

XML Query Use Cases 確認問題

XML コンソーシアム XMLDB 勉強会
株式会社エクサ 宇都木 裕信

以下の問題は「XML Query Use Cases W3C Working Draft 8 June 2006」を元に作成いたしました。

問題 1 FLWOR 表現式の確認

http://bstore1.example.com/bib.xml (XML Query Use Cases 1.1.2 Sample Data)から 1993 年以降に出版された Addison-Wesley 著の本の一覧を出版年とタイトルを含んだ形で作成しようとしています。

bib.xml

```
<bib>
  <book year="1994">
    <title>TCP/IP Illustrated</title>
    <author><last>Stevens</last><first>W.</first></author>
    <publisher>Addison-Wesley</publisher>
    <price>65.95</price>
  </book>
  <book year="1992">
    <title>Advanced Programming in the Unix environment</title>
    <author><last>Stevens</last><first>W.</first></author>
    <publisher>Addison-Wesley</publisher>
    <price>65.95</price>
  </book>
  <book year="2000">
    <title>Data on the Web</title>
    <author><last>Abiteboul</last><first>Serge</first></author>
    <author><last>Buneman</last><first>Peter</first></author>
    <author><last>Suciu</last><first>Dan</first></author>
    <publisher>Morgan Kaufmann Publishers</publisher>
    <price>39.95</price>
  </book>
  <book year="1999">
    <title>The Economics of Technology and Content for Digital TV</title>
    <editor>
      <last>Gerbarg</last><first>Darcy</first>
      <affiliation>CITI</affiliation>
    </editor>
    <publisher>Kluwer Academic Publishers</publisher>
    <price>129.95</price>
  </book>
</bib>
```

以下の「欲しい結果」になるよう「XQuery」の空欄 A に入る文字の中で適切なものを以下の選択肢ア～エの中から選択してください。

欲しい結果

```
<bib>
  <book year="1994">
    <title>TCP/IP Illustrated</title>
  </book>
</bib>
```

XQuery

```
<bib>
{
  for $b in doc("http://bstore1.example.com/bib.xml")/bib/book
  

|   |
|---|
| A |
|---|


}
</bib>
```

解答選択肢

ア)

```
return <book year="{ $b/@year }"> { $b/title } </book>
where $b/publisher = "Addison-Wesley" and $b/@year > 1993
```

イ)

```
where $b/publisher = "Addison-Wesley" and $b/@year > 1993
return <book year="{ $b/@year }"> { $b/title } </book>
```

ウ)

```
where $b/publisher = "Addison-Wesley" and $b/@year > 1993
select <book year="{ $b/@year }"> { $b/title } </book>
```

エ)

```
select <book year="{ $b/@year }"> { $b/title } </book>
where $b/publisher = "Addison-Wesley" and $b/@year > 1993
```

ヒント

XML Query Use Cases 1.1.9.1 Q1

正解

イ

解説

FLWOR 表現式は、名前の通り For、Let、Where、Order by、Return 句を順番に書いて(ただし、For と Let は順番が逆でもよい、以降の解説でも同様)表現式を完成させます。そのため、空欄 A に入るのは、For の続きですので where、return と続いている「イ」が正解になります。

問題 2 FLWOR 表現式の確認 2

http://bstore1.example.com/bib.xml (XML Query Use Cases 1.1.2 Sample Data) から、Addison-Wesley 著の本の一覧をタイトルと価格を含め、タイトルのアルファベット順に並んだ形で作成しようとしています。

以下の「欲しい結果」になるよう「XQuery」の空欄 A に入る文字の中で適切なものを以下の選択肢ア～エの中から選択してください。

欲しい結果

```
<bib>
  <book>
    <title>Advanced Programming in the Unix environment</title>
    <price>65.95</price>
  </book>
  <book>
    <title>TCP/IP Illustrated</title>
    <price>65.95</price>
  </book>
</bib>
```

XQuery

```
<bib>
  {
    for $b in doc("http://bstore1.example.com/bib.xml")//book
      
    return <book> { $b/title } { $b/price } </book>
  }
</bib>
```

