Standards II:

XQuery

Anja Rein

Hauptseminar bei Prof. Bayer, Prof. Kossmann
Technische Universität München

Wo stehen wir?

- Einführung in Seminartechniken
- Anwendungsszenarien für Web-Services
- W3C Standards I :
  XML Protocol (SOAP), WSDL & UDDI

heute: W3C Standards II :
XQuery

Fragen

Was ist XQuery?

Warum soll man XQuery benutzen?

Wie funktioniert XQuery?

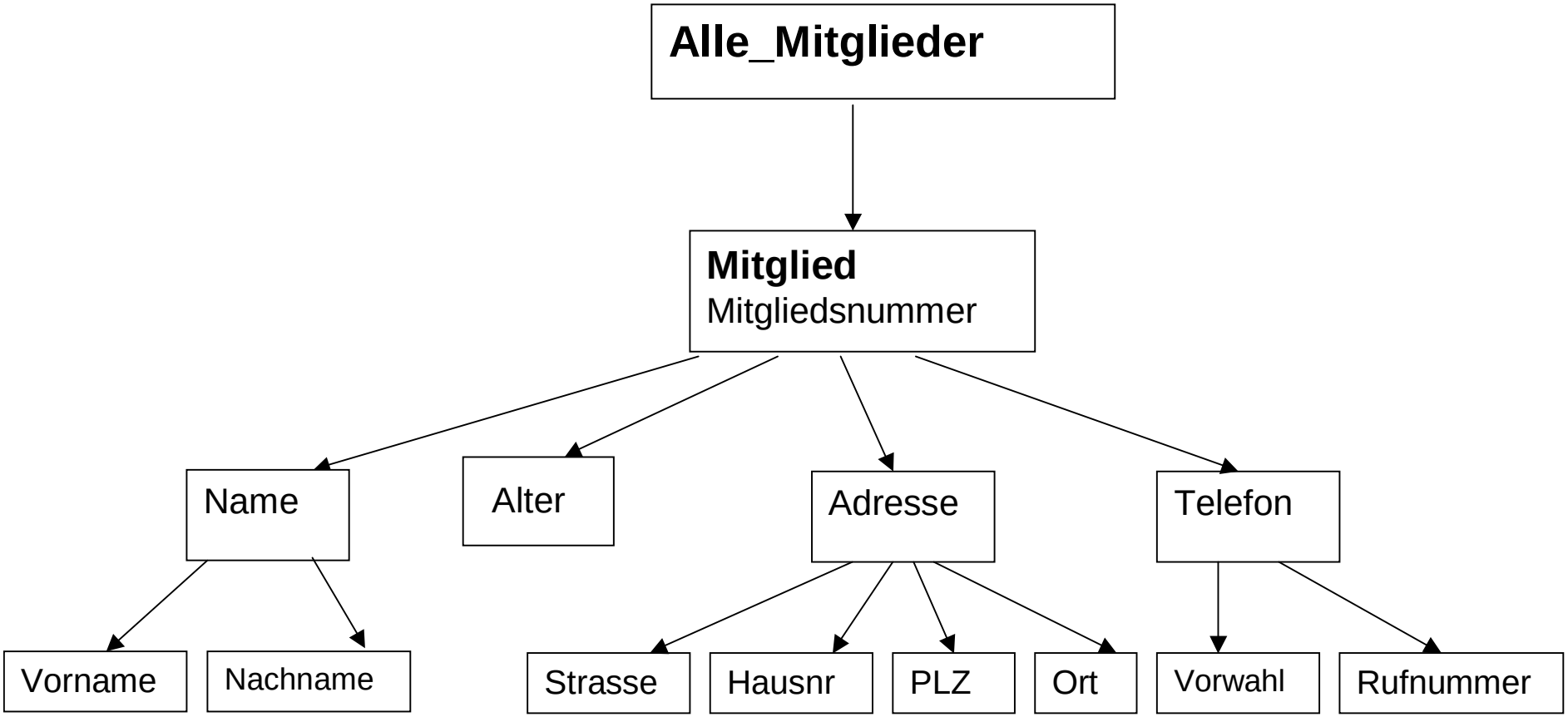
Gliederung

0 XML-Grundlagen
- XML-Dokumente
- XML-Schema

1 XQuery
- Einführung
- Verwendung
- Baum-Modell
- Funktionalitäten

2 Zusammenfassung

XML-Dokumente(1)



