Chapter 7: XQuery

References:

- Scott Boag, Don Chamberlin, Mary F. Fernńndez, Daniela Florescu, Jonathan Robie, Jérôme Siméon (Eds.):
  XQuery 1.0: An XML Query Language.
  W3C Recommendation 23 January 2007.
  [http://www.w3.org/TR/xquery/]
- Wolfgang Lehner, Harald Schöning:
  XQuery. Grundlagen und fortgeschrittene Methoden.
  dpunkt.verlag, 2004, ISBN 3-89864-266-6, 290 Seiten.
  [http://www.xquery-buch.de/]
- Howard Katz (Ed.), Don Chamberlin, Denise Draper, Mary Fernández, Michael Kay, Jonathan Robie, Michael Rys, Jérôme Siméon, Jim Tivy, Philip Wadler:
  XQuery from the Experts. A Guide to the W3C XML Query Language.
  Pearson Education Inc., 2004, ISBN 0-321-18060-7, 484 pages.
- Jim Melton, Stephen Buxton:
  Querying XML: XQuery, XPath, and SQL/XML in Context.
  Morgan Kaufmann/Elsevier, 2006, ISBN 1-55860-711-0, 815 pages.
- Rudolf Jansen:
  XQuery: Eine praxisorientierte Einführung.
  Software & Support Verlag GmbH, 2004, ISBN 3-935042-65-5, 167 Seiten.

Objectives

After completing this chapter, you should be able to:

- read and understand queries in XQuery.
- write queries to XML data in XQuery.
- enumerate and explain the clauses of FLWOR expressions.
- explain the use of direct and computed constructors.
- compare XQuery with SQL.

Overview

Introduction (1)

- One can view XML as a data model, and every data model should have a query language.
- XPath permits to
  - ◇ select nodes in a given XML document, and to
  - ◇ compute values from the data in a document,

  but it does not permit to generate new nodes.
- An XML query language should be able to produce new XML documents as result.

  I.e. to transform given XML documents into documents that are differently structured, contain only a subset of the data, or additional derived data.

Introduction (2)

- Joins are very limited in XPath (semijoins) or can be formulated only procedurally (with `for`-loops).

  Note that many powerful constructs such as `for`-loops got into XPath only in Version 2.0 during the development of XQuery. Although today, XPath is sufficient for quite a lot of queries, the original XPath was much more restricted.

- One cannot sort in XPath.
- Of course, a good XML query language must be at least as powerful as SQL.

Introduction (3)

- The XML format is a common interface to a lot of different data sources (documents, relational databases, object repositories).

  The data might be physically stored as XML, or might be only viable as XML via a middleware.

- An XML query language permits to combine data from different sources.

  This integrating function of an XML query language is natural and important. While also SQL can be used in distributed databases, and there exist relational interfaces to non-relational data, this is much more vendor-dependent (and typically expensive).

History (1)

- In December 1998, the W3C organized a workshop about query languages for XML.

  [http://www.w3.org/TandS/QL/QL98/].

- There was a lot of research about query languages for semi-structured data models and XML in particular (e.g., Lorel, XQL, XML-QL, YATL, Quilt).

  See, e.g.: XML Query Langauges, Experiences and Exemplars.
  [http://www.w3.org/1999/09/ql/docs/xquery.html]

- The XML Query Working Group started in 1999 the work on a W3C standard XML query language.

  [http://www.w3.org/XML/Query]

History (2)

- XPath 1.0 and XSLT 1.0 became a W3C Recommendation in November 1999.
- While it seemed natural that XPath-like expressions should be used also in XQuery, the XQuery committee had quite different ideas for the exact details of syntax and semantics.

  XPath 1.0 came from the document processing community, not from the database community. But having two similar languages that differed in important details was obviously not good. This lead to difficult negotiations and ultimately the development of XPath 2.0.

