1. Übungsblatt (SS 2018)

3.0 VU Semistrukturierte Daten

Informationen zum Übungsblatt

Allgemeines

Inhalt des ersten Übungsblattes ist die Entwicklung eines Schemata für das Datenformat. Zuerst wird ein XML-Schema entwickelt und ein passendes XML Instanzdokument erstellt. Danach erfolgt die Umsetzung mittels Document Type Definition (DTD).

In dieser Übung geben Sie eine einzige ZIP Datei ab (max. 5MB). Diese ZIP Datei enthält ein XML Schema, eine DTD und zu Schema und DTD je ein passendes XML Instanzdokument, d.h. die Dateien:

- `system.xsd`
- `system-xsd.xml`
- `system.dtd`
- `system-dtd.xml`

Beachten Sie die Erklärungen bei den einzelnen Aufgaben. Das Übungsblatt enthält 5 Aufgaben, auf welche Sie insgesamt 10 Punkte erhalten können.

Deadlines

bis 4.5.	**12:00 Uhr**	Upload der Abgabe über TUWEL
bis 4.5.	**12:00 Uhr**	Anmeldung zu einem Kontrollgespräch

Kontrollgespräch

Im Rahmen des Kontrollgespräches wird nicht nur die Korrektheit, sondern vor allem das Verständnis der Konzepte überprüft. Durch die Übung sollen sowohl Ihre praktische Problemlösungskompetenz als auch das theoretische Wissen über Semistrukturierte Daten gefördert werden. Sie müssen daher bei Ihrem Kontrollgespräch in der Lage sein, nicht nur Ihre Beispiele zu erklären, sondern ebenfalls zeigen, dass Sie die in der Vorlesung behandelte Theorie zu diesen Beispielen ausreichend verstanden haben. Dies soll Ihnen die Vorbereitung für die Prüfung erleichtern und so können Sie Ihr Wissen während der Kontrollgespräche selbst testen und gegebenenfalls vertiefen.

Die Bewertung Ihres Übungsblattes basiert zum überwiegenden Teil auf Ihrer Leistung beim Kontrollgespräch! Es ist daher im Extremfall durchaus möglich, dass eine korrekte Abgabe mit 0 Punkten bewertet wird. Insbesonders werden nicht selbstständig gelöste Abgaben immer mit 0 Punkten bewertet!

Erscheinen Sie in Ihrem eigenen Interesse bitte pünktlich zum Kontrollgespräch, da andernfalls nicht garantiert werden kann, dass Ihre gesamte Lösung in der verbleibenden Zeit beurteilt werden kann.

Bringen Sie bitte Ihren Studentenausweis zum Kontrollgespräch mit. Ein Kontrollgespräch ohne Ausweis ist nicht möglich.

Tutorensprechstunden (freiwillig)

Rund eine Woche vor der Abgabedeadline bieten die TutorInnen Sprechstunden an. Falls Sie Probleme mit oder Fragen zum Stoff des Übungsblattes haben, es Verständnisprobleme mit den Beispielen oder technische Fragen gibt, kommen Sie bitte einfach vorbei. Die TutorInnen beantworten Ihnen gerne Ihre Fragen zum Stoff, oder helfen Ihnen bei Problemen weiter.

Ziel der Sprechstunden ist es, Ihnen beim **Verständnis des Stoffs** zu helfen, nicht, das Übungsblatt für Sie zu rechnen, oder die eigenen Lösungen vorab korrigiert zu bekommen.

Die Teilnahme ist vollkommen freiwillig — Termine und Orte der Tutorensprechstunden finden Sie in TUWEL.

Weitere Fragen – TUWEL Forum

Sie können darüber hinaus das TUWEL Forum verwenden, sollten Sie inhaltliche oder organisatorische Fragen haben.

Aufgaben: XML Schema

Sie werden zuerst ein XML Schema für unsere virtuelle Fabrik erstellen. Speichern Sie dieses XML Schema in der Datei `system.xsd`. Zu diesem Schema wird in Aufgabe 3 ein XML Dokument `system-xsd.xml` erstellt.

