

XML-Manipulation in BaseX

Julia Beck | j.beck@ub.uni-frankfurt.de | @j4lib

11.12. und 18.12.2019



Praxislabor Digitale Geisteswissenschaften
Universitätsbibliothek Frankfurt am Main

[3]

Organisatorisches

Melden Sie sich für den moodle-Kurs **Praxislabor Digitale Geisteswissenschaften** an und schreiben Sie sich ein.

moodle

<https://moodle.studiumdigitale.uni-frankfurt.de/moodle/course/view.php?id=511>
[2]

Alle Materialien sind dort verlinkt:

- Diese Präsentation
- Links zu BaseX[1] und Spezifikationen (XQuery)
- Links zu den Beispieldaten, zu den Übungen und den Lösungen

Was ist das Lernziel?

- Was ist mit "meinen" XML-Daten möglich? (Was kann ich für Informationen aus ihnen ziehen? Wie kann ich sie bearbeiten/visualisieren/ändern?)
- Was ist eigentlich XQuery und wo ist der Unterschied zu anderen XML-Technologien?
- Welche Probleme kann ich mit XQuery oder XML-Datenbanken lösen?
- Diese Präsentation dient als Einblick in das Thema und als kleines Nachschlagewerk

Teil 1 (heute)

- Die XML Familie
- Die XML-Datenbank BaseX
- XQuery - Einführung
- XQuery - Sprachelemente
- XQuery - FLWOR
- XQuery - Konstruktion
- XQuery - Update

Teil 2 (nächste Woche)

- Übungen mit BaseX

Die XML Familie

Was ist was?

- XML = Extensible Markup Language
 - Auszeichnungssprache zur Darstellung hierarchisch strukturierter Daten (menschen- und maschinenlesbar);
andere XML-basierte Sprachen: TEI, SVG, GML, ...

XML (kurze Wiederholung)

Beispiel-XML

```
<collection>
  <book id="B1">
    <title>The Raven</title>
    <author>E.A. Poe</author>
  </book>
  <book id="B2">
    <title>The Empty House</title>
    <author>A.C. Doyle</author>
  </book>
</collection>
```

- Wurzelement (<collection>)
- Elemente (<book>, <author>, <title>)
- Attribut (@id)
- Text ("The Raven", "E.A. Poe", ...)

Was ist was?

- XML = Extensible Markup Language
Auszeichnungssprache zur Darstellung hierarchisch strukturierter Daten (menschen- und maschinenlesbar);
andere XML-basierte Sprachen: TEI, SVG, GML, ...
- XPath = XML Path Language
Abfragesprache zur Suche und Navigation, zum Testen und Filtern innerhalb von XML-Dokumenten

XPath (kurze Wiederholung)

- Abfragesprache zur Suche und Navigation, zum Testen und Filtern innerhalb von XML-Dokumenten
- In XPath ist es nicht möglich neue XML-Knoten zu erzeugen oder das XML-Dokument zu ändern
- Reihenfolge der Ergebnismenge entspricht der Reihenfolge im XML-Dokument ("document order")
- XPath ist Teil anderer XML-Technologien wie XQuery oder XSLT

XPath (kurze Wiederholung)

Mini-Übung

Titel aller Bücher

- `//title/text()` oder
- `/collection/book/title/text()`

Alle Ids

- `/collection/book/@id` oder
- `//@id bzw. data(//@id)`

Autor des Buches "The
Empty House"

- `/collection/book[title/text() = "The Empty House"]/author/text()`

```
<collection>
  <book id="B1">
    <title>The Raven</title>
    <author>E. A. Poe</author>
  </book>
  <book id="B2">
    <title>The Empty House</title>
    <author>A. C. Doyle</author>
  </book>
</collection>
```

Die XML Familie

Was ist was?

- XML = Extensible Markup Language
Auszeichnungssprache zur Darstellung hierarchisch strukturierter Daten (menschen- und maschinenlesbar); andere XML-basierte Sprachen: TEI, SVG, GML, ...
- XPath = XML Path Language
Abfragesprache zur Suche und Navigation, zum Testen und Filtern innerhalb von XML-Dokumenten
- XSD = XML Schema Definition
Schemasprache zur Definition und Validierung von XML-Strukturen

Was kann ich mit XSD ausdrücken?

