Klausur "Semistructured Data and XML"
Summer Term 2022
Prof. Dr. Wolfgang May
22. August 2023, 11:00–13:00
Working Time: 120 Minutes
(carried out as a computer-based ILIAS exam)

Vorname:

Nachname:

Matrikelnummer:

Setting: The usage of saxon (with the aliases saxonValid, saxonXQ, and saxonXSL defined as in the course) and xmllint (validation error messages are better with xmllint) is recommended. Web access (e.g. for XPath/XQuery and XSLT documentation) is allowed. It was also recommended to have the slides, and a condensed self-prepared "cheat sheet" (preparation of a cheat sheet is a very effective way to work through the materials).
Answers might be given in English or German (most answers are program code anyway). In the text, the german translation is sometimes given in parentheses.
Give *all* answers via the ILIAS system.
Like in a "paper exam", also solutions that do maybe not work (or do not work completely) can be delivered and will be graded with appropriate partial points.
For **passing** the exam, **50** points are sufficient.

	Max. Punkte	Schätzung für "4"
Aufgabe 1 (XML)	20	15
Aufgabe 2 (DTD)	15	10
Aufgabe 3 (XPath (a))	3	3
Aufgabe 4 (XPath (b))	4	3
Aufgabe 5 (XPath/XQuery (c))	4	3
Aufgabe 6 (XPath/XQuery (d))	6	4
Aufgabe 7 (XPath/XQuery (e))	6	2
Aufgabe 8 (XPath/XQuery (f))	6	5
Aufgabe 9 (XPath/XQuery (g))	8	4
Aufgabe 10 (Data Integration and XPath/XQuery)	14	2
Aufgabe 11 (XSLT)	14	4
Summe	100	55

Note:

Project: A Database about Painters and Paintings

The scenario is about a cultural database about painters and paintings.

1. Painters have a name. There are no two painters with the same name. Additionally, their birthdate, their death date (if they already died) and their birthplace is stored.

   For each painter, also some important cities where he lived for a certain period of his life (here, only the years are considered, not exact dates) are stored.

2. For each city mentioned in the database, also the country is stored because there might be cities with the same name in different countries. Inside each country, the city names are assumed to be unique.

   Paul Cézanne was born on January 19, 1839 in *Aix en Provence* in *France*, where he also lived until his death on October 22, 1906.

   Vincent van Gogh was born on March 30, 1853 in *Zundert* in the *Netherlands*, and died on July 29, 1890 in *Paris* (*France*). He lived in *Paris* from 1885 until 1887, and he lived in *Arles* (*France*) in 1888 and 1889.

   Henri de Toulouse-Lautrec was born on November 24, 1864 in *Albi (France)* and died on September 9, 1901. He lived in *Paris* from 1884 until 1901.

3. Sometimes, (geographical, not political) *regions* play an important role in paintings. For each such region, it is stored which places (cities) are located there. There are no two regions with the same name.

   The cities *Aix en Provence*, *Arles* and *Tarascon* are located in the *Provence*.

4. Museums have a (unique) name, and are located in a city, e.g., the *Musée d'Orsay* is located in *Paris*.

5. For each painting, its title is stored, and who has painted it (note that multiple painters might have drawn paintings that more or less accidentally have the same title). For every painting, if known, it is also stored, when it has been created (only the year), and if applicable, in which museum it is presented (some paintings are not in any museum, but in private property or even lost).

   Vincent van Gogh created the painting *"The Potato Eaters"* in 1885, which is now in the *Van Gogh Museum* in *Amsterdam (Netherlands)*.

6. Some paintings show one ore more (prominent) persons, e.g., painters (sometimes the creator himself), but also other renowned persons. This is also stored:

   Paul Cezanne created a painting *"Paul Alexis reading a Manuscript to Emile Zola"* in 1870, which shows the two writers *Paul Alexis* and *Emile Zola*. The picture is exhibited in the *São Paulo Museum of Art* (in *São Paulo, Brazil*).

   Henri de Toulouse-Lautrec created a painting in 1887 with the title *"Vincent van Gogh"*, which shows van Gogh. The painting is exhibited in the *Van Gogh Museum* in *Amsterdam*.

   In 1890, *Henri de Toulouse-Lautrec* created the painting *"Jane Avril dancing"* (depicting the Moulin Rouge dancer Jane Avril) which is shown in the *Musée d'Orsay*.

   Vincent van Gogh created a *"Self-Portrait with Straw Hat"* in 1887, which is shown in the *National Gallery of Art (NGA)* in *Washington (USA)*. In 1889, he created another *"Self Portrait"* which is shown in the *Musée d'Orsay*.