Achtung: Stellen Sie sicher, dass ihr Schema wohlgeformt und gültig ist! Sollte das nicht der Fall sein, werden Aufgaben 1 und 2 mit 0 Punkten bewertet! Falls Sie nur Teile der Aufgabe umsetzen können stellen Sie trotzdem sicher, dass sie ein gültiges Schema und ein zu diesem Schema gültiges XML Dokument abgeben!

Aufgabe 1 (Definieren von Elementen in `system.xsd`) *[4 Punkte]*

Das XML Schema soll XML Dokumente mit der folgenden Struktur validieren:

Element `system` Das Wurzelelement `system` speichert die gesamten Daten der "Virtuellen Fabrik" und beinhaltet folgende Elemente in dieser Reihenfolge, wobei alle Elemente *optional* sind:

- maximal ein `items` Element;
- maximal ein `stores` Element;
- beliebig viele `area` Elemente.

Element `items` Das `items` Element hat keine Attribute und ist ein Kindelement von `system`. Dieser Teilbaum speichert alle in der "Virtuellen Fabrik" benötigten oder produzierten Güter. Im `items` Element wird eine beliebige Anzahl von `item` Elementen gespeichert.

Element `item` (Kind von `items`) Das `items`–Kindelement `item` hat ein ganzzahliges Attribut `id` und speichert den Namen des produzierten Items (als String).

Element `stores` Das `stores` Element hat keine Attribute und ist ein Kindelement von `system`. Dieser Teilbaum speichert alle in der "Virtuellen Fabrik" vorhandenen Lager. Im `stores` Element wird eine beliebige Anzahl von `store` Elementen gespeichert.

Element `store` Das `store` Element hat ein ganzzahliges Attribut `id` und folgende Kindelemente (in genau dieser Reihenfolge):

- `name`: Speichert den Namen des Lagers als String;
- `type`: Speichert den Typ des Lagers als String, wobei nur die Werte "normal", "waste" und "sent" erlaubt sind;
- `capacity`: Speichert die Kapazität des Lagers als ganze Zahl.

Element `area` Das `area` Element hat ein Attribut `name` und ist ein Kindelement von `system`. Eine Area stellt eine Kette von Slots dar, die benötigt werden um ein Item zu produzieren. Die Slots einer Area sind die benötigten Schritte zur Erzeugung des Items. Deshalb werden in einem `area` Element eine beliebige Anzahl von `slot` Elementen gespeichert.

Element `slot` Ein `slot` Element hat folgende Attribute:

- eine ganzzahlige `id`;
- ein `name` Attribut mit einem String;
- und ein *optionales* `parallel` Attribut, dass nur boolsche Werte (`true` oder `false` annehmen kann, wobei `false` der standardmäßig benutzt Wert sein soll.

Zusätzlich können folgende Elemente (in dieser Reihenfolge) vorkommen:

- eine beliebige Menge von `input` Elementen;
- ein oder mehrere `slot`, `ref`, `conveyor`, `generator`, `machine` und `turntable` Elemente in beliebiger Reihenfolge; und
- eine beliebige Menge von `output` Elementen.

Elemente `input` und `output` Die `input` und `output` Elemente haben keine Attribute und sind Kindelemente von einem `slot` Element. Die `input` und `output` Elemente haben ein bis mehrere `item` Elemente als Kindelemente. *Achtung:* Dieses `item` Element unterscheidet sich vom `items`–Kindelement `item`.

Element `item` (Kindelement von `input` und `output`) Das `input`/`output`–Kindelement `item` speichert eine ganze Zahl (die Anzahl der benötigten bzw. produzierten Items) und hat ein Attribut `id` und ein *optionales* Attribut `store`.

Element `ref` Das `ref` Element ist ein leeres Element und hat ein Attribut `id`. Es dient zum Wiederverwenden von bereits spezifizierten Slots.

Elemente `conveyor`, `generator`, `machine` und `turntable` Diese Elemente haben alle denselben Typ. Sie haben ein *optionales* Attribut `name` und haben je ein `cost` und `time` Element in beliebiger Reihenfolge als Kindelemente.

Aufgabe 2 (Definieren von Keys und Referenzen in `system.xsd`) *[2 Punkte]*

Fügen Sie nun folgende Schlüssel zu ihrem Schema hinzu:

- Einen globalen Schlüssel `itemKeys` für die `id` der `item` Kindelemente von `items`.
- Einen globalen Schlüssel `storeKeys` für die `id` der `store` Kindelemente von `stores`.
- Einen globalen Schlüssel `slotKeys` für die `id` aller vorkommenden `slot` Elemente.