Beispiel (von Wikipedia):

```
<celsiusKoerperTemp>37.0</celsiusKoerperTemp>
```

In XSD können Einschränkungen gemacht werden:

```
<xs:simpleType name="celsiusKoerperTemp">
  <xs:restriction base="xs:decimal">
    <xs:fractionDigits value="1"/>
    <xs:minInclusive value="35.0"/>
    <xs:maxInclusive value="42.5"/>
  </xs:restriction>
</xs:simpleType>
```

Der Wert soll den Typ "Dezimalzahl" haben.

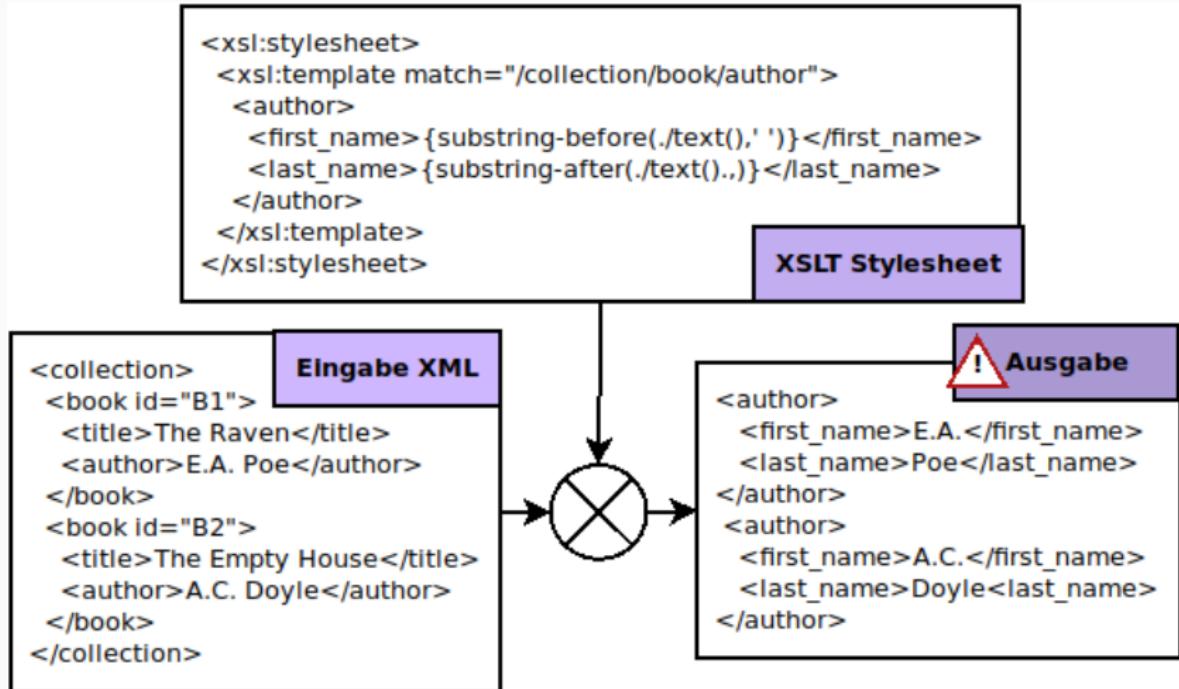
Der Wert soll eine Nachkommastelle haben.

Definition von Minimal- und Maximalwert.

Was ist was?

- XSLT = Extensible Stylesheet Language Transformation
Transformationssprache für XML-Dokumente

XSLT



Achtung nur Beispiel: das Ergebnisdokument wäre nicht valide, da kein Wurzelement vorhanden ist.

Was ist was?

- XSLT = Extensible Stylesheet Language Transformation
Transformationssprache für XML-Dokumente
- XQuery = XML Query Language
Abfragesprache für XML-Datenbanken
- ...

Die XML-Datenbank BaseX

Was ist eigentlich eine XML-Datenbank?

- Datenbank für XML-Dokumente, die dort gespeichert und durchsucht werden können
- dokumentenorientierte Datenbank (NoSQL)
- So wie man mit SQL relationale Datenbanken abfragt, fragt man XML-Datenbanken mit XPath und XQuery ab
- Möglichkeit zur Volltextsuche
- BaseX ist ein Beispiel für eine XML-Datenbank, es gibt noch andere :-)

- Open Source **XML-Datenbank** mit XQuery Prozessor
- erlaubt die Speicherung, Abfrage und Manipulation **großer Textdateien** (XML/JSON/CSV/...)
- enthält einen XQuery **Editor**, um XQuery in Echtzeit zu testen
- bietet verschiedene **Visualisierungsmöglichkeiten** für die Analyse der Daten
- sehr ausführliche **Dokumentation** mit vielen Beispielen

BaseX

C:/Users/user/Desktop/bases/src/factbook.xq [factbook] - BaseX 9.0

Database Editor View Visualization Options Help

Find Find...