7. Some paintings show typical landscape of a certain region, also this is stored:

   Vincent van Gogh painted *"Starry Night Over the Rhône"* in 1888, which is also shown in the *Musée d'Orsay*. The painting shows the landscape of the *Provence*. Also, in 1888, he created *"On the Road to Tarascon"*, which is lost (most probably destroyed in WW2) which also shows the landscape of the *Provence*.

   Paul Cezanne created the painting *"Mont Sainte-Victoire"* in 1888, which also shows the landscape of the *Provence*, and which is exhibited in the *Stedelijk Museum* which is located in *Amsterdam*. Later in 1890, he created the painting *"Mont Sainte-Victoire seen from Gardanne"*, which also shows the landscape of the *Provence*, and which is exhibited in the *NGA* in *Washington*.

Exercise 1 (XML [20 Points])
Design an XML structure (use the frame given in file `exam.xml`) and fill it with some sample data (e.g. with some of the example data given in the text).

Lösung

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE arts SYSTEM "exam.dtd">
<arts>
 <city id="aix" name="Aix en Provence" country="F"/>
 <city id="arles" name="Arles" country="F"/>
 <city id="tarascon" name="Tarascon" country="F"/>
 <city id="albi" name="Albi" country="F"/>
 <city id="paris" name="Paris" country="F"/>
 <city id="ams" name="Amsterdam" country="NL"/>
 <city id="grzundert" name="Zundert" country="NL"/>
 <city id="sp" name="Sao Paulo" country="BR"/>
 <city id="wash" name="Washington" country="USA"/>
 <landscape id="prov" name="Provence" cities="arles aix tarascon"/>
 <painter id="cez" name="Paul Cezanne"
     birthdate="1839-01-19" birthplace="aix" died="1906-10-22">
   <lived city="aix" from="1839" until="1906"/>
   <painting title="Alexis reading to Zola" year="1870"
      shows="zola alexis" museum="spart"/>
   <painting title="Mont Sainte-Victoire" year="1888"
      shows="prov" museum="sted"/>
   <painting title="Mont Sainte-Victoire seen from Gardanne" year="1890"
      shows="prov" museum="nga"/>
 </painter>
 <painter id="vg" name="Vincent van Gogh"
     birthdate="1853-03-30" birthplace="grzundert" died="1890-07-29">
   <lived city="paris" from="1885" until="1887"/>
   <lived city="arles" from="1888" until="1889"/>
   <painting title="Potato Eaters" year="1885" museum="vgmus"/>
   <painting title="Self Portrait with Straw Hat" year="1887"
      shows="vg" museum="nga"/>
   <painting title="Starry Night over the Rhone" year="1888"
      shows="prov" museum="orsay"/>
   <painting title="Road to Tarascon" year="1888" shows="prov"/>
   <painting title="Self Portrait" year="1889" shows="vg" museum="orsay"/>
 </painter>
 <painter id="tl" name="Henri de Toulouse-Lautrec"
     birthdate="1864-11-24" birthplace="albi" died="1901-09-09">
   <lived city="paris" from="1884" until="1901"/>
   <painting title="Vincent van Gogh" year="1887"
      shows="vg" museum="vgmus"/>
   <painting title="Jane Avril dancing" year="1890"
      shows="avril" museum="orsay"/>
 </painter>
 <museum id="nga" name="National Gallery of Arts" city="wash"/>
 <museum id="vgmus" name="Van Gogh Museum" city="ams"/>
 <museum id="sted" name="Stedelijk Museum" city="ams"/>
 <museum id="orsay" name="Musee d'Orsay" city="paris"/>
 <museum id="spart" name="Sao Paulo Museum of Art" city="sp"/>
 <otherperson id="zola" name="Emile Zola" comment="writer"/>
 <otherperson id="alexis" name="Paul Alexis" comment="writer"/>
 <otherperson id="avril" name="Jane Avril" comment="dancer"/>
</arts>
```

- note: There are many different possible structures. It is also possible to have "person" in general with optional contents, and painters are persons with "painting" subelements.

Exercise 2 (DTD [15 Points])
Give the DTD for your document developed in Exercise 1, use the file `exam.dtd`.
Use one of the calls

```
xmllint -loaddtd -valid --noblanks -noout exam.xml
saxonValid.bat -s:exam.xml
```

for validating it (note that xmllint provides better error messages).
Copy-and-paste the DTD from the file `exam.dtd` afterwards here:

Lösung

