# XML Schema Constraints

XML-Schema gibt die Möglichkeit zur

- Definition von Typen, allgemeiner als mittels einer DTD,
- Definition von Bedingungen `unique, key, keyref`, im folgenden *Constraints* genannt.

Frage:

- Können Typdefinitionen und Constraints in dem Sinne *inkonsistent* sein, dass die Menge der gültigen XML-Dokumente leer ist?
- Falls ja, wie aufwendig ist es, dies zu testen (Konsistenz-Problem)?

## Beispiel 1:

DTD:

```
<!ELEMENT teachers (teacher+)>
<!ELEMENT teacher (name, teach)>
<!ELEMENT teach (subject, subject)>
<!ELEMENT subject (taught_by)>
```

Constraints:

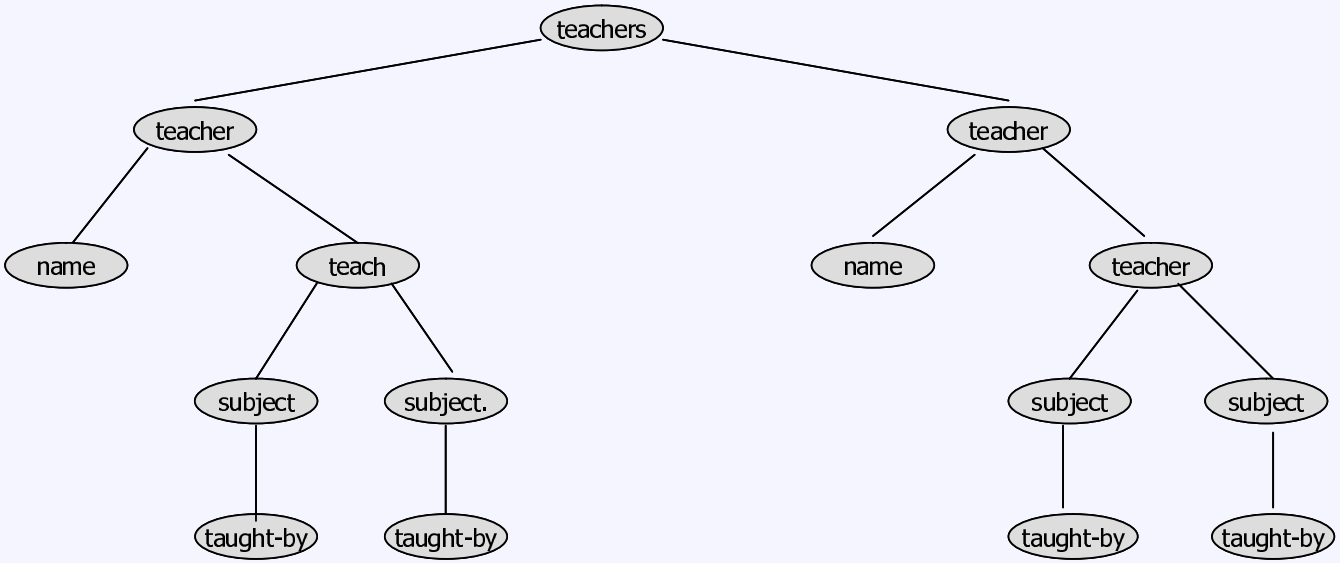
$$(teachers/teacher, \{name\})$$

$$(teachers/teacher/teach/subject, \{taught\_by\})$$

$$(teachers/teacher/teach/subject, \{taught\_by\}) \subseteq_{FK} (teachers/teacher, \{name\})$$

Die DTD zusammen mit der Menge von Constraints ist inkonsistent!

## XML-Baum zu Beispiel 1:



Begründung:

Sei $T$ ein XML-Baum. Es gilt für die jeweiligen Knotenmengen:

$$
\begin{aligned}
&| \, teachers/teacher/name \, | = | \;\; teachers/teacher \, | \\
&| \, teachers/teacher/teach/subject/taught\_by \, | = | \, teachers/teacher/teach/subject \, | \\
&| \, teachers/teacher/teach/subject/taught\_by \, | \leq | \, teachers/teacher/name \, |
\end{aligned}
$$

jedoch verlangt die DTD:

$$2 \, | \, teachers/teacher \, | = | \, teachers/teacher/teach/subject \, |$$

ein Widerspruch.