Editor

stopwords.bx w3-catalog.xml factbook.xq

```
1 for $country in //country
2 where $country/@name = 'Italy'
3 return $country
```

Total Time: 5.68 ms

Compiling:

- pre-evaluate root to document-node(): root() -> db:open-pre("factbook", 0)
- rewrite descendant-or-self step(s)
- convert to child steps: descendant::*:country
- apply attribute index for "Italy"
- rewrite where clause(s)
- simplify gflwor

Optimized Query:

```
db:attribute("factbook", "Italy")/self::*:name/parent::*:country
```

Query:

```
for $country in //country where $country/@name = 'Italy' return $country
```

Result:

- Hits: 1 Item
- Updated: 0 Items
- Printed: 11 kB
- Read Locking: factbook
- Write Locking: (none)

Timing:

- Parsing: 0.88 ms

Result

```
<country id="f0_268" name="Italy" capital="f0_1544"
population="574689272" datacode="IT" total_area="3012300"
population_growth="0.13" infant_mortality="6.9" gdp_agri=
"2.0" gdp_total="1080000" inflation="5.4" indep_date=
"17 03 1861" government="republic" gdp_ind="31.6" gdp_
serv="65.5" car_code="I">
<name>Italy</name>
<city id="f0_3202" country="f0_268">
<name>Novara</name>
<population year="90">103349</population>
</city>
<city id="f0_3207" country="f0_268">
<name>Bergamo</name>
<population year="90">117886</population>
</city>
<city id="f0_3212" country="f0_268">
<name>Brescia</name>
<population year="90">196766</population>
</city>
<city id="f0_3226" country="f0_268">
<name>Padova</name>
```

Latitude

Longitude

mondial

A..	C..	G..	H..	S..	Swi..	Ve..	C..	A..	E..
B..	Fin..	Itali..		Uk..	B..	A..	Eg..	B.C.	Foo..
F..	Ge..			United Kin..			A..	C..	E..
N..	Russia	China	Ind..		M..		E.K..	C..	Gr..
Poland		G..	H..		Nig..	Z..	A..	E..	Inte..
P.Ro..		I..	J..	S..	A..	A..	G..	E..	Inte..
Iran	I..	P..	Ca..	H..	Ind..	Ind..	Afr..	Int..	Wor..
	M..				Int..	Int..			Wor..
Japan	T..	P..	Col..	Mex..	In..	In..			
Ka..	N..	Ta..	C..	El..	P..	L..	Int..	W..	W..
Tur..	U..	S..	P..	United St..	N..	U..		H..	A..
Brazil	P..	United St..	C..	D..	O..	U..			
Vil..	Ar..	S..	T..	P..	P..	U..			

88 MB

Hilfreiche Kommandos in BaseX:

- Erstellen einer Datenbank (geht auch über die GUI)

```
db:create("books")
```

- Öffnen einer Datenbank (geht auch über die GUI)

```
db:open("books")
```

- Dokument einer Datenbank hinzufügen

```
db:add("books", "/home/dir/meinDoc.xml")
```

- Optimieren einer Datenbank

```
db:optimize("books")
```

BaseX - Installation

BaseX benötigt die **Java Runtime Environment (JRE)** mindestens in der Version **Java 8**. (Sie können auch OpenJDK installieren.)

BaseX ist plattformunabhängig und wurde auf Windows, Mac OS X, Linux und OpenBSD getestet.

Bitte installieren Sie die JRE und BaseX bis nächste Woche auf Ihrem Laptop, wenn Sie die Übungen live mitmachen möchten.

Melden Sie sich, falls Sie Probleme bei der Installation haben.

Beispieldaten für die Übungen

Metadaten vom Städel Museum, die Teil des Coding da Vinci Rhein-Main waren. Sie sind unter CC0-Lizenz verfügbar.



Siehe Link im Moodle!