```
<!ELEMENT arts (city*, landscape*, painter*, museum*, otherperson*)>
<!ELEMENT city EMPTY>
<!ATTLIST city id ID #REQUIRED
               name CDATA #REQUIRED
               country CDATA #REQUIRED>
<!ELEMENT landscape EMPTY>
<!ATTLIST landscape id ID #REQUIRED
                   name CDATA #REQUIRED
                   cities IDREFS #IMPLIED>
<!ELEMENT painter (lived*, painting+)>
<!ATTLIST painter id ID #REQUIRED
                  name CDATA #REQUIRED
                  birthdate CDATA #REQUIRED
                  birthplace IDREF #REQUIRED
                  died CDATA #IMPLIED>
<!ELEMENT lived EMPTY>
<!ATTLIST lived city IDREF #REQUIRED
                from CDATA #REQUIRED
                until CDATA #REQUIRED>
<!ELEMENT painting EMPTY>
<!ATTLIST painting title CDATA #REQUIRED
                   year  CDATA #IMPLIED
                   museum IDREF #IMPLIED
                   shows IDREFS #IMPLIED>
<!ELEMENT museum EMPTY>
<!ATTLIST museum id ID #REQUIRED
                 name CDATA #REQUIRED
                 city IDREF #REQUIRED>
<!ELEMENT otherperson EMPTY>
<!ATTLIST otherperson id ID #REQUIRED
                 name CDATA #REQUIRED
                 comment CDATA #IMPLIED>
```

Exercise 3 (XPath (a) [3 Points])
Use your `exam.xml` XML file as a basis for solving this and the following exercises.
None of the results should contain duplicates.

Give an XPath query or an XQuery query that returns the names of those painters from whom there is some painting shown in a museum in Brazil.
Write the query string in the file `query1.xq` and call it with

```
saxonXQ.bat -s:exam.xml query1.xq
```

Copy-and-paste the query from query1.xq afterwards here:

Lösung

```
//painter[painting/id(@museum)/id(@city)/@country='BR']/@name/string()
(: Cezanne :)
```

Exercise 4 (XPath (b) [4 Points])
Give an XPath query or an XQuery query that returns the *names* of those painters where *none* of their paintings (stored in this quite incomplete database) is presented in a museum in their country of birth.
Write the XPath query string in the file `query2.xq` and call it with

```
saxonXQ.bat -s:exam.xml query2.xq
```

Copy-and-paste the query from query2.xq afterwards here:

Lösung

```
//painter[not (painting/id(@museum)/id(@city)/@country
                = id(@birthplace)/@country)]
 /@name/string()
(: Cezanne :)
```

Exercise 5 (XPath/XQuery (c) [4 Points])
Give an XPath or XQuery query that yields the titles of all paintings that are exhibited in a museum that is located in the same country as the landscape shown on the painting.
Copy-and-paste the query from query3.xq afterwards here:

Lösung

```
//painting[ id(@shows)/id(@cities)/@country
            = id(@museum)/id(@city)/@country ]
/@title/string()
 (: Starry Night over the Rhone :)
```

Exercise 6 (XPath/XQuery (d) [6 Points])
Give an XQuery query that outputs for each city, how many paintings that have been created between 1880 and 1889 (inclusive) are exhibited in the museums in that city. Output should be ordered descendingly by the number of paintings in the form

```
<city name="..." number="..."/>
```

Copy-and-paste the query from query4.xq afterwards here:

Lösung

```
for $c in //city
let $paintings := //painting[id(@museum)/@city = $c/@id and
        @year>=1880 and @year<=1889]
let $num := count($paintings)
order by $num descending
return <city name="{$c/@name}" number="{$num}"/>
(: AMS 3 PAR 2 WASH 1 SP 0
   6 results, "The road to Tarascon" is lost   :)
```

Exercise 7 (XPath/XQuery (e) [6 Points])
Give an XQuery query that yields the names of those museums that host at least one painting from *each of the painters* who have created some painting that shows the Provence landscape.

Copy-and-paste the query from query5.xq afterwards here:

Lösung

```
for $m in //museum
where every $p in //painter[painting/id(@shows)/@name = "Provence"]
satisfies $p/painting/@museum = $m/@id
return $m/@name/string()
  (: NGA :)
```

Exercise 8 (XPath/XQuery (f) [6 Points])
Give an XQuery query that for each painting that shows some painter, outputs the title of the painting, the name of the creator, and the name of the depicted painter. For each such painting, the result should be of the form

```
<result painting="..." creator="..." depicted="..."/>
```

Copy-and-paste the query from query6.xq afterwards here:

Lösung

```
for $p in //painting[id(@shows)/name()='painter']
return <result
  painting="{$p/@title}"
  creator="{$p/parent::*/@name}"
  depicted="{id($p/@shows)[name()='painter']/@name}"/>
 (: 3 outputs :)
```

Exercise 9 (XPath/XQuery (g) [8 Points])
Give an XQuery query that outputs the name of the painter who reached the highest age of the painters stored in the database.

Copy-and-paste the query from query7.xq afterwards here:

Lösung

```
 (: this variant even considers painters that are still alive :)
 let $maxage := max(for $p in //painter
 return (:  xs:date($p/@died)  :)
     (if ($p/@died) then xs:date($p/@died) else fn:current-date())
      - xs:date($p/@birthdate))
    let $oldest := //painter[
      (if (@died) then xs:date(@died) else fn:current-date())
      - xs:date(@birthdate) = $maxage]
return ($oldest/@name/string(), $maxage)
(: Cezanne, outputs also the age: P24747D  - means  24747 days, 67.xx years :)
```