## Beispiel 2:

DTD:

```
<!ELEMENT vehicle ((registr | plate), policy, policy)>
<!ATTLIST registr num CDATA #REQUIRED>
<!ATTLIST plate num CDATA #REQUIRED>
<!ATTLIST policy reference CDATA #REQUIRED>
```

Constraints:

$$(vehicle/registr \cup\ vehicle/plate, \{@num\})$$
$$(vehicle/policy, \{@reference\})$$
$$(vehicle/policy, \{@reference\}) \subseteq_{FK} (vehicle/registr \cup vehicle/plate, \{@num\})$$

Die DTD zusammen mit der Menge von Constraints ist inkonsistent! Begründung analog zu Beispiel 1.

## Beispiel 3:

DTD:

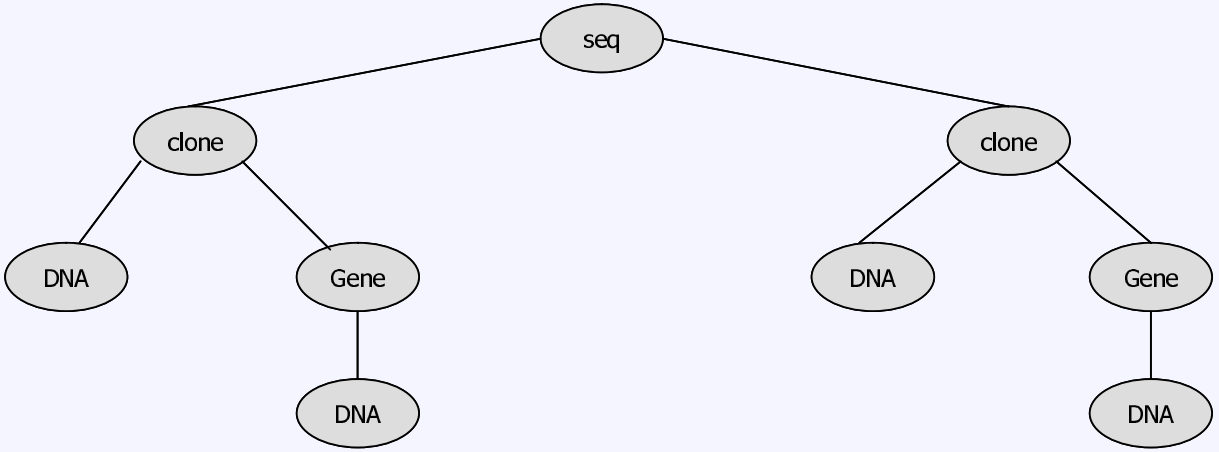
```
<!ELEMENT seq (clone+)>
<!ELEMENT clon (DNA, gene)>
<!ELEMENT gene (DNA)>
```

Constraints:

$$(seq/clone, \{//DNA\})$$

Die DTD zusammen mit der Menge von Constraints ist inkonsistent! Warum?

## XML-Baum zu Beispiel 3:



## Formalisierung

Eine DTD ist ein Tupel

$$D = (E, A, P, R, r),$$

wobei

- $E$ eine endliche Menge von Elementtypen,
- $A$ eine endliche Menge von Attributen, disjunkt zu $E$.
- Für jedes $\tau \in E$, $P(\tau)$ ist ein regulärer Ausdruck $\alpha$, die Elementdefinition von $\tau$:

$$\alpha ::= \mathtt{S} \mid \tau' \mid \epsilon \mid \alpha|\alpha \mid \alpha, \alpha \mid \alpha^*,$$

  wobei S den Typ `string` bezeichnet, $\tau' \in E$, $\epsilon$ das leere Wort und $''\ |''$, $'',''$, und $''*''$ wie gewohnt.
- Für $\tau \in E$ ist $R(\tau)$ eine Menge von Attributen aus $A$.
- $r \in E$ ist der Elementtyp der Wurzel.

