UNIVERSITÄT SIEGEN

Vorlesung Datenbanksysteme II

XPath

Inhalt

- Grundlagen
- Lokalisierungspfade
- Navigationsschritte
- Navigationsrichtungen
- Knoten(typ)tests
- Weitere Selektionsbedingungen
- Vergleich mit relationalen Anfragesprachen

UNIVERSITÄT SIEGEN

Grundlagen von XPath

Motivation

SELECT Name

FROM Adressliste

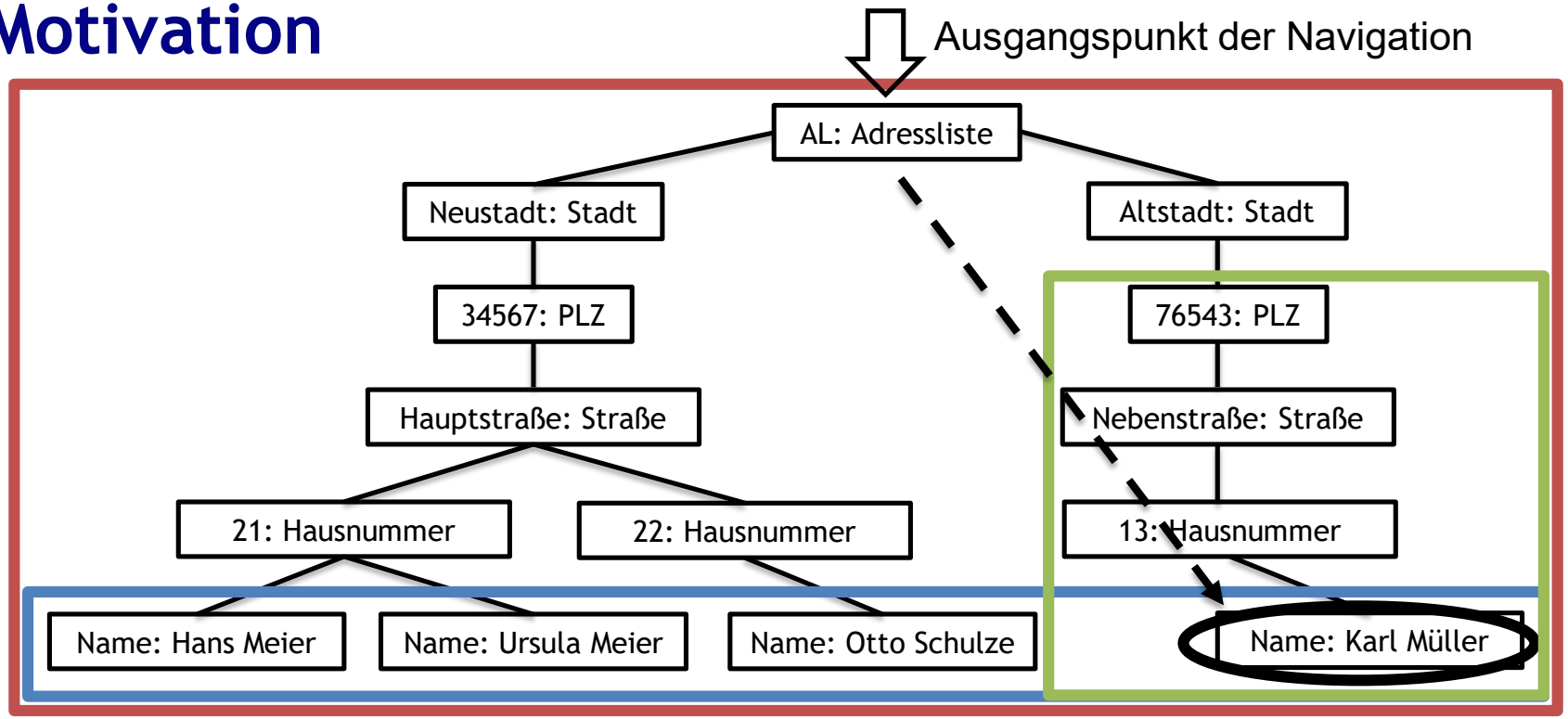
WHERE PLZ > 40000;

Name	Straße	Hausnr.	PLZ	Stadt
Hans Meier	Hauptstr.	21	34567	Neustadt
Ursula Meier	Hauptstr.	21	34567	Neustadt
Karl Müller	Nebenstr.	13	76543	Altstadt
Otto Schulze	Hauptstr.	22	34567	Neustadt
…	…	…	…	…

Rückblick auf Relationale Abfragesprachen:

Selektion/Projektion/Verbund/Aggregation relevanter Teilkomponenten aus (Teil-)Mengen von Tupeln der Tabellen einer Datenbank.

Motivation



Analoges Problem bei XML-Dokumenten:
Navigieren zu einer relevanten Teilmenge der Knoten eines Baumes.