XQuery - Einführung

XQuery

- Abfragesprache für XML-Datenbanken; Dateiendung .xq
- ähnlich zu SQL für relationale Datenbanken
- streng typisierte, funktionale Programmiersprache
→ eine Eingabe liefert eine Ausgabe
- hat den Ruf einfacher zu lernen zu sein als XSLT ;-)

XQuery kann z.B. Aufgaben erfüllen wie:

- **Suche** alle Titel, die mit "A" beginnen.
- Suche alle Titel im Dokument und **sortiere** sie alphabetisch.
- **Zähle** die Anzahl **verschiedener** Autoren im Dokument und sage mir, bei wie vielen Büchern sie jeweils als Autor aufgeführt sind.
- **Erzeuge** ein XML-Dokument oder eine CSV-Tabelle, die jedem Autor, die Liste der von ihnen geschriebenen Büchern zuordnet.

XQuery - Sprachelemente

- grundlegende Datenstruktur ist eine **Sequenz**,
eine geordnete Liste, z.B.:

()

(1,3,2)

(<a/>,hallo,<c/>)

("The Raven","The Empty House")

- Literale, Variablen, Kommentare
 - (: Kommentar - Titel, die mit "T" beginnen :)
 - `declare variable $titles := //title;`
 - `$titles[starts-with(text(),"T")]`
- Ausdrücke, Funktionsaufrufe
 - `1 + 4 * 2`
 - `starts-with(text(),"T")`
- Boolesche Ausdrücke und Vergleiche
 - `1 < 3`
 - `1 < 3 and 2 > 5`

Bedingte Aussagen mit if/then/else:

```
if (starts-with($title/text(), "T"))
then $title/text()
else "Nope!"
```

(Hinweis: für weitere Sprachelemente, siehe auch switch,
Sequenzoperationen, Type Casts, FLWOR)

XQuery - FLWOR

XQuery - FLWOR

- **for** - Schleife (für alle x in xs)

```
for $x in $xs
```

- **let** - Variablendefinition

```
let $x := 5
```

- **where** - Einschränkung

```
where $x > 2
```

- **order by ... descending/ascending**

- Sortieren nach ... absteigend/aufsteigend

```
order by $x descending
```

- **return** - Rückgabe

```
return $x
```

XQuery - FLWOR

Beispiel XML - Variante (books2.xml in Datenbank "books"):

```
<collection>
  <book id="B1">
    <title>The Raven</title>
    <author>E. A. Poe</author>
  </book>
  <book id="B2">
    <title>The Empty House</title>
    <author>A.C. Doyle</author>
  </book>
  <book id="B3">
    <title>A Study in Scarlet</title>
    <author>A.C. Doyle</author>
  </book>
  <book id="B4">
    <title>The Dancing Men</title>
    <author>A.C. Doyle</author>
  </book>
</collection>
```

Minimalbeispiel for:

```
for $title in db:open("books")//title  
return $title/text()
```

Ausgabe:

The Raven

The Empty House

A Study in Scarlet

The Dancing Men

XQuery - FLWOR

Beispiel + let:

```
for $title in db:open("books")//title
let $t := $title/text()
return $t
```

Ausgabe:

The Raven

The Empty House

A Study in Scarlet

The Dancing Men

XQuery - FLWOR

Beispiel + where:

```
for $title in db:open("books")//title
let $t := $title/text()
where starts-with($t, "T")
return $t
```

Ausgabe:

The Raven

The Empty House

The Dancing Men

XQuery - FLWOR

Beispiel + order by:

```
for $title in db:open("books")//title
let $t := $title/text()
where starts-with($t, "T")
order by $t ascending
return $t
```

Ausgabe:

The Dancing Men
The Empty House
The Raven

XQuery - FLWOR

Beispiel author:

```
for $book in db:open("books")//book
let $author := $book/author/text()
return $author
```

Ausgabe:

E.A. Poe
A.C. Doyle
A.C. Doyle
A.C. Doyle

:-)

Beispiel group by:

```
for $book in db:open("books")//book
group by $author := $book/author/text()
return $author
```

Ausgabe:

E.A. Poe

A.C. Doyle

:-)

XQuery - Konstruktion

- Erinnerung: XPath kann nur gewünschte Teilmengen eines XML Dokuments zurückgeben, aber keine neuen Knoten erzeugen
- Mit XQuery kann man sein eigenes Ausgabe-XML bauen!

```
<label>{$title/text()}</label>
```

- oder falls der Elementname variabel ist:

```
element {"label" || $n} {$title/text()}
```

Beispiel Elemente erzeugen:

```
for $title in db:open("books")//title  
return <label>{$title/text()}</label>
```

Ausgabe:

```
<label>The Raven</label>  
<label>The Empty House</label>  
<label>A Study in Scarlet</label>  
<label>The Dancing Men</label>
```