Fügen Sie folgende Schlüsselreferenzen zu ihrem Schema hinzu:

- Die `id` der `ref` Elemente referenziert die `slotKeys`.
- Die `id` der `input`/`output`–Kindelemente `item` referenziert die `itemKeys`.
- Das `store` Attribut der `input`/`output`–Kindelemente `item` referenziert die `storeKeys`.

Fügen Sie nun folgende Uniqueness-Constraint zu ihrem Schema hinzu:

- Stellen Sie sicher, dass alle `name` Attribute aller Elemente nur einmal vergeben werden.

Aufgabe 3 (Erstellen eines XML Dokuments `system-xsd.xml`) *[1 Punkte]*

Erstellen Sie ein XML Dokument `system-xsd.xml` für das Schema `system.xsd`. Das XML Dokument sollte folgende Bedingungen erfüllen:

- Erstellen Sie zumindest 5 `item` Elemente (im Element `items`).
- Erstellen Sie zumindest 3 `store` Elemente.
- Erstellen Sie zumindest 2 `area` Elemente.
- Ein `area` Element hat zumindest 3 `slot` Elemente (als Kind oder Nachfahre).
- Erstellen Sie ein Element pro Schlüsselreferenz aus Aufgabe 2.

Stellen Sie sicher, dass ihr XML Schema `system.xsd` ihr XML Dokument `system-xsd.xml` validiert. Dies können Sie mit folgendem Kommando testen (nachdem Sie `xmllint` installiert haben):

```
xmllint --schema system.xsd system-xsd.xml
```

Downloads und Installationsanweisungen zu `xmllint` können Sie im TUWEL finden.

Achtung: Falls ihr XML Dokument nicht wohlgeformt und gültig bezüglich des Schemas ist, wird diese Aufgabe mit 0 Punkten bewertet!

Aufgaben: Document Type Definition

Sie werden nun für obige Spezifikation eine DTD erstellen.

Aufgabe 4 (Erstellen einer DTD `system.dtd`) *[2 Punkte]*

Erstellen Sie eine Document Type Definition (DTD) `system.dtd`, welche zur obigen Beschreibung passt. Sollte eine Spezifikation nur sehr kompliziert oder gar nicht umsetzbar sein, treffen Sie entsprechende Annahmen.

Einige Spezifikationsmöglichkeiten von XML Schema lassen sich nur sehr umständlich oder gar nicht in einer DTD umsetzen. Welche Funktionalitäten sind dies? Sie müssen in jedem dieser Fälle beim Abgabegespräch begründen können, warum die Umsetzung in der DTD nicht oder nur teilweise möglich ist.

Insbesondere ist es nicht notwendig, große Zahlenbereiche durch Aufzählungen umzusetzen (z.B. "Zahl zwischen 1 und 72" als Aufzählung von 72 Zahlen). Aufzählungen, die sich hingegen in einem kleinen Zahlenbereich bis inklusive 6 befinden (z.B. maximal fünf Levels, etc.) sind explizit umzusetzen!

Achtung: Falls sie eine DTD mit Syntaxfehlern abgeben, sie also nicht zum Validieren verwendet werden kann, wird diese Aufgabe mit 0 Punkten bewertet!

Aufgabe 5 (Erstellen eines XML Documents `system-dtd.xml`) *[1 Punkte]*

Sollte es nicht möglich sein, dass ihr zuvor erstelltes XML Dokument `system-xsd.xml` durch die DTD validiert werden kann, erstellen Sie ein zusätzliches XML Dokument `system-dtd.xml`, dass die selben Daten enthält und durch ihre DTD validiert wird.

Stellen Sie sicher, dass ihre DTD `system.dtd` ihr XML Dokument `system-dtd.xml` validiert. Dies können Sie mit folgendem Kommando testen (nachdem Sie `xmllint` installiert haben):

```
xmllint --dtdvalid system.dtd system-dtd.xml
```

Achtung: Falls ihr XML Dokument nicht wohlgeformt und gültig bezüglich der DTD ist, wird diese Aufgabe mit 0 Punkten bewertet!