Navigierende Abfragesprachen

- Navigierende Abfragesprachen werden benötigt, wenn das zugrundeliegende Datenmodell als Baum / Graph / Netzwerk strukturiert ist.
- Anders als im relationalen Modell gibt es in solchen Datenmodellen Konzepte zur **expliziten** Modellierung semantischer Zusammenhänge zwischen Datenknoten, zumeist in Form von (gerichteten) Kanten.

Was ist/macht XPath?

- Textuelle Sprache für navigierende Abfragen in Baum-basierten Dokumenten.
- Neues Sprachkonstrukt: **Mengenwertiger Navigationsoperator**.
- Ausgangspunkt einer Navigation ist ein beliebiger Knoten des Eingabebaumes.
- Zwischen- und Endergebnis einer (Teil-)Navigation ist eine **Menge** von Referenzen auf relevante Zielknoten.

Was macht XPath nicht?

- Abfragen in XPath produzieren **keine Ausgaben.**
- Die Ergebnismengen von Abfragen werden stattdessen durch andere Technologien **weiterverarbeitet** (XSLT, Xquery – siehe später).
- XPath ist somit zumeist integraler (versteckter) Bestandteil anderer Technologien.

Versionen von XPath

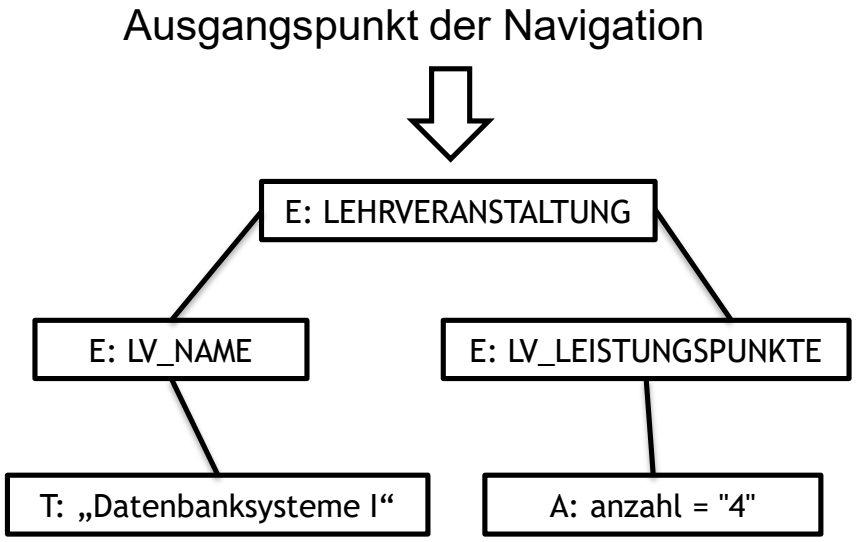
1. XML Path Language (XPath), Version 1.0, W3C Recommendation 16 Nov. 1999 (updated Oct. 2016); http://www.w3.org/TR/xpath-10
2. XML Path Language (XPath) 2.0 (Second Edition), W3C Recommendation 14 Dec. 2010 (updated 3 Jan. 2011 and Oct. 2016); http://www.w3.org/TR/xpath20
3. XML Path Language (XPath) 3.0, W3C Recommendation 08 Apr. 2014; http://www.w3.org/TR/xpath-30

(Wir betrachten im Folgenden Version 1.0. Die Änderungen in Version 2.0/3.0 sind nur bei sehr intensiver Nutzung relevant.)

Ein erstes Beispiel

```
<LEHRVERANSTALTUNG>
<LV_NAME>Datenbanksysteme I</LV_NAME>
<LV_KUERZEL>DBS1</LV_KUERZEL>
<LEISTUNGSPUNKTE anzahl="4" /> ....
</LEHRVERANSTALTUNG>
```

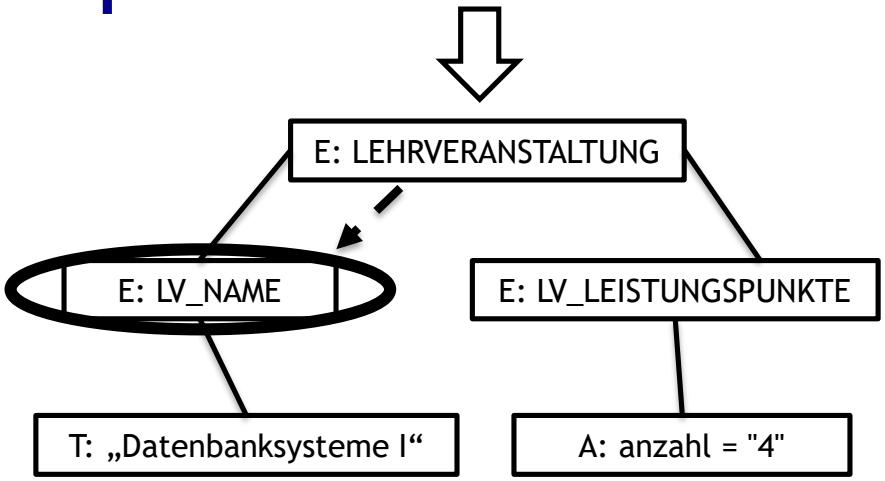
XML-Datei (Auszug)



Syntaxbaum (Auszug)

Ausgangspunkt der Navigation: **ein** Knoten vom Elementtyp „LEHRVERANSTALTUNG“.

