Dotazování nad XML, XML databáze

Obsah

Dotazovací jazyky pro XML 1
Dotazování nad XML 1
XQuery 2
Charakteristika 2
Charakteristika (2) 2
Kde se XQuery použije (a nepoužije) 2
Příklad - zdrojový dokument 2
Příklad - jednoduchý dotaz XPath 3
Příklad - spuštění v Saxon 8.4b 3
Příklad - výsledek 4
XQuery konstrukce 4
FLWOR 4
FLWOR - jednoduchý příklad 5
Implementace XQuery 5
SAXON od verzí 7.x 5
Nativní XML databáze 6
Základní problémy efektivního ukládání a zpracování XML dat 6
Základy efektivního ukládání XML dat 6
Rozhraní pro práci s XML databázemi 6
Rozhraní XML:DB 6
Vrstvy XML:DB API 7
Ukázka XML:DB programu 7
Použití XUpdate v databázích s XML:DB 8
Implementace XML:DB rozhraní 8
Apache Xindice 8
Ukázka interakce s Xindice 8
Ukázka interakce s Xindice (2) 9
Ukázka interakce s Xindice (3) 9
eXist 10
eXist: instalace a spuštění 10
eXist: použití přes webové rozhraní 10
eXist: vložení dokumentu do kolekce 10
eXist: dotazování - zadání dotazu 11
eXist: dotazování - sumarizovaný výsledek dotazu 11
eXist: dotazování - prohlížení jednotlivých výsledků dotazu 11

Dotazovací jazyky pro XML

Dotazování nad XML

../../../PB138-2003-XML.Databaze.pdf

XQuery

Charakteristika

- Jazyk pro specifikaci dotazů k vyhledání a extrakci uzlů XML (elementy, atributy) z dokumentů a konstrukci výstupního XML dokumentu.

Charakteristika (2)

- V současnosti (a zdá se i budoucnosti) je XQuery základním dotazovacím jazykem nad XML dokumenty.
- Definován konsorciem W3C, stane se specifikací - momentálně ve stavu *Last Call Working Draft*, viz http://www.w3.org/XML/Query.
- Založen na XPath 2.0 datovém modelu, operátorech a funkcích
- Podporují ho hlavní producenti databázových strojů (IBM, MS, Oracle a další)

Kde se XQuery použije (a nepoužije)

XQuery se typicky použije např. pro:

- dotazy, kde je složitější extrakční (selekční) část a jednodušší konstrukční část
- v opačném případě je lépe použít XSLT
- nebo dokonce zpracování obecnějším programovacím prostředím - např. manipulaci s (DOM) objektovým stromem dokumentu.

Příklad - zdrojový dokument

Ukázka zdrojového dokumentu, XQuery dotazů nad nimi a jejich výsledku.

Příklad 1. Zdrojový dokument

```
<?xml version="1.0" encoding="Windows-1250"?>
<addressbook>
    <person category="friends">
        <firstname>Petr</firstname>
```

```
        <lastname>Novák</lastname>
        <date-of-birth>1969-05-14</date-of-birth>
        <email>novak@myfriends.com</email>
        <characteristics lang="en">Very good friend</characteristics>
    </person>
    <person category="friends">
        <firstname>Jaroslav</firstname>
        <lastname>Nováèek</lastname>
        <date-of-birth>1968-06-14</date-of-birth>
        <email>novacek@myfriends.com</email>
        <characteristics lang="en">Another good friend</characteristics>
    </person>
    <person category="staff">
        <firstname>Jan</firstname>
        <lastname>Horák</lastname>
        <date-of-birth>1970-02-01</date-of-birth>
        <email>horak@mycompany.com</email>
        <characteristics lang="en">Just colleague</characteristics>
    </person>
    <person category="friends">
        <firstname>Erich</firstname>
        <lastname>Polák</lastname>
        <date-of-birth>1980-02-28</date-of-birth>
        <email>erich@myfriends.com</email>
        <characteristics lang="en">Good friend</characteristics>
    </person>
</addressbook>
```

Příklad - jednoduchý dotaz XPath

Ukázka XQuery dotazu nad výše uvedeným zdrojovým dokumentem. Úloha: "extrakce všech příjmení v adresáři".

Dotaz je v podstatě XPath výrazem - vybere tedy všechny elementy `lastname`.

