

Hochschule Harz	FB Automatisierung und Informatik
Versuch: XML	Thema: Grundkenntnisse in XML mit Schemata

Versuchsziele

- XML-Dateien für eine vorgegebene Schemata-Datei erstellen und ändern können.

Aufgabe

Die mitgelieferten Dateien „Labor2.xml“ und „Labor2.xsd“ beschreiben die Struktur eines Bestellsystems in einer Datenbank. Der aktuelle Stand ist eher auf DTD-Niveau. Es bestehen keinerlei Prüfungen bezüglich der Anzahl und der Wertebereiche. Ändern Sie beide Dateien so, dass die unteren Restriktionen eingehalten werden.

Vorgehensweise:

- Herunterladen der beiden Datei Labor2.xml und Labor2.xsd
- Entzippen und speichern in einem Verzeichnis
- Starten des Programms.
- Laden der beiden Dateien Labor2.xml und Labor2.xsd.
- Testen auf „Validation“ (F5). Eventuelle Fehlermeldungen dann in der XSD bzw. XML-Datei ändern.

1. Teil Restriktionen:

- Die Datenbank kann **mehrere** Bestellungen haben.
- Das Element „artikelnr“ muss mindestens **drei** und darf **maximal 20 Zeichen** haben. Eventuelle Fehler müssen natürlich korrigiert werden.
- Die Anzahl sollte **größer Null** sein. Eventuelle Fehler müssen auch hier korrigiert werden.
- Das Element „status“ soll als **Aufzählung** realisiert werden. Folgende Zeichenketten sind möglich: (Eventuelle Fehler müssen auch hier korrigiert werden).
 - ordered
 - sende
 - cash

2. Teil Aufbau Kunden-Element:

- Im zweiten Teil soll jeweils ein Kunde pro Bestellung eingetragen werden.
- Kommentieren Sie nun alle unteren drei Bestellungen aus:
 - Anfang: < ! . .
 - Ende: -->
- Ein Kunde hat folgende Elemente:
 - vorname
 - Daten xsd:string
 - nachname
 - Daten xsd:string
 - kundenr
 - Daten xsd:string
- Fügen Sie nun **schrittweise** einen Kunden in die erste Bestellung.

3. Teil Restriktionen des Kunden:

- Fügen Sie in den Bestellung-Datensätzen die vier Kunden aus der Datei „kunden.txt“.
 - Quelle: Homepage
- Ein Kunde hat folgende Elemente:
 - vorname
 - Mindestens zwei Zeichen
 - Definieren Sie Pattern für:
 - 1. Zeichen ein Großbuchstabe
 - Danach Kleinbuchstaben
 - Leerzeichen dürfen nicht am Anfang und am Ende stehen
 - nachname
 - siehe Vorname
 - kundennr
 - Definieren Sie Pattern für:
 - 1. Zeichen ein kleines k
 - Danach eine vierstellige Jahreszahl
 - Danach ein Minuszeichen
 - Danach eine beliebige Zahl größer Null
 - Beispiel: k2015-12456
- Ändern Sie die Schemata-Datei „Labor2.xsd“, so dass die XML-Datei valide ist.
- Ändern Sie nun sukzessive die Prüfungen laut der Spezifikation.
- Im Anhang sind Erläuterungen für Pattern.

Datei: Labor2.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<datenbank xsi:noNamespaceSchemaLocation="Labor2.xsd"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <bestellung>
    <artikelnr>cpu-i7-12</artikelnr>
    <anzahl>2</anzahl>
    <bez>Intel i7 260QM, 2.6 GHz</bez>
    <status>ordered</status>
  </bestellung>

  <bestellung>
    <artikelnr>Monitor-17 Montagsproduktion</artikelnr>
    <anzahl>0</anzahl>
    <bez>Asus 1276</bez>
    <status>cash</status>
  </bestellung>

  <bestellung>
    <artikelnr>Maus-ms-12</artikelnr>
    <anzahl>10</anzahl>
    <bez>Microsoft Mouse TR-1234</bez>
    <status>ordered fast</status>
  </bestellung>

  <bestellung>
    <artikelnr>Tablet-asus-10.1</artikelnr>
    <anzahl>2</anzahl>
    <bez>Tablet ASUS 10.1 white, 32 GByte</bez>
    <status> sendet </status>
  </bestellung>

</datenbank>
```

Datei: Labor2.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:element name="datenbank" type="datenbankTyp"/>

  <xsd:complexType name="datenbankTyp">
    <xsd:sequence>
      <xsd:element name="bestellung" type="bestellungTyp" />
    </xsd:sequence>
  </xsd:complexType>

  <xsd:complexType name="bestellungTyp">
    <xsd:sequence>
      <xsd:element name="artikelnr" type="xsd:string" />
      <xsd:element name="anzahl" type="xsd:integer" />
      <xsd:element name="bez" type="xsd:string" />
      <xsd:element name="status" type="xsd:string" />
    </xsd:sequence>
  </xsd:complexType>

</xsd:schema>
```

Pattern

Um die einzelnen Knoten einschränken zu können, stehen mit einem Muster, Pattern, folgende Techniken zur Verfügung:

Ein Pattern beschränkt durch die Angabe eines regulären Ausdrucks den Wertebereich.

Jedes Zeichen kann einzeln definiert werden oder man kann mehrere Zeichen als Gruppe definieren.

Eine zeichenorientierte Definitionen besteht aus eckigen Klammern [abcd] oder [a-z].

Im ersten Fall dürfen nur die Zeichen abcd eingegeben werden.

Im zweiten Beispiel hat man die normalen Kleinbuchstaben.

Die **Anzahl** wird über die geschweiften Klammern geregelt:

{ von,bis }	Minimal: von	Maximal: bis
{ von, }	Minimal: von	Maximal: unbegrenzt
{ n }	Genau n Zeichen	

Weitere Beispiele:

```
<xsd:element name="Buchstabe">
```

```
  <xsd:simpleType>
```

```
    <xsd:restriction base="xsd:string">
```

```
      <xsd:pattern value="[a-z]"/>
```

```
    </xsd:restriction>
```

```
  </xsd:simpleType>
```

```
</xsd:element>
```

```
<xsd:element name="Initialen">
```

```
  <xsd:simpleType>
```

```
    <xsd:restriction base="xsd:string">
```

```
      <xsd:pattern value="[A-Z][A-Z][A-Z]"/>
```

```
    </xsd:restriction>
```

```
  </xsd:simpleType>
```

```
</xsd:element>
```

```

<xsd:element name="Initialen">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:pattern value="[a-zA-Z][a-zA-Z][a-zA-Z]"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>

```

```

<xsd:element name="AuswahleinesZeichens">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:pattern value="[xyz]"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>

```

```

<xsd:element name="artikelnummer">
  <xsd:simpleType>
    <xsd:restriction base="xsd:integer">
      <xsd:pattern value="[0-9][0-9][0-9][0-9][0-9]"/>      1. Variante
      <xsd:pattern value="[0-9]{5}"/>                        2. Variante
      <xsd:pattern value="[0-9]{1,5}"/>                       3. Variante
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>

```

```
<xsd:element name="wort">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:pattern value="([a-z])*"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
```

```
<xsd:element name="letter">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:pattern value="([A-Z][a-z])+"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
```

```
<xsd:element name="geschlecht">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:pattern value="male|female|unisex"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
```

```
<xsd:element name="passwort">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:pattern value="[a-zA-Z0-9_]{8}"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
```