TU München, Fakultät für Informatik
Lehrstuhl III: Datenbanksysteme
Prof. Alfons Kemper, Ph.D.

# Übung zur Vorlesung *Einsatz und Realisierung von Datenbanksystemen* im SoSe18

Alexander van Renen, Maximilian E. Schüle (i3erdb@in.tum.de)
http://db.in.tum.de/teaching/ss18/impldb/

**Blatt Nr. 10**

**Hinweise** XQuery-Aufgaben können auf `http://xquery.db.in.tum.de/` getestet werden. Die Daten für das Unischema können mit doc('uni2') geladen werden. Zur Lösung der Aufgaben können Sie die folgenden XQuery-Funktionen verwenden:

1. max(NUMBERS) - Returns largest number from list
2. count(LIST) - Return the number of elements in the list
3. tokenize(STR,SEP) - Splits up the string at the seperator
4. sum(NUMBERS) - Returns sum of all numbers in list
5. contains(HAY,NEEDLE) - Checks if the search string (NEEDLE) is contained in the string (HAY)
6. distinct-values(LIST) - Returns the distinct values from the list

## Hausaufgabe 1 Reines XPath

Lösen Sie in XPath folgende Aufgaben und testen Sie diese auf `xquery.db.in.tum.de`.

1. Lassen Sie sich das gesamte Schema anzeigen.

```
doc('uni')
```

2. Finden Sie die Namen aller Fakultäten.

```
doc('uni')//FakName
```

3. Finden Sie die Namen aller Studenten, die Vorlesungen hören.

```
doc('uni')//Student[./hoert]/Name
```

4. Ermitteln Sie die Anzahl der Vorlesungen.

```
count(doc('uni')//Vorlesung)
```

5. Zählen Sie, in wie vielen verschiedenen Semestern die Studenten sind.

```
count(distinct-values(doc('uni')//Student/Semester))
```

## Hausaufgabe 2 Mehr mit XQuery

Lösen Sie mit XQuery folgende Anfragen und testen Sie diese auf `xquery.db.in.tum.de`.

1. Geben Sie eine nach Rang sortierte Liste der Professoren aus (C4 oben).

```
for $p in doc('uni')//ProfessorIn
   order by $p/Rang descending
   return $p
```

2. Finden Sie die Namen der Professoren, die die meisten Assistenten haben.

```
let $maxAssi := max(
   for $p in doc('uni')//ProfessorIn
      return count($p//Assistent)
   )
return doc('uni')//ProfessorIn[count(.//Assistent)=$maxAssi]/Name
```

3. Finden Sie für jede von einem Student gehörte Prüfung den Namen des Prüfers und Vorlesung.

```
<Studenten>
{
   let $pr := doc('uni')//Assistent union doc('uni')//ProfessorIn
   for $s in doc('uni')//Student
      return <Student>
      {$s/Name}
      <Pruefungen>
      {
      for $p in $s//Pruefung
         let $prName := $pr[./@PersNr=$p/@Pruefer]/Name
         let $vlTitel := doc('uni')
            //Vorlesung[./@VorlNr=$p/@Vorlesung]/Titel
         return <Pruefung Pruefer="{$prName}">{$vlTitel}</Pruefung>
      }
      </Pruefungen>
      </Student>
}
</Studenten>
```

**Hausaufgabe 3** XML gegeben, XQuery gesucht

Schreiben Sie eine Anfrage, die folgendes zurück gibt:

```
<Universitaet>
  <Fakultaet Name="Philosophie" AnzahlAssistenten="3">
    <Professor Name="Sokrates" AnzahlAssistenten="2"/>
    <Professor Name="Russel" AnzahlAssistenten="1"/>
  </Fakultaet>
  <Fakultaet Name="Physik" AnzahlAssistenten="2">
    <Professor Name="Kopernikus" AnzahlAssistenten="2"/>
  </Fakultaet>
  <Fakultaet Name="Theologie" AnzahlAssistenten="1">
    <Professor Name="Augustinus" AnzahlAssistenten="1"/>
  </Fakultaet>
</Universitaet>
```

```
<Universitaet>
{for $f in doc('uni')//Fakultaet
    let $fa := count($f//Assistent)
    order by $fa descending
        return <Fakultaet Name="{$f/FakName}" AnzahlAssistenten="{$fa}">{
            for $p in $f//ProfessorIn
            let $pa := count($p//Assistent)
            where $pa > 0
            order by $pa descending
        return <Professor Name="{$p/Name}" AnzahlAssistenten="{$pa}" />
    }</Fakultaet>
}
</Universitaet>
```

**Hausaufgabe 4** JSON

Überlegen Sie sich, wie Ihre Visitenkarte im JSON-Format aussähe und stellen Sie diese in der Übung vor.

```
{
  "vorname": "Maximilian",
  "nachname": "Schuele",
  "e-mail": "i3erdb@in.tum.de",
  "adresse": {
    "raum": "2.11.060",
    "strasse": "Boltzmannstr.␣3",
    "stadt": "Garching",
    "bundesland": "Bayern",
    "postleitzahl": "85748"
  },
  "Telenummern": [
    {
      "typ": "fon",
      "nummer": "089␣289␣17250"
    },
    {
      "typ": "fax",
      "nummer": "089␣289␣17263"
    }
  ]
}
```

**Hausaufgabe 5**

Vervollständigen Sie die untere Anfrage um die Namen der Freunde von Personen mit dem Vornamen *Sokrates* zu finden, die älter als 30 Jahre sind. Die *foaf* Onthology is unter `http://xmlns.com/foaf/spec/` beschrieben. Nutzen Sie `https://rdf.db.in.tum.de/` für Ihre Abfrage.

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?name2
WHERE {
          ....
      }
```

Lösung:

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?name2
WHERE {
         ?person1 foaf:knows ?person2 .
         ?person1 foaf:firstName "Sokrates" .
         ?person2 foaf:name  ?name2 .
         ?person22 foaf:name ?name2
         ?person22 foaf:age ?age .
         FILTER ( ?age > 30 )
      }
```

## Hausaufgabe 6

Berechnen Sie für folgende drei Dokumente die TF-IDF-Werte:

1. „Beim Fußball dauert ein Spiel neunzig Minuten – und am Ende gewinnen die Deutschen“
2. „Beim Fußball muss das Runde (der Ball) in das Eckige (das Tor)“
3. „Nie war ein Tor so wertvoll wie jetzt“

Welches Ranking ergibt sich gemäß der Relevanzwerte für die Anfrage: „Fußball“ $\wedge$ „Tor“. Zur Ermittlung des TF Wertes gehen sie davon aus, dass alle Wörter eines Dokuments *interessant* sind?
Zur Berechnung des Rankings reicht es nur die TF-IDF-Werte von *Fußball* und *Tor* zu berechnen.

**Fußball** IDF: 0.176

| | Dokument 1 | Dokument 2 | Dokument 3 |
|---|---|---|---|
| TF | 0.077 | 0.083 | 0 |
| TF-IDF | 0.014 | 0.015 | 0 |

**Tor** IDF: 0.176

| | Dokument 1 | Dokument 2 | Dokument 3 |
|---|---|---|---|
| TF | 0 | 0.083 | 0.125 |
| TF-IDF | 0 | 0.015 | 0.022 |

**Ranking** Dokument 2: 0.029
Dokument 3: 0.022
Dokument 1: 0.014