Příklad 2. XQuery

```
doc('myaddresses.xml')/addressbook/person/lastname
```

Příklad - spuštění v Saxon 8.4b

XSLT procesor Saxon je od verzí 8.x rovněž XQuery procesorem. K vykonání XQuery dotazu je třeba:

- instalovat Saxon, např. 8.4b ("b" značí open-source větev) rozbalením do adresáře - např. `c:\devel\saxon8.4b`WIKIPEDIA [http://cs.wikipedia.org/wiki/Speci%C3%A1ln%C3%AD:Search?search=c:\devel\saxon8.4b]
- přepnout se do tohoto adresáře: **`cd c:\devel\saxon8.4b`**WIKIPEDIA **[http://cs.wikipedia.org/wiki/Speci%C3%A1ln%C3%AD:Search?search=c:\devel\saxon8.4b]** WIKIPEDIA [http://cs.wikipedia.org/wiki/Speci%C3%A1ln%C3%AD:Search?search=cd]
- uložit výše uvedený dotaz např. do souboru `lastnames.xq`WIKIPEDIA [http://cs.wikipedia.org/wiki/Speci%C3%A1ln%C3%AD:Search?search=lastnames.xq].
- výše uvedený dokument s "addressbook" uložíme do `myaddresses.xml`WIKIPEDIA [http://cs.wikipedia.org/wiki/Speci%C3%A1ln%C3%AD:Search?search=myaddresses.xml] ve stejném adresáři.
- z příkazové řádky spustit: **java -classpath saxon8.jar net.sf.saxon.Query -o result.xml lastnames.xq**WIKIPEDIA [http://cs.wikipedia.org/wiki/Speci%C3%A1ln%C3%AD:Search?search=java -classpath saxon8.jar net.sf.saxon.Query -o result.xml lastnames.xq]

Příklad - výsledek

Uvedený XQuery dotaz vrátí do souboru `result.xml`WIKIPEDIA [http://cs.wikipedia.org/wiki/Speci%C3%A1ln%C3%AD:Search?search=result.xml] nad zmíněným dokumentem toto:

Příklad 3. Výsledek aplikace dotazu

```
<lastname>Novák</lastname>
<lastname>Nováček</lastname>
<lastname>Horák</lastname>
<lastname>Polák</lastname>
```

XQuery konstrukce

FLWOR

FLWOR je zkrácené označení pro strukturu XQuery dotazů.

Je to akronym z:

(F)or — Úvodní část dotazu specifikuje cyklus vč. řídicí proměnné, do níž jsou postupně přiřazovány jednotlivé hodnoty vybrané XPath výrazem za klíčovým slovem "in".

(L)et — V této sekci lze provést přiřazení do dalších proměnných použitelných následně.

(W)here — Specifikuje selekční podmínku, tzn. které uzly (hodnoty) vybrané ve "for" budou skutečně použity.

V podmínce lze použít i proměnné vázané v sekci "let".

(O)rder — Takto vybrané uzly (hodnoty) lze výrazem v této sekci uspořádat.

(R)eturn — Co bude vráceno, zkonstruováno ze získaných uzlů (hodnot).

FLWOR - jednoduchý příklad

Selekci uzlů lze specifikovat buďto přímo v XPath výrazu ve "for" nebo až v selekčním "where".

"Vrať datum narození Poláka."

Příklad 4. FLWOR

```
for $person in doc('myaddresses.xml')/addressbook/person
where $person/lastname='Polák'
return $person/date-of-birth
```

Spuštění vrátí:

```
<?xml version="1.0" encoding="UTF-8"?>
<date-of-birth>1980-02-28</date-of-birth>
```

Implementace XQuery

SAXON od verzí 7.x

- instalovat (rozbalit) Saxon od verze 7.x (lze i 8.x) rozbalením do libovolného adresáře
- přepnout se do tohoto adresáře a
- z příkazové řádky spustit: **java -classpath saxon8.jar net.sf.saxon.Query -o result.xml query.xq** WIKIPEDIA The Free Encyclopedia [http://cs.wikipedia.org/wiki/Speci%C3%A1ln%C3%AD:Search?search=java -classpath saxon8.jar net.sf.saxon.Query -o result.xml query.xq]

Nativní XML databáze

Nativní XML databázové systémy často podporují dotazování přes XQuery.