解答選択肢

ア)

```
where $b/publisher = "Addison-Wesley"
order $b/title
```

イ)

```
order $b/title
where $b/publisher = "Addison-Wesley"
```

ウ)

```
order by $b/title
where $b/publisher = "Addison-Wesley"
```

エ)

```
where $b/publisher = "Addison-Wesley"
order by $b/title
```

ヒント

XML Query Use Cases 1.1.9.7 Q7

正解

エ

解説

FLWOR 表現式は、名前の通り For、Let、Where、Order by、Return 句を順番に書いて表現式を完成させます。そのため、空欄 A に入るのは、For と Return の間には where、order by と順に入るため「エ」が正解になります。

問題 3 FLWOR 表現式の確認 3

http://bstore1.example.com/bib.xml (XML Query Use Cases 1.1.2 Sample Data)から、result要素でグループ化されたタイトルと編集者の一覧表を作成しようとしています。

以下の「欲しい結果」になるよう「XQuery」の空欄 A に入る文字の中で適切なものを以下の選択肢ア～エの中から選択してください。

欲しい結果

```
<results>
  <result>
    <title>TCP/IP Illustrated</title>
  </result>
  <result>
    <title>Advanced Programming in the Unix environment</title>
  </result>
  <result>
    <title>Data on the Web</title>
  </result>
  <result>
    <title>The Economics of Technology and Content for Digital TV</title>
    <editor>
      <last>Gerbarg</last><first>Darcy</first>
      <affiliation>CITI</affiliation>
    </editor>
  </result>
</results>
```

XQuery

```
<results>
{
  for $b in doc("http://bstore1.example.com/bib.xml")/bib/book
  return
  

|          |
|----------|
| <b>A</b> |
|----------|


}
</results>
```

解答選択肢

- ア) <result> { \$b/title } { \$b/editor } </result>
- イ) <result> { \$b/book/title } { \$b/book/editor } </result>
- ウ) <result> \$b/title \$b/editor </result>
- エ) <result> \$b/book/title \$b/book/editor </result>

ヒント

XML Query Use Cases 1.1.9.3 Q3

正解

ア

解説

return 句の中で、変数の値を表示するには{ }で囲む必要があります。

問題 4 くり返し (Iteration)

http://bstore1.example.com/bib.xml (XML Query Use Cases 1.1.2 Sample Data)から、全ての本のタイトルと著書の組み合わせの一覧表を作成しようとしています。

以下の「欲しい結果」になるように「XQuery」の空欄 A に入る文字の中で適切なものを以下の選択肢ア～エの中から選択してください。

欲しい結果

```
<results>
  <result>
    <title>TCP/IP Illustrated</title>
    <author>
      <last>Stevens</last>
      <first>W.</first>
    </author>
  </result>
  <result>
    <title>Advanced Programming in the Unix environment</title>
    <author>
      <last>Stevens</last>
      <first>W.</first>
    </author>
  </result>
  <result>
    <title>Data on the Web</title>
    <author>
      <last>Abiteboul</last>
      <first>Serge</first>
    </author>
  </result>
  <result>
    <title>Data on the Web</title>
    <author>
      <last>Buneman</last>
      <first>Peter</first>
    </author>
  </result>
  <result>
    <title>Data on the Web</title>
    <author>
      <last>Suciu</last>
      <first>Dan</first>
    </author>
  </result>
</results>
```


XQuery

```
<results>
{
  for A
  return
    <result>
      { $t }
      { $a }
    </result>
}
</results>
```

解答選択肢

ア)

```
$b in doc("http://bstore1.example.com/bib.xml")/bib/book,
  $t in $b/title,
  $a in $b/author
```

イ)

```
$t in doc("http://bstore1.example.com/bib.xml")/bib/book/title,
$a in doc("http://bstore1.example.com/bib.xml")/bib/book/author
```

ウ)

```
$b in doc("http://bstore1.example.com/bib.xml")/bib,
  $t in $b/book/title,
  $a in $b/book/author
```

エ)

```
$b in doc("http://bstore1.example.com/bib.xml"),
  $t in $b/bib/book/title,
  $a in $b/bib/book/author
```

