Big Data – Exercises
Fall 2016 – Week 8 – ETH Zurich

1. Information Sets

Build an information set tree of the following XML documents.

You can provide only the following set of properties: (parent/children connection, attributes and namespaces)

```xml
<?xml version="1.0"?>
<catalog>
  <!-- Start book list to be defined -->
  <Book id='bk101'>
    <author>Gambardella, Matthew</author>
    <title>XML Developer's Guide</title>
    <genre>Computer</genre>
    <price>44.95</price>
    <publish_date version='hard' version2='soft'>2000-10-01</publish_date>
  </Book>
</catalog>

<?xml version="1.0"?>
<!DOCTYPE eth]
  <date>13.11.2006</date>
  <president number="1">Empty</president>
  <xmldb:Rektor>Name 2</xmldb:Rektor>
</eth>
```
Doc 1:

Solution

Doc 2:
2. XDM: XPath and XQuery Data Model

2.1. Name several differences between XDM and Information Sets

2.2. Which of the following XDM sequences are equivalent:

1. (1, 2, 3, 4, 5, <a/>, <foo><bar/</foo>)
2. (1, (2, 3), 4, (5, <a/>, <foo><bar/</foo>))
3. (1, <foo><bar/></foo>, 2, 3, 4, 5, <a/>)
4. (<a/>)
5. <a/>
6. ()
7. (1, 1, 1)
8. ((),(),())
9. (1)

Solution

2.1.

• the XDM supports simple and complex types, which the information set does not (very basic and limited attribute types only)
• XDM is sequence-based, while the information set only models the tree for one document. Then, the XDM can directly be processed by querying languages like XPath and XQuery.
• The XDM supports both validated and non-validated (that is, only well-formed) data. The information set doesn't support validation (it's the PSVI that extends the information set to do so). The XDM also supports in subsequent versions other items (functions, maps, arrays).

2.2.

• 1 and 2 are equivalent
• 4 and 5 are equivalent
• 6 and 8 are equivalent

3. Data types (from XML Schema and XML)

Create a one-to-one mapping between the following set of XDM sequences. Use each XDM sequence and XML schema type (with cardinality) only once:

(1, 2, 3, 4, 5)
(<foo/>, <foo/>)
("foo", "bar")
("foo")
()
and the following set of XML schema types and cardinalities:

\[
\begin{align*}
\text{x}s & : \text{integer} + \\
\text{x}s & : \text{integer} * \\
\text{x}s & : \text{string} ? \\
\text{x}s & : \text{string} * \\
\text{x}s & : \text{element} () +
\end{align*}
\]

**Solution**

- (1, 2, 3, 4, 5) is of type \text{x}s:integer+
- \(<\text{foo}/>, <\text{foo}/>\) is of type \text{element}()+
- \("\text{foo}\", \"\text{bar}\") is of type \text{x}s:string*
- \("\text{foo}\") is of type \text{x}s:string?
- () is of type \text{x}s:integer*

### 4. DTD validation

4.1. Does DTD support documents that have namespaces?

4.2. Does DTD has a type system?

4.3. What is the difference between CDATA and PCDATA?

4.4. Is the following XML document valid under the given DTD. If not modify the document so that it is a valid document. Also modify the DTD so that it validates the original document.

```xml
<?xml version="1.0"?>
<!DOCTYPE catalog SYSTEM "catalog.dtd">
<catalog>
  <!-- Start book list to be defined -->
  <Book id='101'>
    <author>Gambardella, Matthew</author>
    <title>XML Developer's Guide</title>
    <genre>Computer</genre>
    <price>44.95</price>
    <publish_date version='hard' version2='soft'>2000-10-01</publish_date>
  </Book>
</catalog>
```
Solution

4.1. Yes but won’t actually understand them and will treat xmlns:... as regular attributes.

4.2. Yes but a very basic one, doesn’t understand integers, dates, booleans, etc

4.3. PCDATA means "parsed character data". It means that this character data is to be parsed. In particular:

- Entity references (< > ' and &) will be resolved (to < > ' and & respectively), as well as any additional entities defined in the DTD.
- It may not contain any unencoded < or & characters, because they would be confused with an opening tag or an entity reference. In the DTD, PCDATA is used to say that an element may only contain parsed character data, without child elements.

CDATA is more confusing, because it can have several meanings in the XML world:

- In the DTD, it is used to give the most general type for an attribute: an attribute of type CDATA may contain any attribute value. Note, however, that entity references are resolved, and that & and < must be encoded as well. In addition, the single quote ’ must be encoded to ’ if the attribute value is single-quoted, and the double quote " must be encoded to " if the attribute value of double-quoted.
- Where PCDATA is expected in an element, one can explicitly use a CDATA construct to escape the special XML characters like < or >, which will not be recognized as markup. The only sequence recognized as markup in a CDATA section is ]]>>, which is interpreted as the end of the CDATA section.

<![CDATA[
if (a<2) { // notice the use of < without needing to encode it as &lt;
    Writeln("The number is too low");
}
]]>
4.4.

- title must come before author
- id must not start with an integer
- genre can't be a child of Book
- attribute currency is required and must be present in price
- version2 is an undeclared attribute
- book must contain a publisher

```xml
<?xml version="1.0"?>
<!DOCTYPE catalog SYSTEM "catalog.dtd">
<catalog>
  <!-- Start book list to be defined -->
  <Book id='bk101'>
    <title>XML Developer's Guide</title>
    <author>Gambardella, Matthew</author>
    <price currency="chf">44.95</price>
    <publish_date version='hard'>2000-10-01</publish_date>
    <publisher>Springer</publisher>
  </Book>
</catalog>
```

<xml version="1.0" encoding="UTF-8"?>
<!ELEMENT catalog (Book+) >
<!ELEMENT Book
( author, title, genre, price, publish_date, publisher?) >
<!ATTLIST Book id CDATA #REQUIRED>
<!ELEMENT author (#PCDATA)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT genre (#PCDATA)>
<!ATTLIST title lang CDATA 'en'>
<!ELEMENT price (#PCDATA)>
<!ATTLIST price currency CDATA #IMPLIED>
<!ELEMENT publish_date (#PCDATA)>
<!ATTLIST publish_date version CDATA #IMPLIED>
<!ATTLIST publish_date version2 CDATA #IMPLIED>
<!ELEMENT publisher (#PCDATA) >

5. XML Schema

Create a possible XML schema which validates the following document
Solution
### 6. Document Validation

Connect each document type with the type of schema that can be used to validate it.

<table>
<thead>
<tr>
<th>Document Type</th>
<th>Schema Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>XML</td>
<td>XML Schema</td>
</tr>
<tr>
<td>JSON</td>
<td>DTD</td>
</tr>
<tr>
<td>Protocol buffers</td>
<td>Schematron</td>
</tr>
<tr>
<td>XHTML</td>
<td>RelaxNG</td>
</tr>
<tr>
<td></td>
<td>JSON Schema</td>
</tr>
<tr>
<td></td>
<td>JSound</td>
</tr>
<tr>
<td></td>
<td>Kwalify</td>
</tr>
<tr>
<td></td>
<td>The XML Schema for XHTML document</td>
</tr>
<tr>
<td></td>
<td>proto3</td>
</tr>
<tr>
<td></td>
<td>No schema at all</td>
</tr>
</tbody>
</table>

### Solution

- XML is validated with XML Schema, DTD, Schematron, RelaxNG, No schema at all
- JSON is validated with JSON Schema, JSound, Kwalify, No Schema ta all
- Protocol buffers is validated with Protocol buffer schema language only
- XHTML is validated with XHTML XML Schema