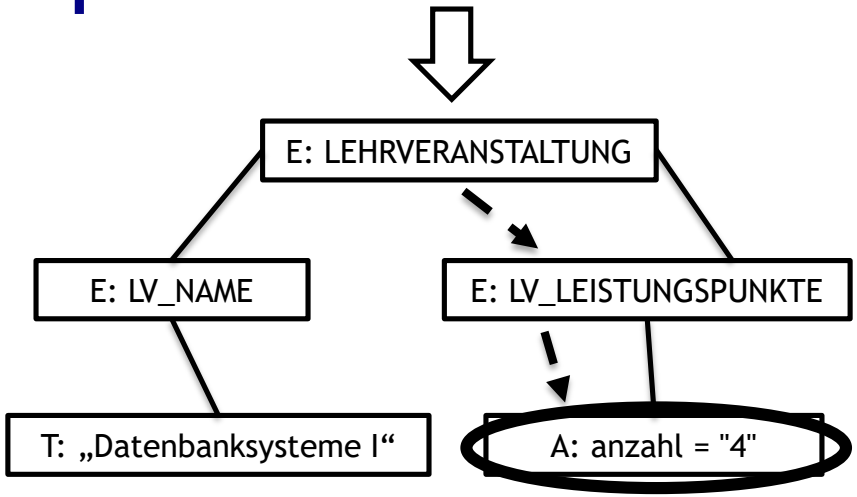
Ein erstes Beispiel



XPath-Ausdruck: „LV_NAME“

- „LV_NAME“ ist direkter Kindknoten von „LEHRVERANSTALTUNG“
- Navigiert zu **allen** solchen direkten Kindknoten des Startknotens vom Typ „LEHRVERANSTALTUNG“

Ein erstes Beispiel



XPath-Ausdruck: „LEISTUNGSPUNKTE / @anzahl“

- Attributknoten „anzahl“ ist 2 Navigationsschritte vom Startknoten entfernt.
- Notation „/“ ähnlich wie Pfadnamen in Dateisystemen.
- Notation „@anzahl“ für Attributzugriffe (siehe später).

Lokalisierungspfade

Syntax von Lokalisierungspfaden

```
LocationPath                ::= RelativeLocationPath |
                                AbsoluteLocationPath
AbsoluteLocationPath        ::= '/'
                                RelativeLocationPath? |
                          AbbreviatedAbsoluteLocationPath
RelativeLocationPath        ::= Step |
                                RelativeLocationPath
                                '/' Step |
                        AbbreviatedRelativeLocationPath
```

Kontextknoten eines Lokalisierungspfades

Der Kontextknoten ist der Ausgangspunkt einer Navigation.

- Bei **absoluten** Pfaden: Wurzelknoten des Dokuments ist Startpunkt des ersten Navigationsschrittes.
- Bei **relativen** Pfaden: beliebiger Knoten des Eingabebaumes als (Zwischen-)Ergebnis vorheriger Navigationsschritte.

Einfache Navigationsschritte

Ein Navigationsschritt, angewandt auf **einen** Kontextknoten im Eingabesyntaxbaum, liefert eine **Menge** von (Referenzen auf) Zielknoten im Eingabesyntaxbaum.