ヒント

XML Query Use Cases 1.1.9.1 Q2 と同じ XQuery です。

正解

ア

解説

for 句での変数へのバインドに「ア」の 2 行目ような形で既存の変数の指定を行うと、互いの変数は直積にはならず、くり返しのネスト処理になります。

「ウ」「エ」は、それぞれくり返しを行う要素の指定が正しくなく、意図した結果にはなりません。

問題 5 結合 (JOIN)

http://bstore1.example.com/bib.xml (XML Query Use Cases 1.1.2 Sample Data) と http://bstore2.example.com/reviews.xml (XML Query Use Cases 1.1.4 Sample Data for Q5) から、bstore1.example.com と bstore2.example.com の両方にある本のタイトルとそれぞれの価格を取得しようとしています。

reviews.xml

```
<reviews>
  <entry>
    <title>Data on the Web</title>
    <price>34.95</price>
    <review>
      A very good discussion of semi-structured database
      systems and XML.
    </review>
  </entry>
  <entry>
    <title>Advanced Programming in the Unix environment</title>
    <price>65.95</price>
    <review>
      A clear and detailed discussion of UNIX programming.
    </review>
  </entry>
  <entry>
    <title>TCP/IP Illustrated</title>
    <price>65.95</price>
    <review>
      One of the best books on TCP/IP.
    </review>
  </entry>
</reviews>
```

以下の「欲しい結果」になるよう「XQuery」の空欄 A に入る文字の中で適切なものを以下の選択肢ア～エの中から選択してください。

欲しい結果

```
<bib>
  <book>
    <title>TCP/IP Illustrated</title>
    <bstore1>
      <price>65.95</price>
    </bstore1>
    <bstore2>
      <price>65.95</price>
    </bstore2>
  </book>
  <book>
    <title>Advanced Programming in the Unix environment</title>
    <bstore1>
      <price>65.95</price>
    </bstore1>
    <bstore2>
      <price>65.95</price>
    </bstore2>
  </book>
  <book>
    <title>Data on the Web</title>
    <bstore1>
      <price>39.95</price>
    </bstore1>
    <bstore2>
      <price>34.95</price>
    </bstore2>
  </book>
</bib>
```

XQuery

```
<bib>
{
  

|   |
|---|
| A |
|---|


  return
    <book>
      { $b1/title }
      <bstore1>
        { $b1/price }
      </bstore1>
      <bstore2>
        { $b2/price }
      </bstore2>
    </book>
}
</bib>
```

解答選択肢

ア)

```
let $b1 := doc("http://bstore1.example.com/bib.xml")//book
let $b2 := doc("http://bstore2.example.com/reviews.xml")//entry
where $b1/title = $b2/title
```

イ)

```
for $b1 in doc("http://bstore1.example.com/bib.xml")//book,
$b2 in doc("http://bstore2.example.com/reviews.xml")//entry
where $b1/title = $b2/title
```

ウ)

```
for $b1 in doc("http://bstore1.example.com/bib.xml")//book,
$b2 in doc("http://bstore2.example.com/reviews.xml")//entry
where $b1 = $b2
```

エ)

```
let $b1 := doc("http://bstore1.example.com/bib.xml")//book
let $b2 := doc("http://bstore2.example.com/reviews.xml")//entry
where $b1 = $b2
```

ヒント

XML Query Use Cases 1.1.9.5 Q5

正解

イ

解説

結合したい XML 文書をそれぞれ For 句で別々の変数にバインドし、Where 句で結合条件である任意の要素（今回は title）を指定することにより JOIN を行うことができます。

問題 6 要素数の扱い

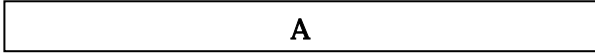
<http://bstore1.example.com/bib.xml> (XML Query Use Cases 1.1.2 Sample Data)から、author要素を持つ本のタイトルと、最初から 2 人目までの著者のリスト (3 人以上著者がいる場合は、<et-al/>要素を出力する) を作成しようとしています。

以下の「欲しい結果」になるよう「XQuery」の空欄 A に入る文字の中で適切なものを以下の選択肢ア～エの中から選択してください。

欲しい結果

```
<bib>
  <book>
    <title>TCP/IP Illustrated</title>
    <author>
      <last>Stevens</last>
      <first>W.</first>
    </author>
  </book>
  <book>
    <title>Advanced Programming in the Unix environment</title>
    <author>
      <last>Stevens</last>
      <first>W.</first>
    </author>
  </book>
  <book>
    <title>Data on the Web</title>
    <author>
      <last>Abiteboul</last>
      <first>Serge</first>
    </author>
    <author>
      <last>Buneman</last>
      <first>Peter</first>
    </author>
    <et-al/>
  </book>
</bib>
```