History (3)

- XML Schema became a W3C recommendation in May 2001.
- Steps of the XQuery standardization:
  - ◇ First Working Draft: February 15, 2001.
  - ◇ Last Call Working Draft: November 12, 2003

    The last call period ended on February 15, 2004. Several updates were published afterwards.
  - ◇ W3C Candidate Recommendation: Nov. 3, 2005

    Update: June 8, 2006
  - ◇ W3C Proposed Recommendation: Nov. 21, 2006
  - ◇ W3C Recommendation: January 23, 2007

History (4)

- The following eight documents were developed together:
  - ◇ XQuery 1.0
  - ◇ XQueryX 1.0: XML Syntax for XQuery 1.0
  - ◇ XPath 2.0
  - ◇ XSLT 2.0
  - ◇ XQuery 1.0/XPath 2.0 Data Model
  - ◇ XQuery 1.0/XPath 2.0 Formal Semantics
  - ◇ XQuery 1.0/XPath 2.0 Functions and Operators
  - ◇ XSLT 2.0/XQuery 1.0 Serialization

History (5)

- Related documents (Working Group Notes):
  - ◇ XML Query Use Cases

    [http://www.w3.org/TR/xquery-use-cases/]
  - ◇ XML Query (XQuery) Requirements

    [http://www.w3.org/TR/xquery-requirements/]
- Extensions (Candidate Rec. May/Aug. 2008):
  - ◇ XQuery and XPath Full Text 1.0

    [http://www.w3.org/TR/xpath-full-text-10/]
  - ◇ XQuery Update Facility 1.0

    [http://www.w3.org/TR/xquery-update-10/]

XQuery vs. XSLT

- The two languages have overlapping, but not identical goals.
- XSLT was developed by the document processing community. Main use: rendering XML documents.

  Although it can also be used for selecting and restructuring data.
- XQuery is a database language.
- Databases store very many / very large documents: indexes and query optimization are important (the data does not fit completely into main memory).

  The data is also more regularly structured (in most cases).

XQuery Implementations (1)

- IPSI XQ

  Written in Java. [http://sourceforge.net/projects/ipsi-xq]
  [http://www.ipsi.fraunhofer.de/oasys/projects/ipsi-xq/]

- AltovaXML

  The engine used in XMLSpy is free (contains validator: DTD/Schema, XSLT 1.0/2.0, XQuery). [http://www.altova.com/altovaxml.html]

- Galax

  Open source, from some authors/editors of the XQuery Specification.
  [http://www.galaxquery.org/]

- eXist (open source native XML database)

  [http://exist.sourceforge.net/]
  Online demo: [http://demo.exist-db.org/sandbox/sandbox.xql]

XQuery Implementations (2)

- X-HIVE

  Commercial XML-DBMS, Online demo evaluator.
  [http://support.x-hive.com/xquery/] (select first any demo, then "your own"). [http://support.x-hive.com/xquery/basicservlet?demo=demo0&xquery=xquery&todo=showframes]

- Saxon (from Michael Kay)

  M. Kay is editor of the XSLT 2.0 Spec. Basic version (without static type checking and XQuery→Java compiler) is open source. Supports XSLT 2.0, XPath 2.0, XQuery 1.0. [http://saxon.sourceforge.net/]

- Qizx/open (open source Java implementation)

  In Java. Limited version is free. [http://www.axyana.com/qizxopen/]
  Online demonstration:
  [http://www.xmlmind.com:8080/xqdemo/xquery.html]

Example Document (1)

STUDENTS

SID	FIRST	LAST	EMAIL
101	Ann	Smith	···
102	Michael	Jones	(null)
103	Richard	Turner	···
104	Maria	Brown	···

EXERCISES

CAT	ENO	TOPIC	MAXPT
H	1	Rel. Algeb.	10
H	2	SQL	10
M	1	SQL	14

RESULTS

SID	CAT	ENO	POINTS
101	H	1	10
101	H	2	8
101	M	1	12
102	H	1	9
102	H	2	9
102	M	1	10
103	H	1	5
103	M	1	7

Example Document (2)

- Translation to XML with data values in elements:

```
<?xml version='1.0' encoding='ISO-8859-1'?>
<GRADES-DB>
  <STUDENTS>
    <STUDENT>
      <SID>101</SID>
      <FIRST>Ann</FIRST>
      <LAST>Smith</LAST>
    </STUDENT>
    ...
  </STUDENTS>
  ...
</GRADES-DB>
```