Mengenwertige Navigationsschritte

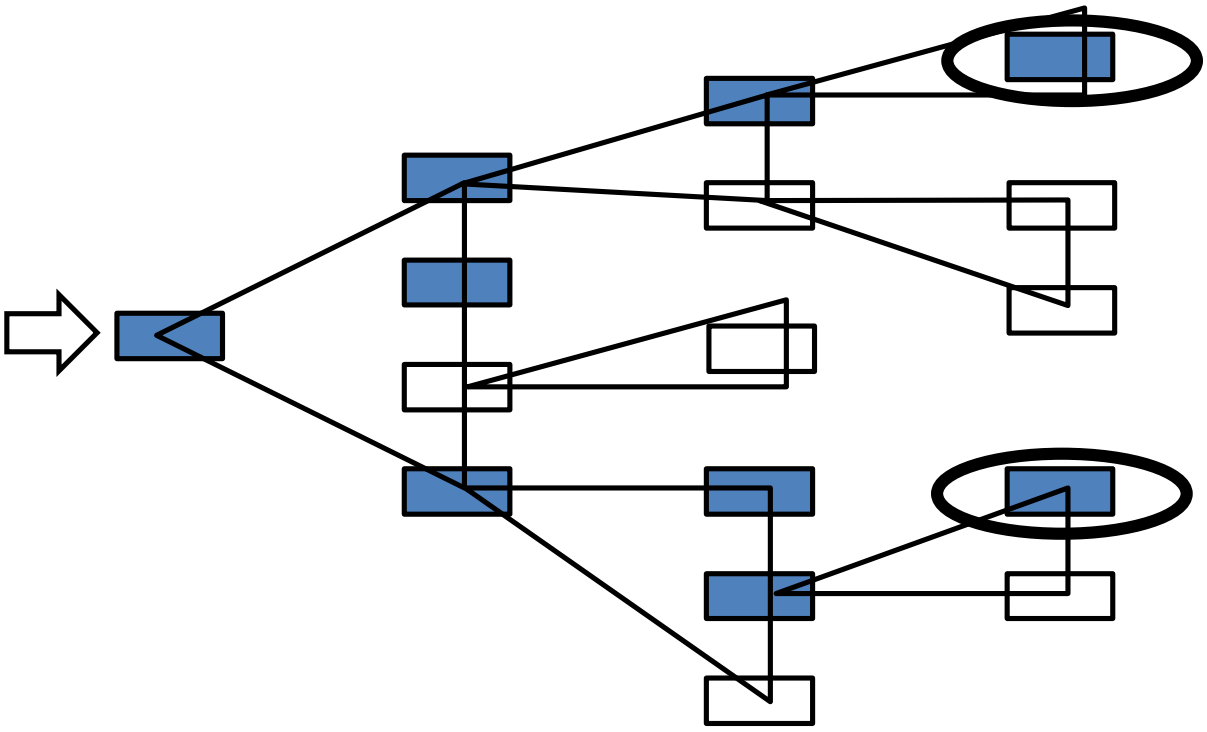
- Ein Navigationsschritt, angewandt auf **eine Menge** von Kontextknoten im Eingabesyntaxbaum, liefert für **jeden einzelnen Knoten** der Kontextmenge eine Menge von (Referenzen auf) Zielknoten im Eingabesyntaxbaum.
- Das Ergebnis des Navigationsschritte ist die (duplikatenfreie!) **Vereinigung** dieser Zielknotenmengen.

Sequenzen von Navigationsschritten

Beispiel: „schritt1 / schritt2“

- Auswertung erfolgt von links nach rechts.
- Kontextknoten von „schritt1“ ist der Kontextknoten des Lokalisierungspfads.
- Kontextknoten von „schritt2“ ist die Menge der Zielknoten von „schritt1“.
- Zielknoten von „schritt1 / schritt2“ ist die Menge der Zielknoten von „schritt2“ angewendet auf die Zielknoten von „schritt1“.

Beispiel mit drei Navigationsschritten



schritt1 / schritt2 / schritt3

= Knoten in Zielmenge eines Schrittes

= Knoten nicht in Zielmenge eines Schrittes

= Knoten in Ergebnismenge der Navigation

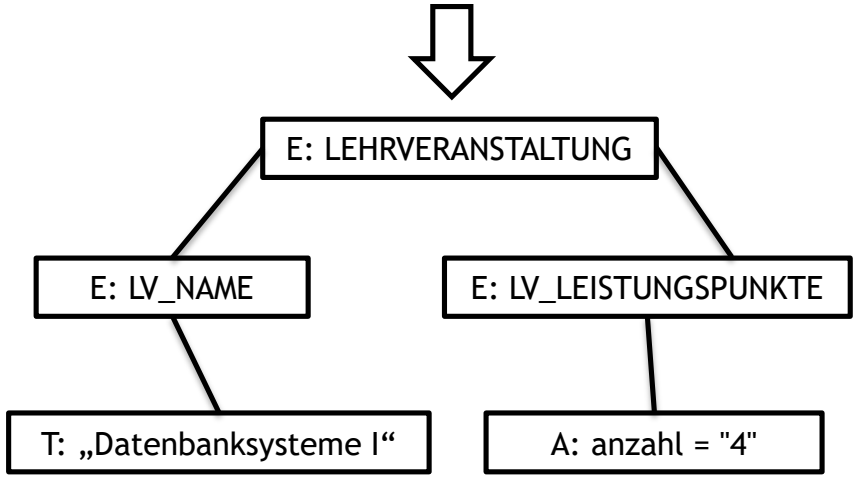
Navigationsschritte

Einfache Navigationsschritte

Häufigster und einfachster Fall eines Navigationsschrittes:

- Angabe eines gesuchten Kindknoten**typs** oder Attributs.
- Bedeutung: gehe vom aktuellen Knoten aus zu **allen** Kindknoten mit dem angegebenen Typ bzw. zu diesem Attribut.

Beispielaufgabe



Lokalisiere alle Attributknoten vom Typ „anzahl“ derjenigen Elementknoten vom Typ „LV_LEISTUNGSPUNKTE“.

Beispielaufgabe

Lösung: „LV_LEISTUNGSPUNKTE/@anzahl“.

Anmerkung: Dies ist eigentlich bereits eine abkürzende Schreibweise (siehe später).

Beispielaufgabe

Wieder bezogen auf das Lehrveranstaltungen-Beispiel, aber jetzt für einen bisher „unbekannten“ Teil des Baumes.

Lokalisiere alle Attributknoten vom Typ „semester“ der Elementknoten vom Typ „DURCHFUEHRUNG“.

Beispielaufgabe

Lösung: „DURCHFUEHRUNG/@semester“.