```
 //painter[xs:date(@died) - xs:date(@birthdate)
           = max(//painter/(xs:date(@died) - xs:date(@birthdate)))]
   /@name/string()
```

Exercise 10 (Data Integration and XPath/XQuery [14 Points])
This exercise combines the data in your exam.xml file with the data in mondial (use your local mondial.xml or the one at `http://www.dbis.informatik.uni-goettingen.de/Mondial/mondial-europe.xml`.

Assume here that the name of each used city is unique in its country, so one does not need to consider the province names.

Give an XQuery query (query-int.xq) that returns a modified document that follows basically the structure of mondial.xml and adds the museums into the city elements, according to the following DTD and instructions:

```
<!ELEMENT mondial (country*)>     # only countries where at least one museum is l
<ELEMENT country (name, city*)>   # only cities where at least one museum is loca
<!ELEMENT city (name, population?, museum*)>   # the most recent population, if a
<!ELEMENT population (#PCDATA)>
<!ELEMENT museum (name, painting*)>
<!ELEMENT painting (painter, title)>
<!ELEMENT painter (#PCDATA)>
<!ELEMENT title (#PCDATA)>
```

Copy-and-paste the query from query-int.xq afterwards here:

Lösung

```
<mondial>
{ for $country in doc("http://www.dbis.informatik.uni-goettingen.de/Mondial/mondial.xml"
  let $muscities :=
    for $city in $country//city
     where some $muscity in //city
       satisfies ($country/@car_code = $muscity/@country
                  and $city/name = $muscity/@name)
     return $city
  return if ($muscities)
    then <country>
     { $country/name[1] }
     { for $muscity in $muscities
       return
       <city>
        { $muscity/name[1] }
        <population> {$muscity/population[last()]/string()} </population>
        { for $m in //museum[id(@city)/@name = $muscity/name
                             and id(@city)/@country = $country/@car_code]
            (: care for cities with same name in different countries :)
          return
          <museum>
            <name> {$m/@name/string()} </name>
            { for $p in //painting[@museum = $m/@id]
              return
              <painting>
                <painter> {$p/parent::painter/@name/string()} </painter>
                <title> {$p/@title/string()} </title>
              </painting>   }
          </museum>
        }
      </city>
    }
    </country>
    else ()
  }
</mondial>
```

Exercise 11 (XSLT [14 Points])
Extend the given XSL stylesheet frame `exam.xsl` to an XSLT stylesheet that generates an HTML page that contains one table for each museum as follows:
The name of the museum, and the name of the city where it is located. Below this, for each of the painters (beginning with those that were born in the country where the museum is located) that are presented in this museum, the name of the painter and a listing of his paintings in this museum (as HTML list or nested table) should follow.
Use the following call to execute it:

```
saxonXSL.bat -s:exam.xml exam.xsl
```

Copy-and-paste the XSLT stylesheet from exam.xsl afterwards here:

Lösung

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  version="2.0">

  <xsl:template match="arts">
    <html><body>
    <xsl:apply-templates select="museum"/>
    </body></html>
  </xsl:template>

  <xsl:template match="museum">
   <table>
    <tr><td colspan="5">
      <xsl:value-of select="@name"/> (<xsl:value-of select="@city/@name"/>)
    </td></tr>
    <xsl:apply-templates
    select="//painter[painting[@museum=current()/@id] and
              id(@birthplace)/@country = current()/id(@city)/@country ]">
      <xsl:with-param name="museum" select="current()/@id"/>
    </xsl:apply-templates>
    <xsl:apply-templates
      select="//painter[painting[@museum=current()/@id] and
              id(@birthplace)/@country != current()/id(@city)/@country]">
      <xsl:with-param name="museum" select="current()/@id"/>
    </xsl:apply-templates>
   </table>
  </xsl:template>

  <xsl:template match="painter">
    <xsl:param name="museum"/>
    <tr> <td><xsl:value-of select="@name"/>: </td> </tr>
    <xsl:for-each select="painting[@museum=$museum]">
      <tr><td><xsl:value-of select="@title"/></td></tr>
    </xsl:for-each>
  </xsl:template>
</xsl:stylesheet>
```

The following frames can be used:

- Frame for XML file `exam.xml`:

  ```
  <?xml version="1.0" encoding="UTF-8"?>
  <!DOCTYPE put-name-here SYSTEM "exam.dtd">
    to be extended here
  ```

- Frame for XML stylesheet `exam.xsl`:

  ```
  <xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  ```

```
  version="2.0">
  to be extended here
</xsl:stylesheet>
```