents allowed

Exercise #1: XML schemas (10 points)

Translate the following DTD into an XML Schema.

```
<!ELEMENT data_warehouse (fact)*>
     <!ATTLIST data_warehouse wid ID #REQUIRED>
     <!ELEMENT fact (dimension+, measure+)>
          <!ATTLIST fact fid ID #REQUIRED>
          <!ELEMENT dimension (hierarchy_level)+>
               <!ATTLIST dimension did ID #REQUIRED>
               <!ELEMENT hierarchy_level EMPTY>
                    <!ATTLIST hierarchy_level lid ID #REQUIRED rollup IDREF "ALL">
          <!ELEMENT measure EMPTY>
               <!ATTLIST measure mid ID #REQUIRED>
```

Exercise #2: XQuery (10 points)

Let us consider the three XML documents: `articles.xml`, `images.xml` and `inventory.xml`, which are related to the industrial heritage (mostly textile) of the Lille area. Inventory is a list of buildings; images a list of pictures; and articles a list of press articles referring to pictures and/or buildings. The structure of these XML documents is depicted below.

articles.xml	images.xml	inventory.xml
corpus	List	inventory
document+	record+	record+
description	code-pic	code-bui
author+	description	description
title	title	owner
subtitle?	date	address
date	keyword*	building-type
language	geo-keyword*	roof-type
text	origin	floors
references		condition
ref-pic*		
ref-bui*		
evaluation		
grade		

Formulate the following XQueries. Use XPath whenever possible.

1. Distinct addresses of buildings.
2. Articles whose description includes a subtitle.
3. Codes of images taken before year 2000 and whose description includes a Factory keyword.
4. Title and author(s) of the best-graded article.

5. Image information formatted as follows: `<image code="C" status="S" />`; where *C* is the image's code-pic and *S* is "old" if the picture was taken before 2000 and "recent" otherwise.
6. Picture and inventory codes of images and buildings that share identical geo-keyword and address. Also indicate either the geo-keyword or address in the result.
7. For each image, provide its title (within an image element) and the title of the article it appears in (within an article element).
8. Average number of floors per address.
9. Total number of references (both image and building references) per article (title).
10. Number of articles referring to each building. Indicate code-bui and the whole building description in the result.

Solution of Exercise #1

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
     <xsd:element name="data_warehouse">
          <xsd:complexType>
               <xsd:sequence minOccurs="0" maxOccurs="unbounded">
                   <xsd:element ref="fact" />
               </xsd:sequence>
               <xsd:attribute name="wid" type="xsd:ID" use="required" />
          </xsd:complexType>
     </xsd:element>
     <xsd:element name="fact">
          <xsd:complexType>
               <xsd:sequence>
                   <xsd:element ref="dimension" maxOccurs="unbounded" />
                   <xsd:element ref="measure" maxOccurs="unbounded" />
               </xsd:sequence>
               <xsd:attribute name="fid" type="xsd:ID" use="required" />
          </xsd:complexType>
     </xsd:element>
     <xsd:element name="dimension">
          <xsd:complexType>
               <xsd:sequence>
                   <xsd:element ref="hierarchy_level" maxOccurs="unbounded" />
               </xsd:sequence>
               <xsd:attribute name="did" type="xsd:ID" use="required" />
          </xsd:complexType>
     </xsd:element>
     <xsd:element name="hierarchy_level">
          <xsd:complexType>
               <xsd:attribute name="lid" type="xsd:ID" use="required" />
               <xsd:attribute name="rollup" type="xsd:IDREF" default="ALL" />
          </xsd:complexType>
     </xsd:element>
     <xsd:element name="measure">
          <xsd:complexType>
               <xsd:attribute name="mid" type="xsd:ID" use="required" />
          </xsd:complexType>
     </xsd:element>
</xsd:schema>
```

Solution of Exercise #2

```
(: 1 :)
distinct-values(//address)

(: 2 :)
//document[description/subtitle]

(: 3 :)
//record[year-from-date(description/date) < 2000 and
         description/keyword = "Factory"]/code-pic

(: 4 :)
(for $a in //document
order by number($a/evaluation/grade) descending
return <result>
          {$a/description/title}
          {$a/description/author}
       </result>)[1]
```



```
(: 5 :)
for $i in /list/record
return if (year-from-date($i/description/date) < 2000)
       then <image code="{data($i/code-pic)}" status="old" />
       else <image code="{data($i/code-pic)}" status="recent" />

(: 6 :)
for $i in /list/record,                  for $i in /list/record,
    $b in /inventory/record                  $b in /inventory/record[
where $i//geo-keyword = $b//address                 description/address =
return <result>                                     $i/description/geo-keyword]
         {$i/code-pic}                   return <result>
         {$b/code-bui}                             {$i/code-pic}
         {$i//geo-keyword}                         {$b/code-bui}
       </result>                                   {$i//geo-keyword}
                                                 </result>

(: 7 :)
for $i in /list/record,                  for $i in /list/record,
    $a in //document                         $a in //document[$i/code-pic =
where $a//ref-pic = $i/code-pic                             references/ref-pic]
return                                   return
 <result>                                 <result>
  <image>{data($i//title)}</image>         <image>{data($i//title)}</image>
  <article>{data($a//title)}</article>     <article>{data($a//title)}</article>
 </result>                                </result>

(: 8 :)
for $b in /inventory//description
group by $a := $b/address
return <building address="{$a}"
                nbfloors="{avg($b/floors)}" />

(: 9 :)
for $a in //document
group by $t := $a//title
return <article title="{$t}"
               nbreferences="{count($a//ref-pic) + count($a//ref-bui)}" />

(: 10 :)
for $b in /inventory/record,             for $b in /inventory/record,
    $a in //document                         $a in //document[$b/code-bui =
where $a//ref-bui = $b/code-bui                             references/ref-bui]
group by $c := $b/code-bui               group by $c := $b/code-bui
return                                   return
 <result>                                 <result>
  {$b/code-bui}                            {$b/code-bui}
  {$b/description}                         {$b/description}
  <nbarticles>{count($a)}</nbarticles>     <nbarticles>{count($a)}</nbarticles>
 </result>                                </result>
```