Universität Freiburg
Institut für Informatik
Prof. Dr. Peter Fischer
Lecuture XML and Databases
Winter Semester 2011/12

18.11.2011

Solution Sheet 3

XML Data Modelling, XML Schema

Exercise 1: XML Data

1.1. Here is an example:

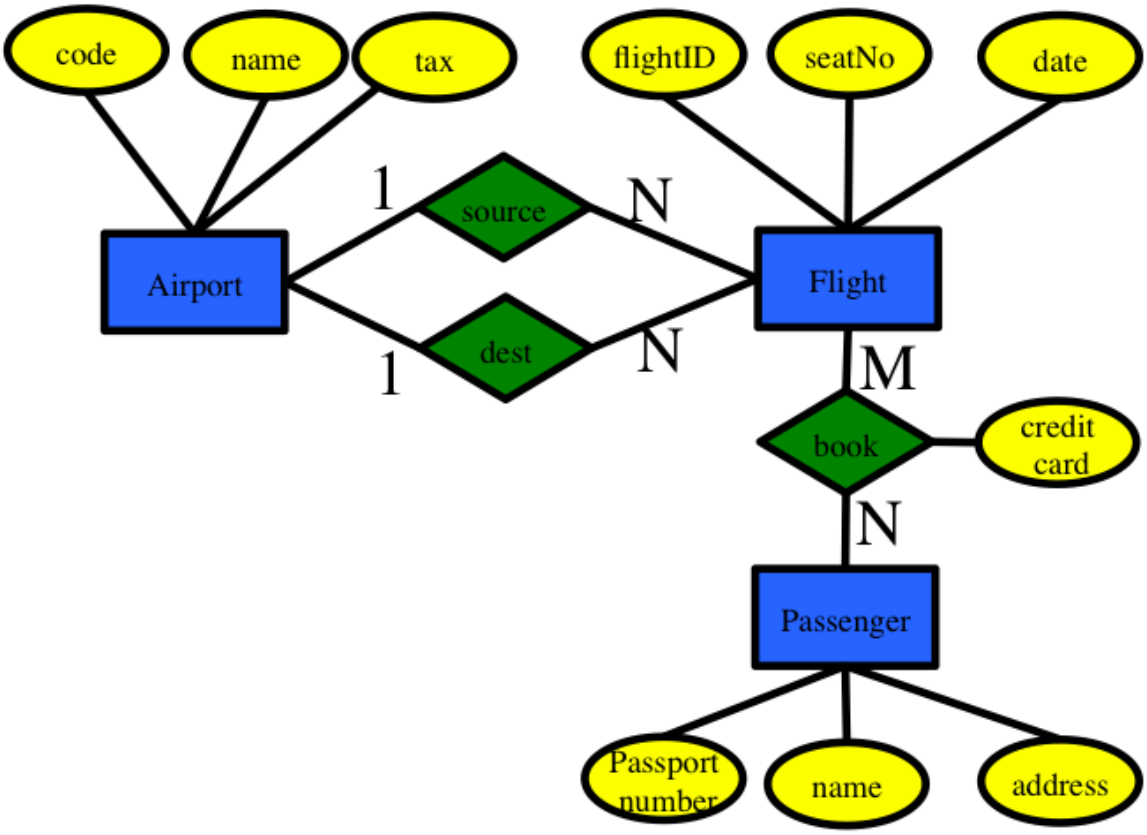
```
<?xml version="1.0" encoding="UTF-8"?>
<doc>
    <Airport airId="LHR">
        <name>London Heathrow</name>
        <tax>100</tax>
    </Airport>

    <Airport airId="ZRH">
        <name>Zurich</name>
        <tax>150</tax>
    </Airport>

    <Flight flightId="LX123">
        <seats>100</seats>
        <date>2005-08-09</date>
        <source>LHR</source>
        <destination>ZRH</destination>
    </Flight>

    <Passenger>
        <name>Student</name>
        <passportnumber>123456</passportnumber>
        <address>Univstr. 6</address>
    </Passenger>

    <Reservation>
        <flightRef>LX123</flightRef>
        <passRef>123456</passRef>
        <creditCard>1234 5678</creditCard>
    </Reservation>
</doc>
```

1.2. By using a DTD or a schema!

1.3. Do it!

Exercise 2: XML Schema Modelling

2.1. Here is a possible Entity-Relationship Model:



2.2. This file contains several inconsistencies which makes it difficult to process. Although it is correct XML:

- element names are inconsistent (passenger, Passenger)
- sometimes the same information is expressed using attributes, and sometimes as sub-elements (flightId)
- redundancy: Flight Information appears as both a standalone element and inside the reservation of a passenger; passenger information appears standalone, and inside another reservation.
- inconsistency: sometimes flights are referenced (using their flightID, sometimes they are copied completely)
- extraneous data: the <note/> to call the boss is not consistent with the application domain.

These inconsistencies can be avoided by using a schema or DTD and enforce its use through the application which processes the XML Data (XML Database, XML Editor).