First Example

- Print all results for Homework 1:

```
for $s in /GRADES-DB/STUDENTS/STUDENT,
    $r in /GRADES-DB/RESULTS/RESULT
where $s/SID = $r/SID and
      $r/CAT = 'H' and
      $r/ENO = 1
return <h1>
         { $s/FIRST, $s/LAST, $r/POINTS }
       </h1>
```


Overview

Overall Syntax (1)

- The basic XQuery unit is a module.
- A module can be
  - ◇ a library module (contains mainly function declarations),
  - ◇ a main module (contains mainly the query).
- Each module may optionally start with a version declaration:

```
xquery version "1.0";
```

  One can also specify the encoding, but the treatment of this is implementation-dependent: `xquery version "1.0" encoding "utf-8";`

Overall Syntax (2)

- A main module consists of a prolog (which can be empty) and the query ("QueryBody").
- A library module consists of a module declaration and a prolog.
- The prolog can contain
  - ◇ First an arbitrary sequence of namespace declaractions, module import commands (for schemas and modules), and XQuery parameter settings,
  - ◇ and then an arbitrary sequence of variable, function, and option declarations.

Overall Syntax (3)

- The query itself ("QueryBody") is an expression.
- XPath-expressions are also XQuery-expressions.

  However, the grammar in the XQuery standard completely defines expressions. Basically, XPath is a restricted version of XQuery. Since XQuery has extensions in many places, it was not possible to simply embedd an XPath expression as defined in the XPath standard.

- As in XPath, all data values are sequences of items, where items are atomic values or nodes.
- Expressions can be arbitrarily nested.

  While only recent SQL DBMS support the use of an SQL query with one result value as a term, the arbitrary nesting was a basic design principle in XQuery. It is sometimes called a funtional language.

Expressions (1)

- On the top level, an expression consists of one or more subexpressions ("ExprSingle") separated by "," (sequence concatenation operator).
- On the next level, an expression ("ExprSingle") is
  - ◇ a FLWOR-expression,
  - ◇ a quantified expression (`some`, `every`)
  - ◇ a typeswitch expression (see below)
  - ◇ an `if`-expression,
  - ◇ or an expression with the usual logical, comparison and arithmetic operators (see below).

Expressions (2)

- In comparison, the `XPath 2.0` grammar has
  - ◇ a `for`-expression instead of the `FLWOR`-expression,
  - ◇ no `typeswitch` expression.
- Note that the `for`-expressions in XPath 2.0 are valid FLWOR-expressions in XQuery:
  - ◇ They have only the `for` and the `return` part.

    It is legal in XQuery to leave out the other parts.

  - ◇ The `for`-clause is simplified: XQuery permits to declare a type for the variable, and to add a positional variable (see below).

Operator Precedences (1)

Prio	Operator	Assoc.
1	, (comma)	left
2	:= (assignment)	right
3	for, some, every, typeswitch, if	left
4	or	left
5	and	left
6	eq,ne,lt,le,gt,ge,=,!=,<,<=,>,>=,is,<<,>>	left
7	to	left
8	+, -	left
9	*, div, idiv, mod	left
10	union, |	left

(continued on next slide)

Operator Precedences (2)

(continued from previous slide)

Prio	Operator	Assoc.
11	`intersect`, `except`	left
12	`instance of`	left
13	`treat`	left
14	`castable`	left
15	`cast`	left
16	- (unary), + (unary)	right
17	?, *, + (Occurrence Indicators)	left
18	/, //	left
19	[ ], ( ), {}	left

Only differences (additions) to XPath: `:=`, `typeswitch`.

Namespaces

- Namespaces can be defined in the Prolog:
  - ◇ `declare namespace Prefix = "URI";`
  - ◇ `declare default element namespace "URI";`
  - ◇ `declare default function namespace "URI";`
- The following namespace prefixes are predeclared:
  - ◇ `xml = http://www.w3.org/XML/1998/namespace`
  - ◇ `xs = http://www.w3.org/2001/XMLSchema`
  - ◇ `xsi = http://www.w3.org/2001/XMLSchema-instance`
  - ◇ `fn = http://www.w3.org/2005/xpath-functions`
  - ◇ `local = http:// www.w3.org/2005/xquery-local-functions`