Patří mezi ně např.:

- Berkeley DB XML [http://www.sleepycat.com/products/index.shtml]
- eXist [http://exist.sourceforge.net/]

Základní problémy efektivního ukládání a zpracování XML dat

Základy efektivního ukládání XML dat

Přednáška Petra Adámka (PDF) [../../../PB138-2003-XML.Databaze.pdf]

Rozhraní pro práci s XML databázemi

Rozhraní XML:DB

- Specifikováno konsorciem XML:DB [http://xmldb.org]
- Podobná koncepce jako JDBC [http://java.sun.com/products/jdbc/]
- Rozhraní je specifikováno na poměrně abstraktní úrovni, implementační detaily jsou skryty.
- Základní objekty:
  - `Driver` - podobně jako JDBC Driver - abstrahuje přístup ke konkrétnímu DBS, implementuje rozhraní Database
  - `DatabaseManager` - řídí zavádění a správu jednotlivých ovladačů (Driver) databázových systémů
  - `Collection` - kolekce XML dokumentů v databázi. Konceptuálně srovnatelné s relační tabulkou (či celou databází). Kolekce totiž mohou být libovolně vnořené.
  - Services - rozhraní konkrétních služeb. Bez nich by XML:DB takřka nemělo smysl - teprve služby definují, co databáze „umí“. Typickou službou je např. `XPathQueryService` na vyhledávání dokumentů a jejich částí přes XPath. Další službou je např. `XUpdateQueryService`.
  - `Resource` - zhruba odpovídá JDBC resource. Obecně „nějaký“ zdroj - nemusí být jen XML, ale i binární. Je-li XML, pak např. SAX, DOM, XML text...

Vrstvy XML:DB API

Rozhraní XML:DB je pro pohodlí programátora členěno do úrovní. Vždy si jednu z nich vybereme a využíváme její nabídky:

- XML:DB Core Level 0 - musí implementovat všechny DBS. Obsahuje základní rozhraní pro *kolekce* (collections), *zdroje* (resources), and *služby* (services).
- XML:DB Core Level 1 - navíc obsahuje `XPathQueryService`.

Ukázka XML:DB programu

Příklad 5. Příklad programu využívajícího XML:DB

```
import org.xmldb.api.base.*;
import org.xmldb.api.modules.*;
import org.xmldb.api.*;
public class Query {
    public static void main(String[] args) throws Exception {
        Collection col = null;
        try {
            String driver = null;
            String prefix = null;

            if ( ( args.length == 1 ) && args[0].equals("dbxml") ) {
                driver = "org.dbxml.client.xmldb.DatabaseImpl";
                prefix = "xmldb:dbxml:///db/";

            } else {
                driver = "org.xmldb.api.reference.DatabaseImpl";
                prefix = "xmldb:ref:///";
            }

            Class c = Class.forName(driver);
            Database database = (Database) c.newInstance();

            if ( ! database.getConformanceLevel().equals("1") ) {
                System.out.println("This program requires a Core Level 1 XML:DB "
                System.exit(1);
            }
            DatabaseManager.registerDatabase(database);

            col = DatabaseManager.getCollection(prefix + "addresses");
            String xpath = "/address[@id = 1]";
```

```
            XPathQueryService service = (XPathQueryService) col.getService("XPathQ

            ResourceSet resultSet = service.query(xpath);
            ResourceIterator results = resultSet.getIterator();

            while (results.hasMoreResources()) {
                Resource res = results.nextResource();
                System.out.println((String) res.getContent());
            }
        } catch (XMLDBException e) {
            System.err.println("XML:DB Exception occurred " + e.errorCode + " " +
        } finally {
            if (col != null) { col.close(); }
        }
    }
}
```

Použití XUpdate v databázích s XML:DB

Příklad 6. Příklad modifikace pomocí XUpdate

```
<xupdate:modifications version="1.0" xmlns:xupdate="http://www.xmldb.org/xupdate">
   <xupdate:update select="/address[@id = 1]/name/last">Herman</xupdate:update>
</xupdate:modifications>
```

Tento XUpdate dotaz nahradí příjmení za „Herman“.

Implementace XML:DB rozhraní

Apache Xindice

- Open-source referenční implementace XML:DB.
- Původně nazývána dbXML.
- Velmi dobré úvodní informace lze získat v Introduction to Xindice [http://www-106.ibm.com/developerworks/web/library/wa-xindice.html]

Ukázka interakce s Xindice

Kolekce mohou nebo nemusí mít XML Schema.