Bestandteile eines Navigationsschrittes

- **Navigationsrichtung** (“*axis*”) im Syntaxbaum.
  Beispiel: „child“ für direkte Kindknoten.
- **Selektion** der Kandidatenknoten anhand eines Knoten(typ)tests.
  Beispiel: Knotentyp „DURCHFUEHRUNG“.
- **Weitere Selektionsbedingungen** (optional).
  Beispiel: Knoten hat Attribut „semester“ mit Wert „xyz“.

Beispielaufgabe

Lokalisiere alle Attributknoten vom Typ „semester“ der Elementknoten vom Typ „DURCHFUEHRUNG“ mit dem Wert ’xyz’.

Beispielaufgabe

Lösung:

child::DURCHFUEHRUNG [@semester = ’xyz’]

Syntax von Navigationsschritten

Step ::= AxisSpecifier NodeTest
Predicate* | AbbreviatedStep

AxisSpecifier ::= AxisName ’::’ |
AbbreviatedAxisSpecifier

Auswertungssemantik von Navigationsschritten

- Auswertung der **Navigationsrichtung**:
  Bestimme ausgehend vom Kontextknoten die **Menge *M1* potentieller Ergebnisknoten.**

- Auswertung des **Knoten[typ]tests**:
  Selektiere anhand des Knotentyps die **Teilmenge** *M2* von *M1*.

Auswertungssemantik von Navigationsschritten

Die Selektion erfolgt:

(a) anhand der 7 grundlegenden Knotentypen eines XML-Syntaxbaums,

(b) im Fall von Elementknoten und Attributknoten durch deren „(Typ-)Namen“.

Auswertungssemantik von Navigationsschritten

Auswertung der weiteren Selektionsbedingung: Selektiere anhand der Bedingung (in der Regel bezogen auf Attributwerte) Teilmenge **M3** von **M2.**

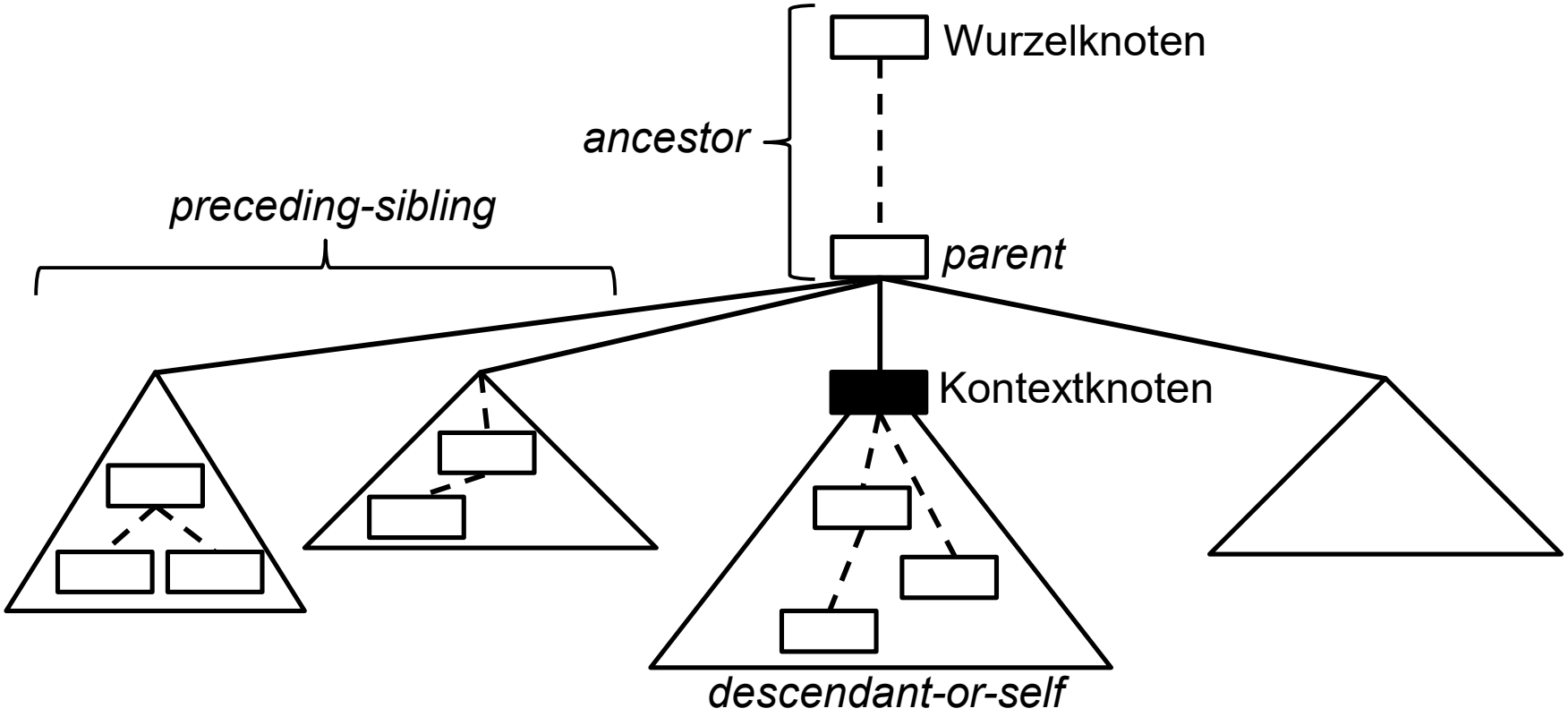
UNIVERSITÄT SIEGEN

Navigationsrichtungen

Syntax von Navigationsrichtungen (Achsen)