XML-Dokumente(2)

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE Alle_Mitglieder SYSTEM Alle_Mitglieder.dtd>

<Alle_Mitglieder>
    <Mitglied>
        <Mitglied Mitgliedsnummer="001">     -- Attribut
        <Name>
          <Vorname>Anja</Vorname>
          <Nachname>Rein</Nachname>
        </Name>
        ...
        ...
     </Mitglied >
    ...
    ...
</Alle_Mitglieder>
```

XML-Schema(1)

```
<xds:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema>
   <xsd:complexType name="Mitglied">
      <xsd:attribute name="Mitgliedsnummer" type="xsd:string">  -- Attribut
      <xsd:union>
         <xsd:complexType name="Name">
            ...
         </xsd:complexType name="Name">
         <xsd:simpleType>
            ....
         </xsd:simpleType>
       </xsd:union>
    </xsd:complexType name="Mitglied">
</xsd:schema>
```

Komplexe Typen

```
<xsd:complexType name=”Name”>
    <xsd:all>
        <xsd:element name=“Vorname“ type=“xsd:string“/>
        <xsd:element name=“Nachname“ type=“xsd:string“/>
    </xsd:all>
</xsd:complexType name=”Name”>
```

Einfache Typen

<xsd:**simpleType**>
    <xsd:restiktion base="xsd:integer">
    <xsd:minInclusive value="5">
    <xsd:maxInclusive value="100">
< /xsd:**simpleType**>

Gliederung

0 XML-Grundlagen
- XML-Dokumente
- XML-Schema

1 XQuery
- Einführung
- Verwendung
- Baum-Modell
- Funktionalitäten

2 Zusammenfassung

Einführung

- **XQuery und XPath**