Příklad 7. Vytvoření a manipulace s kolekcí „mojedok“

```
xindiceadmin ac -a /db -n mo
```

systém odpoví

```
Created: /db/mojedok
```

dotaz na seznam kolekcí (Listing Collection - lc) v `/db`

```
xindiceadmin lc -c /db
```

zrušení kolekce (Delete Collection - dc) v `/db`

```
xindiceadmin dc -c /db -n mojedok
```

Ukázka interakce s Xindice (2)

Příklad 8. Přidání, získání, zrušení dokumentu do/z kolekce „mojedok“

Přidání souboru (-Add Document, -File [filename], -n specifikuje klíč) :

```
xindice ad -c /db/mojedok -f c:/devel/mojedokumenty/md.xml -n mujklic
```

Získání dokumentu zpět (-Retrieve Document, -File)

```
xindice rd -c /db/mojedok -f c:/devel/mojedokumenty/md.out.xml
```

Zrušení dokumentu z databáze (-Delete Document)

```
xindice dd -c /db/mojedok -n mujklic
```

Ukázka interakce s Xindice (3)

Příklad 9. Dotazování na kolekci „mojedok“

zrušení kolekce (Delete Collection - dc) v `/db`

```
xindice xpath -c /db/mojedok -q /parts/part[@sku="101"]
```

eXist

`eXist` je podobně jako `Xindice` open-source databáze podporující XML:DB.

Je možné ji provozovat jako:

- samostatně běžící (standalone) server, přístupný soketovým spojením (XML-RPC, HTTP)
- jako in-process (embedded) -server běžící v témže běhu JVM jako aplikace, která jej používá
- jako webová aplikace - .war archív, který se instaluje (deploy) na servletový kontejner (např. Tomcat, Jetty, Bajie...)
- eXist má Jetty server přibalen v instalačním balíku, viz eXist download [???].

eXist: instalace a spuštění

Je možno instalovat na Win NT/2000, Linuxu...

Postupujeme přesně podle instrukcí v eXist Quickstart [http://exist-db.org/quickstart.html].

Doporučuji (odzkoušeno na Win 2000 Pro):

- spustit **java -jar eXist-0.9.1-install.jar** WIKIPEDIA
  [http://cs.wikipedia.org/wiki/Speci%C3%A1ln%C3%AD:Search?search=java -jar eXist-0.9.1-install.jar]
- řídit se instalačními pokyny, instalovat např. do `\devel\eXist` WIKIPEDIA
  [http://cs.wikipedia.org/wiki/Speci%C3%A1ln%C3%AD:Search?search=\devel\eXist].
- přepnout se do instalačního adresáře, otevřít Command Shell/Prompt a spustit eXist přes Jetty webový server: **bin\startup.bat** WIKIPEDIA
  [http://cs.wikipedia.org/wiki/Speci%C3%A1ln%C3%AD:Search?search=bin\startup.bat].
- eXist ohlásí, že se spustil. Případné chybové hlášky loggeru ignorovat.

eXist: použití přes webové rozhraní

Pokud se v konfiguracích nic neměnilo, je služba eXist dostupná přes URL podobné tomuto: http://kleopatra.fi.muni.cz:8080/exist/. (tj. port 8080, cesta /exist)

Nyní můžeme vytvořit kolekci, přidat soubor, dotazovat se...

eXist: vložení dokumentu do kolekce

Vytvoříme kolekci `mydocs` a do ní přidáme dokument `databases.xml` WIKIPEDIA [http://cs.wikipedia.org/wiki/Speci%C3%A1ln%C3%AD:Search?search=databases.xml] (tyto slidy):

eXist: dotazování - zadání dotazu

Zadáme XPath dotaz, specifikujeme rozsah (ve které koleci hledat), uvedeme, kolik vyhovujících dokumentů v odpovědi vrátit.

eXist: dotazování - sumarizovaný výsledek dotazu

eXist sdělí, ve kterých kolekcích které dokumenty vyhovují dotazu:

eXist: dotazování - prohlížení jednotlivých výsledků dotazu

eXist pro každý vyhovující dokument vrátí uzly, které dotazu vyhovují