2.3. Here is how the schema could look like:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">


    <!-- Passenger Element -->
    <xs:element name="Passenger">
    <xs:complexType>
            <xs:sequence>
                <xs:element name="name" type="xs:string"/>
                <xs:element name="passportnumber" type="xs:string"/>
                <xs:element name="address" type="xs:string"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

    <!-- Flight Element -->
    <xs:element name="Flight">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="seats">
                    <xs:simpleType>
                        <xs:restriction base="xs:int">
                            <xs:minExclusive value="30"/>
                            <xs:maxInclusive value="1000"/>
                        </xs:restriction>
                    </xs:simpleType>
                </xs:element>
                <xs:element name="date" type="xs:date"/>
                <xs:element name="source" type="xs:string"/>
                <xs:element name="destination" type="xs:string"/>
            </xs:sequence>
            <xs:attribute name="flightId">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                        <xs:minLength value="0"/>
                        <xs:maxLength value="5"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
        </xs:complexType>
    </xs:element>

    <!-- Airport Element -->
    <xs:element name="Airport">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="name" type="xs:string"/>
                <xs:element name="tax" type="xs:float"/>
            </xs:sequence>
            <xs:attribute name="airId">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                        <xs:minLength value="0"/>
                        <xs:maxLength value="3"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
        </xs:complexType>
    </xs:element>

    <!-- Here goes the booking element definition, see exercise 3.2. -->


   <!-- Root element of the document.-->
   <xs:element name="doc">
      xs:complexType>
       <xs:sequence>
         <xs:element ref="Airport" minOccurs="2" maxOccurs="unbounded" />
```

```
        <xs:element ref="Flight" minOccurs="1" maxOccurs="unbounded" />
        <xs:element ref="Passenger" minOccurs="1" maxOccurs="unbounded" />
        <!-- Here goes the booking element, see exercise 3.2. -->
      </xs:sequence>
    </xs:complexType>
    <!-- Here go the keys and keyrefs, see exercise 3.1. and 3.3. -->
  </xs:element>
  <!-- root -->

</xs:schema>
```

2.4. Do it!

Exercise 3: Comparison with Relational Databases

3.1. We need primary keys for an airport, a flight, and a person (we identify a person with her pass).

```
<!-- Key for airport -->
<xs:key name="airportKey" id="airportKey">
    <xs:selector xpath=".//Airport"/>
    <xs:field xpath="@airId"/>
</xs:key>
<!-- Key for Flight -->
<xs:key name="flightKey">
    <xs:selector xpath=".//Flight"/>
    <xs:field xpath="@flightId"/>
</xs:key>
<!-- Key for Person -->
<xs:key name="personKey">
    <xs:selector xpath=".//Passenger"/>
    <xs:field xpath="passportnumber"/>
</xs:key>
```

3.2. Flights and persons are in an M:N relationship. The M:N relationship "Reservation" between Passengers and Flight can be represented by:

- introducing another entity Reservation (with an attribute 'creditcard')
- AND two additional relationships: Reservation - Flight (M:1 relationship), and Reservation - Passenger (M:1).

Its schema could be defined as:

```
<xs:element name="Reservation">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="flightRef" type="xs:string"/>
      <xs:element name="passRef" type="xs:string"/>
      <xs:element name="creditCard" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

One also has to add it to the sequence of the doc element:

```
<xs:element ref="Reservation" minOccurs="1" maxOccurs="unbounded" />
```

3.3. A flight refers to two airports, a reservation refers to a flight and to a person.

```
<!-- source and destination for flight -->
<xs:keyref refer="airportKey" name="srcKeyRef">
    <xs:selector xpath=".//Flight"/>
    <xs:field xpath="source"/>
</xs:keyref>
<xs:keyref refer="airportKey" name="destKeyRef">
    <xs:selector xpath=".//Flight"/>
    <xs:field xpath="destination"/>
</xs:keyref>
<!-- flight and person for reservation -->
<xs:keyref refer="flightKey" name="flightKeyRef">
    <xs:selector xpath=".//Reservation"/>
    <xs:field xpath="flightRef"/>
</xs:keyref>
<xs:keyref refer="personKey" name="passKeyRef">
    <xs:selector xpath=".//Reservation"/>
    <xs:field xpath="passRef"/>
</xs:keyref>
```

3.4. DTDs can express unique constraints on the IDS of elements. (this satisfies one requirement for primary keys). Attributes defined in a DTD as having the type „ID“ have this property by default. However, the IDS of elements can only be enforced to be unique in the context of the whole document, as opposed to XML Schema, where the domain can be limited through the **selector** expression. In a DTD, references can be enforced using attributes with the IDREF type, which should have the same value as another ID in the same document.
When ID and IDREF are specified, only an “identifier” value can be used as their value (it is not valid to have the id=”5” if the attribute id is declared as having type ID in a certain DTD; for correctness the value should start with a letter (id=”a5”)). XML Schema has the advantage that not only attributes, but also any values can be defined as keys (thus, it is more expressive than a DTD).

Exercise 3: DTD and XML Schema

A possible schema is:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
   <xs:element name="movies">
       <xs:complexType>
           <xs:sequence>
               <xs:element ref="Movie" maxOccurs="unbounded"/>
           </xs:sequence>
       </xs:complexType>
   </xs:element>
    <xs:element name="Movie">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="title" type="xs:string"/>
                <xs:element name="year" type="xs:nonNegativeInteger"/>
                <xs:element name="_director">
                    <xs:complexType>
                        <xs:simpleContent>
                            <xs:extension base="xs:string">
                                <xs:attribute name="name"
                                              type="xs:string"/>
```

```
                        </xs:extension>
                    </xs:simpleContent>
                </xs:complexType>
            </xs:element>
            <xs:choice minOccurs="1" maxOccurs="unbounded">
                <xs:element name="comment">
                    <xs:complexType>
                        <xs:simpleContent>
                            <xs:extension base="xs:string">
                                <xs:attribute name="lang"
                                              type="xs:string"/>
                            </xs:extension>
                        </xs:simpleContent>
                    </xs:complexType>
                </xs:element>
                <xs:element name="newcomment" type="xs:string"/>
            </xs:choice>
        </xs:sequence>
        <xs:attribute name="id" type="xs:ID"/>
    </xs:complexType>
  </xs:element>
</xs:schema>
```