AxisName ::= ’ancestor’ | ’ancestor-or-self’ |
’attribute’ | ’child’ | ’descen-
dant’ | ’descendant-or-self’ |
’following’ | ’following-sibling’ |
’namespace’ | ’parent’ |
’preceding’ | ’preceding-sibling’ |
’self’

Beispiele für Navigationsrichtungen



UNIVERSITÄT SIEGEN

Klassifizierung der Navigationsrichtungen

Klassen ergeben sich aus den Knotentypen, die in der Navigationsrichtung enthalten sein können:

1. Navigationsrichtungen, die **Attributknoten** enthalten können (nur „attribute“).
2. Navigationsrichtungen, die **Namensraum-knoten** enthalten können (nur „namespace“).
3. Navigationsrichtungen, die **Elementknoten** enthalten können (alle anderen).

Klassifizierung der Navigationsrichtungen

- Achtung: „child“ und ähnliche Ausdrücke aus Klasse 3 enthalten deshalb im Folgenden **keine Attribut- und Namensraumknoten.**
- Namensräume werden wir später behandeln.

Zielknotenmengen der Navigationsrichtungen

- „attribute“: alle Attribute des Kontextknotens; leere Menge, falls der Kontextknoten kein Elementknoten ist.
- „namespace“: alle Namensraumknoten eines Elements; leere Menge, falls der Kontextknoten kein Element ist.
- „self“: der Kontextknoten selbst.
- „child“: alle direkt enthaltenen Kindknoten.

Zielknotenmengen der Navigationsrichtungen

- „descendant“: die Nachfahren, d.h. alle Kindknoten und deren Nachfahren-Knoten (d.h. alle textuell **innenliegenden** Knoten).
- „descendant-or-self“: alle „descendant“-Knoten und der Kontextknoten selbst.
- „parent“: der direkte Elternknoten des Kontextknotens, sofern vorhanden (d.h. Kontextknoten ungleich Dokumentwurzel).

Zielknotenmengen der Navigationsrichtungen

- „ancestor“: alle Vorfahren bis einschließlich der Dokumentwurzel.
- „ancestor-or-self“: alle „ancestor“-Knoten und der Kontextknoten selbst.
- „following-sibling“: nachfolgende Geschwister, also Kinder des gleichen Elternknotens, die in **Dokumentordnung (= textuelle Anordnung) hinter** dem Kontextknoten liegen.

Zielknotenmengen der Navigationsrichtungen

- „preceding-sibling“: vorhergehende Geschwister (analog zu „following-sibling“).
- „following“: alle Knoten, die in der Dokument-ordnung **nach** dem Ausgangsknoten folgen (keine Nachfahren, die liegen darunter).
- „preceding“: alle Knoten, die in der Dokument-ordnung **vor** dem Kontextknoten kommen (keine Vorgänger, die liegen darüber).

Abkürzungen für Navigationsrichtungen

keine Angabe = „child::“ (= Vorgabewert)

Beispiel:
LEHRVERANSTALTUNG / DURCHFUEHRUNG =
child::LEHRVERANSTALTUNG / child::DURCHFUEHRUNG

(Abkürzung „/Vorgaben“ ersetzt
„Navigationsrichtung::“)

Abkürzungen für Navigationsrichtungen

@ = attribute::

Beispiel:
@anzahl =
attribute::anzahl

Abkürzungen für Navigationsrichtungen

. = self::node()

.. = parent::node()

- Diese Abkürzungen ersetzen die **kompletten** Angaben zwischen zwei Schrägstrichen „/“.
- Nach „.“ und „..“ sind **keine** weiteren Selektionsbedingung erlaubt.
- Nach „.“ bzw. „..“ kann ein weiterer Schrägstrich „/“ folgen.

Abkürzungen für Navigationsrichtungen

// = / descendant-or-self::node() /

- Hier sind auch die beiden Schrägstriche, die Navigationsschritte trennen, in der Abkürzung enthalten.

Abkürzungen für Navigationsrichtungen

Beispiel 1:

. // @semester =

. / descendant-or-self::node() / attribute::semester

Unterschied?