Sei $T$ ein XML-Baum und sei $D = (E, A, P, R, r)$ eine DTD. Ist $T$ valid bzgl. $D$ (oder *conforms to*), $T \models D$, dann kann $T$ wie folgt charakterisiert werden:

$$T = (V, lab, ele, att, val, root),$$

wobei

- $V$ ist eine endliche Menge von Knoten,
- $lab$ ist eine Funktion $V \rightarrow E \cup A \cup \{\mathtt{S}\}$. $v \in V$ ist ein Element vom Typ $\tau$, sofern $lab(v) = \tau$; ein Attribut, sofern $lab(v) \in A$; ein Textknoten, sofern $lab(v) = \mathtt{S}$.
- $ele$ ist eine Funktion, die für jeden Elementtyp $\tau$ jedes Element vom Typ $\tau$ auf eine Liste $[v_1, \ldots, v_n]$, $n \geq 0$, von Element- und Textknoten in $V$ abbildet, so dass $lab(v_1) \ldots lab(v_n)$ enthalten in der regulären Sprache zu $P(\tau)$.
- $att$ ist eine partielle Funktion von $V \times A$ nach $V$ so, dass für jedes $v \in V$ und $@l \in A$, $att(v, @l)$ definiert gdw. $lab(v) = \tau, \tau \in E$ und $@l \in R(\tau)$.
- $val$ ist eine partielle Funktion von $V$ nach `string` so, dass für jedes $v \in V$, $val(v)$ definiert gdw. $lab(v) = \mathtt{S}$, oder $lab(v) \in A$.
- $root$ ist die Wurzel von $T$ und $lab(root) = r$.

Sei $D = (E, A, P, R, r)$ eine DTD.

- Ein Key über $D$ ist ein Ausdruck der Form

$$(P, \{Q_1, \ldots, Q_n\}),$$

  wobei $n \geq 1$ und $P, Q_1, \ldots, Q_n$ XPath-Ausdrücke. $P$ ist der Selektor des Ausdrucks und die $Q_i$ die Felder.

- Ein Foreign-Key über $D$ ist ein Ausdruck der Form

$$(P, \{Q_1, \ldots, Q_n\}) \subseteq_{FK} (U, \{S_1, \ldots, S_n\}),$$

  wobei $n \geq 1$ und $P, U, Q_1, S_1, \ldots, Q_n, S_n$ XPath-Ausdrücke. $P, U$ sind die Selektoren des Ausdrucks und die $Q_i, S_i$ die Felder.

Sei $T = (V, lab, ele, att, val, root)$ ein XML-Baum.

(1) Sei $(P, \{Q_1, \ldots, Q_n\})$ ein Key. $T$ erfüllt $(P, \{Q_1, \ldots, Q_n\})$,

$$T \models (P, \{Q_1, \ldots, Q_n\}),$$

wenn gilt:

(a) Für jeden Knoten $x \in nodes_P(root)$ und jedes $i, 1 \leq i \leq n$, existiert genau ein Knoten $y_i$ so dass $T \models Q_i(x, y_i)$. Weiter, $lab(y_i) \in A$, oder $lab(y_i) = \mathsf{S}$.

(b) Seien $x_1, x_2 \in nodes_P(root)$. Wenn $val(x_1.Q_i) = val(x_2.Q_i)$, $1 \leq i \leq n$, dann $x_1 = x_2$.

(2) Sei $(P, \{Q_1, \ldots, Q_n\}) \subseteq_{FK} (U, \{S_1, \ldots, S_n\})$ ein Foreign-Key. $T$ erfüllt $(P, \{Q_1, \ldots, Q_n\}) \subseteq_{FK} (U, \{S_1, \ldots, S_n\})$,

$$T \models (P, \{Q_1, \ldots, Q_n\}) \subseteq_{FK} (U, \{S_1, \ldots, S_n\}),$$

wenn zunächst

$$T \models (U, \{S_1, \ldots, S_n\})$$

und weiter gilt:

(a) Für jeden Knoten $x \in nodes_P(root)$ und jedes $i, 1 \leq i \leq n$, existiert genau ein Knoten $y_i$ so dass $T \models Q_i(x, y_i)$. Weiter, $lab(y_i) \in A$, oder $lab(y_i) = \mathtt{S}$.

(b) Für jedes $x \in nodes_P(root)$ existiert ein Knoten $x' \in nodes_U(root)$, so dass $val(x.Q_i) = val(x'.S_i)$, $1 \leq i \leq n$.

(1)(a), (2)(a) sind XML-Schema-spezifische Eindeutigkeitsbedingungen; mit diesen Bedingungen reden wir über XML-Schema-Keys, ohne über Normal-Keys.