- **strikte Typisierung**
- **funktionale Programmiersprache**
- **vollständige Anfragesprache**
- **vollständige Transformationssprache**

Gliederung

0 XML-Grundlagen
- XML-Dokumente
- XML-Schema

1 XQuery
- Einführung
- Verwendung
- Baum-Modell
- Funktionalitäten

2 Zusammenfassung

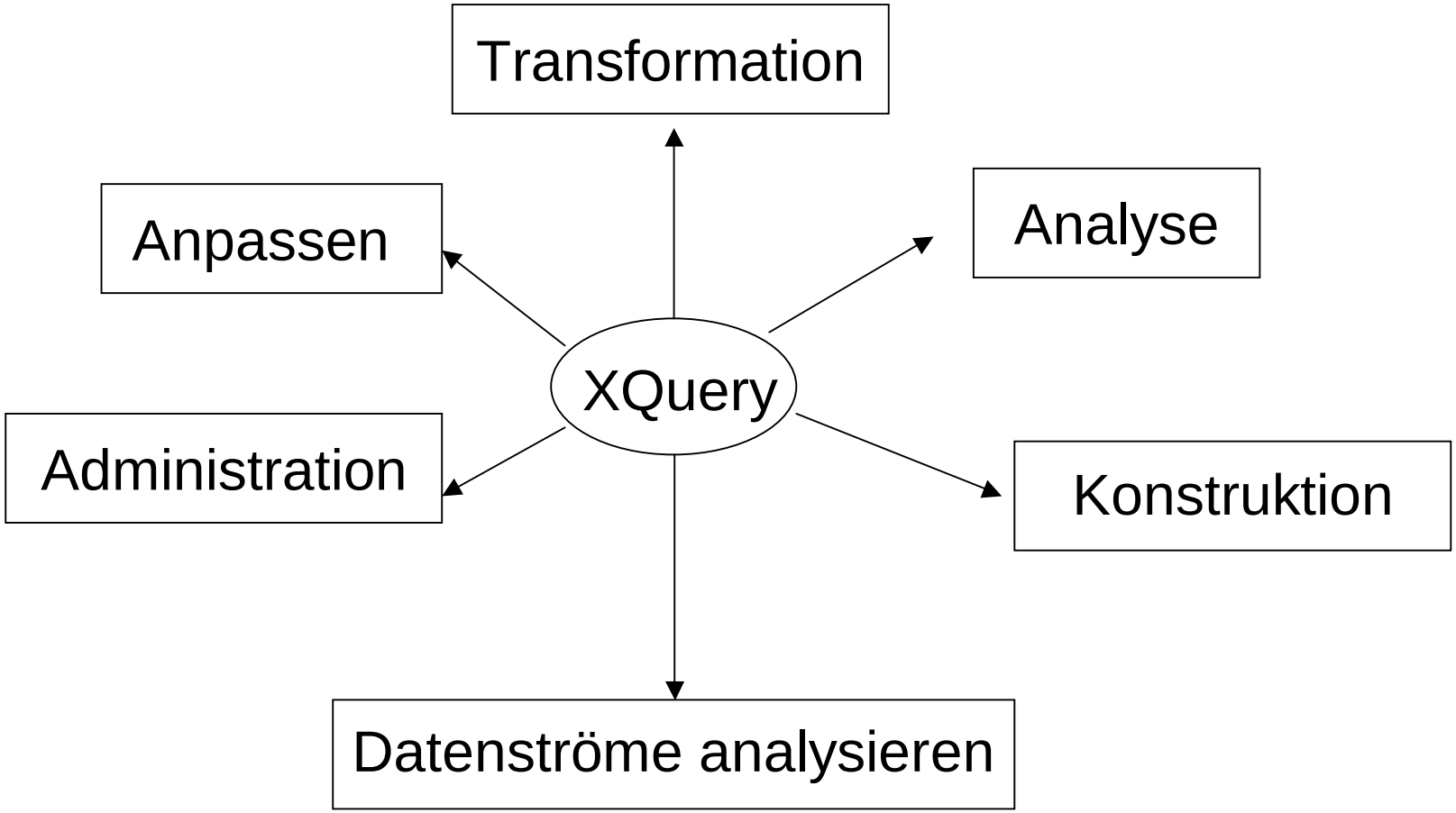
Verwendung – Warum?

- **Informationen aus XML extrahieren**
- **SQL ⇔ XQuery**

	SQL	**XQuery**
Daten	Strukturierte Daten	Semi-/Strukturierte Daten
Daten-manipulation	SELECT FROM WHERE	FLWR
Sortierung	im SELECT-Ausdruck	Sortby
Datenmodell	Tabellen	Baum

- **Kosten**

Verwendung – Wo?



Verwendung – Wie?

- XML-Dokument
- URL
- JSP
- String in Programmiersprache
- Protokoll

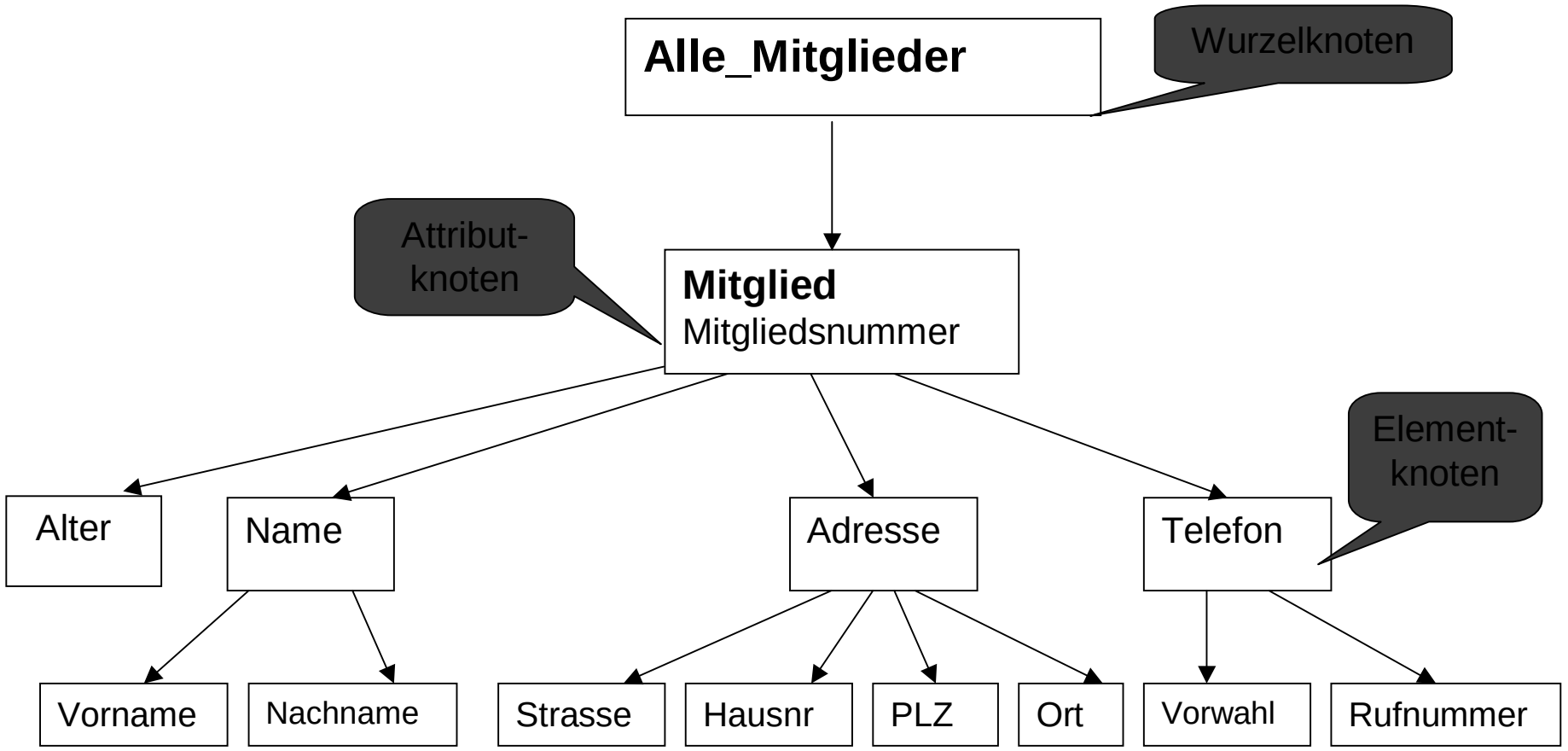
Gliederung

0 XML-Grundlagen
- XML-Dokumente
- XML-Schema

1 XQuery
- Einführung
- Verwendung
- Baum-Modell
- Funktionalitäten

2 Zusammenfassung

Baum-Modell



- Dokumentordnung
- Sequenz

Gliederung

0 XML-Grundlagen
- XML-Dokumente
- XML-Schema

1 XQuery
- Einführung
- Verwendung
- Baum-Modell
- Funktionalitäten

2 Zusammenfassung

Einfache Ausdrücke(1)

Konstanten

- elementare Werte
  (String, Integer, Double oder Float)

Variablen

- beginnt mit **$**-Zeichen
- Bindung an Ausdruck
- Typbestimmung vor der Auswertung
- Gültigkeit beschränkt sich auf den aktuellen und alle eingeschlossenen Anfrageausdrücke

Kommentar

{-- Das ist ein Kommentar **--}**

Arithmetik und logische Ausdrücke