Beispiel 2:

// @semester =

/ descendant-or-self::node() / attribute::semester

Abkürzungen in Navigationsrichtungen

- Nicht erlaubt: „//“ am Ende des Pfadausdrucks, stattdessen muss immer ein kompletter weiterer Navigationsschritt folgen.
- Somit ist auch folgendes **syntaktisch falsch**:
  `// [@attribut = 4]`

Knoten(typ)tests

Arten von Knoten(typ)tests

Die Art des Knoten(typ)tests ist **abhängig von der Klasse der Navigationsrichtung:**

- Navigationsrichtungsklasse „Attribute“: Alle Zielknoten sind **Attribut-Knoten.**
- Navigationsrichtungsklasse „Namensräume“: Alle Zielknoten sind **Namensraum-Knoten.**
- Restliche Fälle: als Zielknoten können **Element-Knoten** auftreten.

Knotentest für Attributknoten

- Einzelne Namen zur Selektion eines bestimmten Attributs (falls vorhanden, andernfalls leere Menge).
  Beispiel: DURCHFUEHRUNG/@semester
- „*“ zur Selektion aller Attribute.
- Nicht erlaubt: Textmuster für Attributnamen, z.B. @seme*

Knotentest für Elementknoten

- „node()“: Selektion aller Knoten, also auch Textknoten, Kommentare,...
- „*“: Selektion aller Elemente (nicht alle Knoten).
- „:einzelnerName“: Selektion der Elemente, deren Typ diesen Namen hat.
  Beispiel:
  LEHRVERANSTALTUNG/DURCHFUEHRUNG

Knotentest für Elementknoten

- Nicht erlaubt: Textmuster für Elementnamen, z.B. LEH*T*G/DURCHF*.
- „text()“: Selektion aller Textknoten.
- „comment()“: Selektion aller Kommentarknoten.
- „processing-instruction()“: Selektion aller Verarbeitungsanweisungen.

Syntax der Knotentests

```
NodeTest   ::= NameTest | NodeType '(' ')' |
               'processing-instruction' '(' Literal ')'
NameTest   ::= '*' | NCName ':' '*' | QName
NodeType   ::= 'comment' | 'text' |
               'processing-instruction' | 'node'
```

Weitere Selektionsbedingungen

Arten weiterer Selektionsbedingungen

Selektionsbedingungen können sich beziehen auf:

- Die Werte von **anderen** Knoten, die über einen inneren (relativen) Pfadausdruck spezifiziert werden können.
  Beispiel: // LEHRVERANSTALTUNG [./LV_KUERZEL = 'DBS_II'] / DURCHFUEHRUNG
- Die **Struktur** des Syntaxbaums (z.B. Kontextknoten ist erster von mehreren Geschwisterknoten).

Diskussion: Weitere Selektionsbedingungen

- Insgesamt deutlich unsystematischer als im relationalen Modell.
- Hier nur auszugsweise dargestellt, komplette Spezifikation, siehe: www.w3.org/TR/xpath

Syntax weiterer Selektionsbedingungen (Auszug)

Einfache oder komplexe Bedingung in „[...]“:

```
Step           ::= AxisSpecifier NodeTest
                   Predicate* | ...
Predicate      ::= '[' PredicateExpr ']'
PredicateExpr  ::= Expr
```

Syntax weiterer Selektionsbedingungen (Auszug)

Boolesche Ausdrücke bestehen aus elementaren Booleschen Ausdrücken, die wie üblich mit Booleschen Operatoren verbunden werden können.

- Schlüsselworte: „or“, „and“, …
- Vordefinierte Boolesche Funktion „not(...)“.

Beispiele

Kontextknoten sei ein LEHRVERANSTALTUNG-Knoten.

Gesucht: alle DURCHFUEHRUNG-Knoten dieser LEHRVERANSTALTUNG, bei denen das Attribut dozentId den Wert ”NN” hat.

Beispiele

Lösung:

DURCHFUEHRUNG [@dozentId = ’NN’]

bzw.

child::DURCHFUEHRUNG [attribute::dozentId = ’NN’]

Beispiele

Erläuterungen:

- „child::DURCHFUEHRUNG“ liefert Referenzen auf alle zugehörigen DURCHFUEHRUNG-Knoten.
- Das Datum, anhand dessen wir einen Kandidaten selektieren, steht nicht im Kandidatenknoten selbst, sondern muss von dort aus durch den Pfad „@dozentId“ lokalisiert werden.

Numerische Vergleiche

Beispiel: LEHRFORM [@sws_umfang > 0]

Selektiert LEHRFORM-Elemente, bei denen das Attribut „sws_umfang“ einen numerischen Wert größer 0 hat.

Arithmetische Ausdrücke

Beispiel: LEHRFORM [@sws_umfang = ../LEISTUNGSPUNKTE/@anzahl * 0.5]

- Selektiert alle LEHRFORM-Elemente, die zwei Leistungspunkte pro SWS haben.
- Besonderheit: ein innerhalb und ein außerhalb des Selektionskandidaten liegendes Attribut werden verglichen und es wird ein numerischer Ausdruck benutzt.

Inhalte von Textknoten

Beispiel: LEHRFORM [. = ’Vorlesung’]

Selektiert alle LEHRFORM-Elemente, deren Inhalt (= enthaltene Textknoten inkl. Leerraum) = ’Vorlesung’ ist.