## Komplexitätsergebnisse für das Konsistenz-Problem:

| | Normal-Keys | XML-Schema-Keys |
|---|---|---|
| beliebige Keys und Foreign-Keys | unentscheidbar | unentscheidbar |
| unäre Keys und unäre Foreign-Keys | NP-complete | PSPACE-hart |
| Keys (ohne Foreign-Keys) | lineare Zeit | NP-hart |
| ohne Keys | lineare Zeit | lineare Zeit |

## Das Konsistenz-Problem ist im Allgemeinen unentscheidbar.

- Sei $F$ eine Menge funktionaler Abhängigkeiten und $I$ eine Menge Inklusionsabhängigkeiten eines relationalen Datenbankschemas. Sei $d$ eine funktionale oder eine Inklusionsabhängigkeit. Das Implikationsproblem $F \cup I \models d$ ist unentscheidbar.
- Sei $\Sigma$ eine Menge von Keys und Foreign Keys eines Relationalen Datenbankschemas. Sei $\phi$ ein Key. Das Implikationsproblem $\Sigma \models \phi$ ist unentscheidbar; damit auch das komplementäre Problem $\Sigma \models \neg\phi$.
- $\Sigma \models \neg\phi$ gdw. es existiert eine Instanz $I$ des relationalen Schema, so dass $I \models \Sigma$, $I \not\models \Sigma \wedge \phi$, d.h., $\Sigma \wedge \neg\phi$ ist konsistent.
- Das komplementäre (relationale) Key-Implikationsproblem kann auf das XML-Schema-Konsistenzproblem reduziert werden.