XQuery - Konstruktion

Beispiel group by:

```
for $book in db:open("books")//book
group by $author := $book/author/text()
return
<author>
  <name>{$author}</name>
</author>
```

Ausgabe:

```
<author>
  <name>E. A. Poe</name>
</author>
<author>
  <name>A. C. Doyle</name>
</author>
```

Beispiel - group by + count:

```
for $book in db:open("books")//book
group by $author := $book/author/text()
return
<author>
  <name>{$author}</name>
  <bookCount>{count($book/title/text())}</bookCount>
</author>
```

XQuery - Konstruktion

Ausgabe:

```
<author>
  <name>E.A. Poe</name>
  <bookCount>1</bookCount>
</author>
<author>
  <name>A.C. Doyle</name>
  <bookCount>3</bookCount>
</author>
```

XQuery - Konstruktion

Beispiel - Verschachtelung:

```
for $book in db:open("books")//book
let $title := $book/title/text()
group by $author := $book/author/text()
return <author>
        <name>{$author}</name>
        {for $t in $title
          order by $t ascending
          return <wrote>{$t}</wrote>}
</author>
```

Ausgabe:

```
<author>
  <name>E.A. Poe</name>
  <wrote>The Raven</wrote>
</author>
<author>
  <name>A.C. Doyle</name>
  <wrote>A Study in Scarlet</wrote>
  <wrote>The Dancing Men</wrote>
  <wrote>The Empty House</wrote>
</author>
```

XQuery - Update

XQuery - Update

Mit Updating-Ausdrücken kann man z.B.:

- Knoten **einfügen**

```
insert node <type/> into $book
```

- bestimmte Knoten **löschen**

```
delete node //type
```

- Knoten oder ihre Werte durch andere **ersetzen**

```
replace node //type with <description/>
replace value of node //type with "book"
```

- Knoten **umbenennen**

```
rename node //type as "description"
```

Nach Updating-Ausdrücken immer **db:optimize("book")** ausführen!

XQuery - Update

Beispiel Elemente hinzufügen (insert node ... into ...):

```
for $book in db:open("books")//book
return if ($book/author/text() = "A.C. Doyle")
    then insert node
        <genre>Kriminalliteratur</genre>
        into $book,
db:optimize("books")
```

XQuery - Update

```
<collection>
  <book id="B1">
    <title>The Raven</title>
    <author>E.A. Poe</author>
  </book>
  <book id="B2">
    <title>The Empty House</title>
    <author>A.C. Doyle</author>
    <genre>Kriminalliteratur</genre>
  </book>
  <book id="B3">
    <title>A Study in Scarlet</title>
    <author>A.C. Doyle</author>
    <genre>Kriminalliteratur</genre>
  </book>
  <book id="B4">
    <title>The Dancing Men</title>
    <author>A.C. Doyle</author>
    <genre>Kriminalliteratur</genre>
  </book>
</collection>
```

XQuery - Update

Beispiel Wert ersetzen (replace value of node ... with "..."):

```
for $book in db:open("books")//book
let $author := $book/author
return if ($author/text() = "A.C. Doyle")
    then replace value of node $author
        with "Arthur Conan Doyle"
    else if ($author/text() = "E.A. Poe")
        then replace value of node $author
            with "Edgar Allan Poe"
    else (),
db:optimize("books")
```

XQuery - Update

Datenbank nach Update:

```
<collection>
  <book id="B1">
    <title>The Raven</title>
    <author>Edgar Allan Poe</author>
  </book>
  <book id="B2">
    <title>The Empty House</title>
    <author>Arthur Conan Doyle</author>
    <genre>Kriminalliteratur</genre>
  </book>
  <book id="B3">
    <title>A Study in Scarlet</title>
    <author>Arthur Conan Doyle</author>
    <genre>Kriminalliteratur</genre>
  </book>
  <book id="B4">
    <title>The Dancing Men</title>
    <author>Arthur Conan Doyle</author>
    <genre>Kriminalliteratur</genre>
  </book>
</collection>
```

Fragen?

References i

-  Basex.
[http://baseX.org/.](http://baseX.org/)
-  moodle.
[https://moodle.studiumdigitale.uni-frankfurt.de/moodle/course/view.php?id=511.](https://moodle.studiumdigitale.uni-frankfurt.de/moodle/course/view.php?id=511)
-  M. Vogelgesang.
Metropolis beamer theme.
Metropolis is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.