Innere Pfadausdrücke

- Pfadausdruck in Boole‘schen Ausdrücken in weiteren Selektionsbedingungen.
- Die lokalisierten Knoten können an beliebiger Stelle im Syntaxbaum liegen (auch “oberhalb” oder “seitlich”).
- Die lokalisierten Knoten werden auf alle Knoten des vorherigen Knotentests angewandt.

Innere Pfadausdrücke

Beispiel: / A / B / child::C [D/@id = ’3’]

- Kontextknoten für Auswertungen von „child::C [. . .]“ ist ein B-Element im Eingabebaum.
- Kontextknoten für Auswertungen von „D / attribute::id“ ist ein C-Element im Eingabebaum und ist zugleich Kontextknoten für den 1. Schritt des inneren Pfadausdrucks.

Einfache Boole‘sche Ausdrücke

- Vergleichsoperatoren für numerische Werte:
  =, !=, <, <=, <, >=, >
- Vergleichsoperatoren für Texte:
  = (und indirekt != mittels not(... = ...))

Automatische Konversionen:

- Zu Zahlen bei numerischen Vergleichen.
- Zu Texten bei Vergleich mit Text-Konstanten.
- Textueller Wert von Elementen: „s.xsl:value-of“

Implizite Existenz-Quantoren

Beispiel: Gesucht sind alle Lehrveranstaltungen, die (wenigstens) einmal vom Dozenten „NN“ durchgeführt wurden.

Implizite Existenz-Quantoren

Lösung: // LEHRVERANSTALTUNG [DURCHFUEHRUNG / @dozentId = ’NN’]

- Linker Teil des Boole‘schen Ausdrucks ist Pfadausdruck: „DURCHFUEHRUNG / @dozentId .“
- Alle Treffer werden mit der Konstanten „NN“ verglichen. Ergebnis ist TRUE, wenn **mindestens ein** passender Treffer **existiert**.

Existenz-Test

Selektion der Kandidaten, falls ein anderer Knoten **existiert.**

- Syntax: Angabe eines Pfades (impliziter Existenz-Quantor).
- Test bestanden, wenn wenigstens ein Treffer existiert.

Existenz-Test

Beispiel: LERNZIELE [BEWERTUNGSKOMPETENZ]

Selektiert die LERNZIELE-Elemente, in denen wenigstens ein Element des Typs BEWERTUNGSKOMPETENZ enthalten ist.

Existenz-Test

Beispiel: LEHRVERANSTALTUNG [LERNZIELE/BEWERTUNGSKOMPETENZ]

Selektiert die LEHRVERANSTALTUNGen, die wenigstens ein LERNZIELE-Element enthalten, in dem ein Element des Typs BEWERTUNGSKOMPETENZ enthalten ist.

Selektion durch Position

Kandidaten können auch anhand der Position im “aktuellen Kontext” selektiert werden.

Selektion durch Position

Beispiel:

DURCHFUEHRUNG [position() = 1] bzw.

DURCHFUEHRUNG [1]

Selektiert das **erste** von potentiell mehreren DURCHFUEHRUNG-Elementen innerhalb eines umgebenden Elements.

Selektion durch Position

Beispiel: DURCHFUEHRUNG [position() = last()]

Selektiert das **letzte** von potentiell mehreren DURCHFUEHRUNG-Elementen.

Positionsfunktionen

- „position()“ liefert Positionsnummer des gerade behandelten Elements in der Menge *M2* (1. Element steht an Position 1).
- „last()“ liefert Positionsnummer des letzten Elements von *M2* (= Anzahl der Knoten in *M2*).

Zählfunktionen

Funktion „count(…)“ liefert die Zahl der Knoten in der Knotenmenge, die als Argument angegeben wird.

Zählfunktionen

Beispiel: Lokalisiere alle Lehrveranstaltungen, die mehr als zwei Mal durchgeführt wurden bzw. werden.

Zählfunktionen

Lösung: // LEHRVERANSTALTUNG [count (DURCHFUEHRUNG) > 2]

Vergleich mit relationalen Anfragesprachen

Diskussion: XPath allein

- Dominierende Operation: Navigation.
- Jeder Navigationsschritt entspricht einem impliziten Verbund.
- Keine Operatoren für Projektion, Aggregation, Sortierungen, allgemeine Verbunde (siehe XSLT).

Diskussion: XPath mit XSLT

- Projektion: unterstützt.
- Sortierungen: unterstützt.
- Aggregationen: eingeschränkt und umständlich.
- Verbunde: eingeschränkt und umständlich.

Kritische Bewertung von XPath

- Kryptische Syntax.
- Vermischung von Layoutgenerierung und reinen Suchproblemen (Fachlogik).
- Für “kleine” Anwendungen vertretbar, für komplexe Anwendungen fragwürdig (kein Modulkonzept, keine vernünftigen Abstraktionsmechanismen, …).

Prüfungsstoff

- Grundlegende Konzepte von XPath 1.0 erklären können.
- XPath auf konkrete Beispiele anwenden können.
- Die wesentlichen Unterschiede zwischen XPath und relationalen Sprachen kennen und bewerten können.