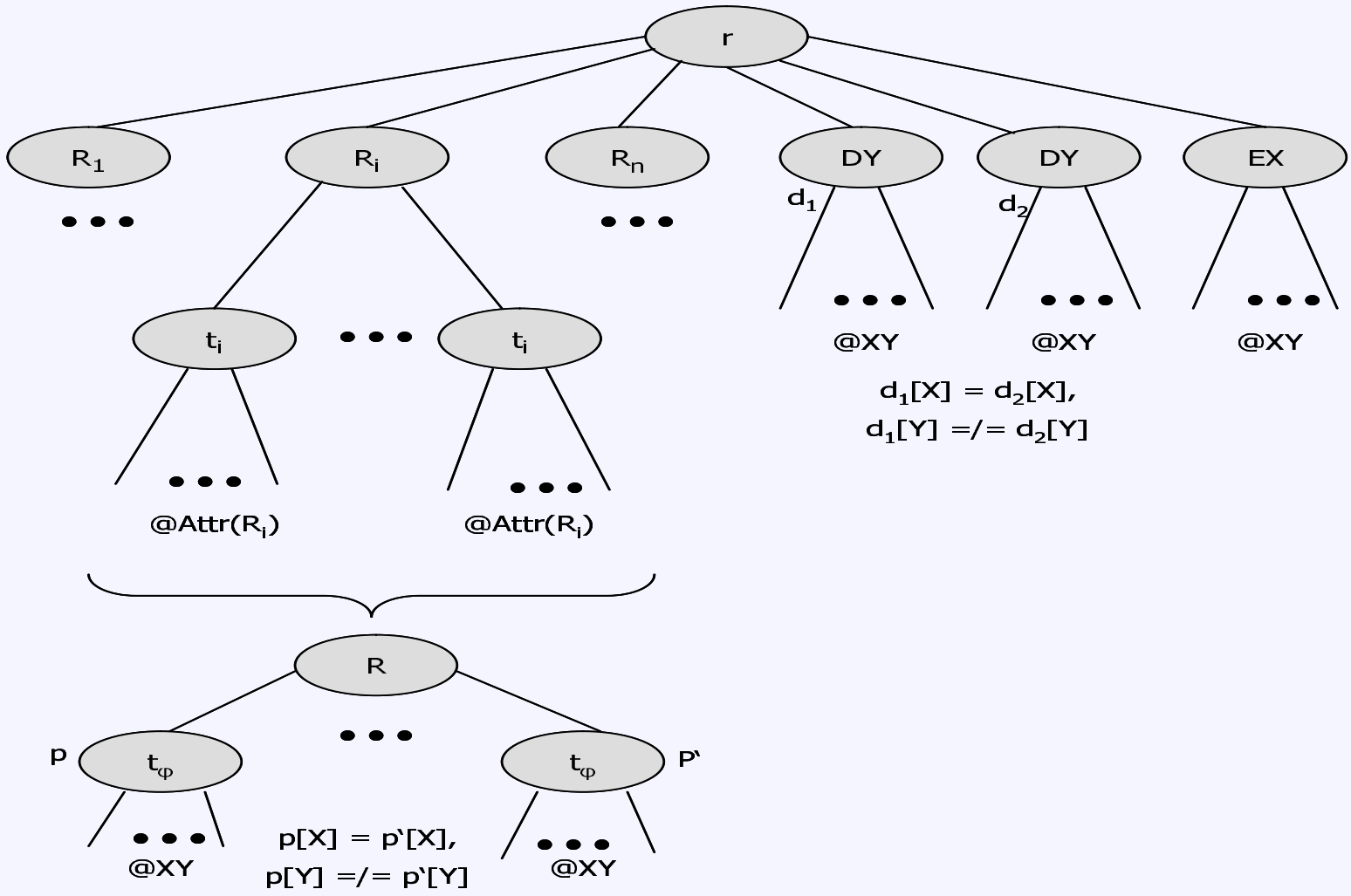
## Reduktion

Sei $\mathbf{R} = (R_1, \ldots, R_n)$ ein relationales Datenbankschema, $\Theta$ eine Menge von Keys und Foreign Keys über $\mathbf{R}$, $\phi = R[X] \rightarrow R$ ein Key über $\mathbf{R}$. Sei $Y = Attr(R) \setminus X$.

Sei eine DTD $D$ zusammen mit einer Menge $\Sigma$ von Key- und Foreign-Key-Constraints eine Kodierung von $\mathbf{R}$, $\Theta$ und $\phi$, wobei $D$ wie in der folgenden Abbildung ersichtlich und zusätzlich $\Sigma = \Sigma_\Theta \cup \Sigma_\phi$ wie folgt:

- $(r/R_i/t_i, Z) \in \Sigma_\Theta$, wenn $R_i[Z] \rightarrow R_i \in \Theta$,
- $(r/R_i/t_i, Z) \subseteq_{FK} (r/R_j/t_j, Z') \in \Sigma_\Theta$, wenn $R_i[Z] \subseteq_{FK} R_j[Z'] \in \Theta$,
- $\Sigma_\phi = \Big\{ \quad (r/DY, @Y),\ (r/EX, @X),$
  $(r/DY, @X) \subseteq_{FK} (r/EX, @X),$
  $(r/DY, @XY) \subseteq_{FK} (r/R/t_\phi, @XY) \quad \Big\}$

## XML-Baum zur Reduktion:



Das Konsistenz-Problem mit Keys, jedoch ohne Foreign-Keys, ist entscheidbar in linearer Zeit, sofern alle Felder eines Keys Attribute des durch den Selektor des Keys bestimmten Elementtyps sind.

- Das Konsistenz-Problem ohne Keys und Foreign-Keys kann in linearer Zeit entschieden werden.
- Sei $D$ eine DTD und $\mathcal{C}_\mathcal{K}$ eine zugehörige Menge von Keys. Sei $T_1$ ein XML-Baum valid zu $D$. Modifiziere $T_1$ zu einem XML-Baum $T_2$, indem $val_1$ zu $val_2$ so abgeändert wird, dass für je zwei Knoten $v_1, v_2$ mit $label(v_1) = label(v_2) = \tau$ gerade $val_2(att(v_1, @l)) \neq val_2(att(v_2, @l))$ für alle durch $D$ dem Elementtyp $\tau$ zugeordneten Attribute $@l$.

Bemerkung: Sind alle Felder eines Key Attribute, dann ist der Key ein Normal-Key.

Literatur:

Peter Buneman, Susan Davidson, Wenfei Fan, Carmen Hara, Wang-Chiew Tan. Keys for XML. Proceedings WWW10, 2001.
Wenfei Fan, Leonid Libkin. On XML Integrity Constraints in the Presence of DTDs. JACM, Vol 49, No. 3, 2002.
Marcelo Arenas, Wenfei Fan, Leonid Libkin. What's Hard about XML Schema Constraints? DEXA 2002, LNCS 2453, Springer Verlag, 2002.
Marcelo Arenas, Wenfei Fan, Leonid Libkin. On Verifying Consistency of XML Specifications. Proceedings ACM PODS, 2002.