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

Due date/discussion: 25.11.2011

Solution Sheet 4

XML Data Modelling, XML Schema (2)

Exercise 1: Comparison with Relational Databases

1.1. 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>
```

1.2. 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" />
```

1.3. 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>
```

1.4. 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 2: Schema for Schemas (time consuming!)

An official solution can be found in the XML Schema Specification:
http://www.w3.org/TR/xmlschema-1/#normative-schemaSchema

An XML Schema document is also in particular an XML document and is valid against the XML Schema for XML Schemas. The XML Schema for XML Schemas has the nice property of being valid against itself.