Arithmetik

- +, -, *, DIV, MOD
- Klammerregeln der Algebra

Beispiel:

//alter+1

Logische Ausdrücke

- AND, OR, NOT

Beispiel:

NOT($mitglied/name [vorname="Anja"]==mitglied[@id="002"])

Vergleiche(1)

- **=, !=, <, <=, >, >=** (Sequenzen)
- **EQ, NE, LT, LE, LT, GT, GE** (elementare Werte)
- **==, !==** (Knotengleichheit überprüfen)
- **<<, >>** (Knotenabfolge bzgl. der Dokumentordnung)

Vergleiche(2)

Beispiele:

$mitglied[name="Anja Rein"]==$mitglied[@id="001"]

$mitglied[name="Anja Rein"]==$mitglied[@id="001"] ==
count($mitglied)

interessant:

$mitglied/name/vorname != „Anja“

gleich?

NOT($mitglied/name/vorname = „Anja Rein“)

Navigation mit Pfadausdrücken(1)

- Navigation im XML-Baum
- bestehen aus **Steps**
- relativer, absoluter Pfad

Pfadausdrücke:

.	Kontextknoten
..	Vaterknoten
/	Wurzelknoten oder trennt Pfadausdrücke
//	alle direkten Nachkommen des Kontextknotens
@attribut	Attribut des Kontextknotens
[n]	n-tes Element aus einer Sequenz
[expr1 TO expr2]	Teilsequenz vom ersten bis zweiten Ausdruck

Navigation in Pfadausdrücken(2)

Beispiele:

/mitglied[3]/name/nachname

/mitglied[last()]/name/nachname

/mitglied[1 TO 100]/name/nachname

Kontrollbeschreibung durch FLWR

- **F**OR/**L**ET/**W**HERE/**R**ETURN ⇔ SELECT-FROM-WHERE

- (FOR-Ausdruck | LET-Ausdruck)+
  WHERE-Ausdruck? RETURN-Ausdruck

- FOR ($Variablenname IN Ausdruck)+
  LET ($VariabelInname IN Ausdruck)+

Beispiel:

```
LET $mitgliedernamen := //name/nachname
        RETURN $mitgliedernamen
```

Bedingte Ausdrücke

- IF (Bedingung) THEN expr_1 ELSE expr_2

- ELSE optional

Beispiel:

```
LET $alter := /mitglied/alter
   IF ($alter < 21)
         THEN  $status := „Jugendlicher“
   ELSE  $status := „Erwachsener“
```

Verbunde(1)

- analog zu Join inSQL
- gleiche Werte bei gemeinsamen Elemente:
  natürlicher Verbund
- sonst: **karthesisches Produkt**

Verbunde(2)

```
<vorstandsmitgliederliste>
FOR $mitglied IN /
        $mname IN
                dokument(Vorstandsmitglieder.xml)//name/nachname
WHERE   $mitglied/name/nachname =  $mname
RETURN
    <vorstandsmitglied>
          <name>
               {$mitglied/name/nachname , $mitglied/name/vorname}
           </name>
          <adresse>
                ...
    </vorstandsmitglied>
</vorstandsmitgliederliste>
```

Sichten

- analog “view” bei relationalen Datenbanken
- Raster über Daten
- verwendet Funktionsdefinition

Beispiel:

define function alter() returns xs:integer{
    dokument(„Alle_Mitglieder.xml“)/mitglied/alter
}

Aufruf:

FOR $a **IN** alter() **WHERE** $a>65
    RETURN $a

Funktionsaufrufe

- Funktionsbibliothek
  - Typcastings
  - einfache Funktionen auf Sequenzen
    (z.B. last(), count())
- selbstdefinierte Funktionen

Weiterführende Funktionalitäten

- Sortieren
- Gruppieren
- Elementkonstruktoren
- All- und Existenzquantor
- Anfrageprolog

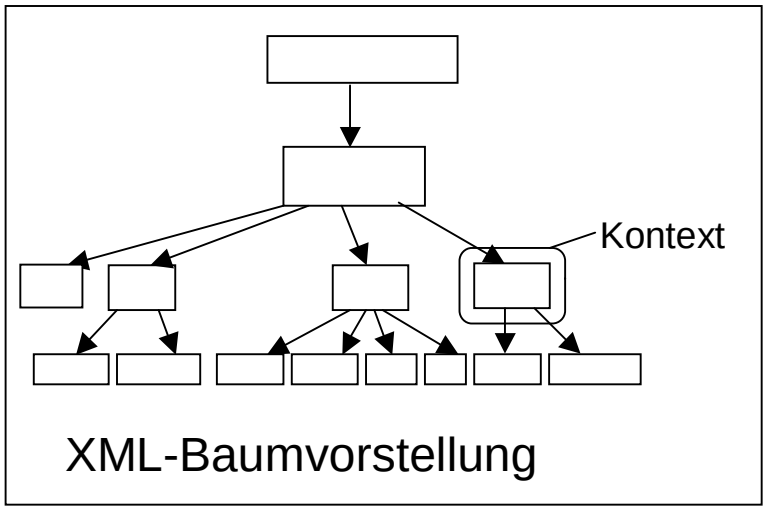
Gliederung

0 XML-Grundlagen
- XML-Dokumente
- XML-Schema

1 XQuery
- Einführung
- Verwendung
- Baum-Modell
- Funktionalitäten

2 Zusammenfassung

Zusammenfassung



XML-Baumvorstellung

Strenge Typisierung:

XML-Schema

F(x)

Funktionale Programmiersprache

SQL⇔XQuery