XQuery

```
<bib>
{
  for $b in doc("http://bstore1.example.com/bib.xml")//book
  where count($b/author) > 0
  return
    <book>
      { $b/title }
      {
        
      }
      return $a
    }
  {
    if (count($b/author) > 2)
    then <et-al/>
    else ()
  }
}
</bib>
```

解答選択肢

ア)

```
for $a in $b/author[position()<=2]
```

イ)

```
for $a in $b/author
where count($a) < 2
```

ウ)

```
for $a in $b/author
where count($b/author) < 2
```

エ)

```
for $a in $b/author[<=2]
```

ヒント

XML Query Use Cases 1.1.9.6 Q6 と同じ

正解

ア

解説

Count 関数を使用することにより、引数に指定された要素の数が分かります。また、posiotion 関数を使うことにより、任意の個数目の要素を指定することができます。

position 関数は、=の場合省略が可能です (`[position() = 1]`は、`[1]`と省略することができます) が、不等号が含まれる場合省略は出来ません。

問題7 文字列の検索

http://bstore1.example.com/bib.xml (XML Query Use Cases 1.1.2 Sample Data)から、著書名に“Bune”を含む本を検索し、タイトルと著書名を出力しようとしています。

以下の「欲しい結果」になるように「XQuery」の空欄 A に入る文字の中で適切なものを以下の選択肢ア～エの中から選択してください。

欲しい結果

```
<book>
  <title>Data on the Web</title>
  <author><last>Abiteboul</last><first>Serge</first></author>
  <author><last>Buneman</last><first>Peter</first></author>
  <author><last>Suciu</last><first>Dan</first></author>
</book>
```

XQuery

```
for $b in doc("http://bstore1.example.com/bib.xml")//book
```

```
let $e := $b/author/* 

|   |
|---|
| A |
|---|


```

```
where exists($e)
```

```
return <book> { $b/title } { $b/author } </book>
```

解答選択肢

ア)

```
[contains(string(.), "Bune ") ]
```

イ)

```
(contains(string(.), "Bune ") )
```

ウ)

```
[cont(string(.), "Bune ") ]
```

エ)

```
(cont ("Bune ", string(.)))
```

ヒント

XML Query Use Cases 1.1.9.8 Q8

正解

ア

解説

文字列検索には CONTAINS 関数 (CONTAINS(“ 検索対象文字列 ”, “ 検索文字列 ”)) を用います。
今回は CONTAINS 関数を XPATH の述部表に使用するので “[” “] ” で囲み使用します。

問題 8 文字列の検索 2

books.xml (XML Query Use Cases 1.1.6 Sample Data for Q9)から、タイトルに“Syntax”を含む本を検索し、そのタイトルを出力しようとしています。

以下の「欲しい結果」になるように「XQuery」の空欄 A に入る文字の中で適切なものを以下の選択肢ア～エの中から選択してください。

欲しい結果

```
<results>
  <title>Syntax For Data Model</title>
  <title>Basic Syntax</title>
</results>
```

XQuery

```
<results>
{
  for $t in doc("books.xml")//(chapter | section)/title
  
  return $t
}
</results>
```

解答選択肢

- ア)
let \$t := contains(\$t/text() in "Syntax")
- イ)
where contains(\$t/text() in "Syntax")
- ウ)
let \$t := contains(\$t/text(), "Syntax")
- エ)
where contains(\$t/text(), "Syntax")

ヒント

XML Query Use Cases 1.1.9.8 Q9

正解

エ

解説

文字列検索には CONTAINS 関数 (CONTAINS(“ 検索対象文字列 ”, “ 検索文字列 ”)) を用います。where 句で使用しすることによって文字列 “ Syntax ” を含む title 要素のみを出力するよう、結果の絞り込みが行えます。

問題9 ノード比較演算子

http://bstore1.example.com/bib.xml (XML Query Use Cases 1.1.2 Sample Data)から、タイトル「Data on the Web」より前に記述されている本のタイトルを出力しようとしています。

以下の「欲しい結果」になるよう「XQuery」の空欄 A に入る文字の中で適切なものを以下の選択肢ア～エの中から選択してください。

欲しい結果

```
<results>
  <title> TCP/IP Illustrated</title>
  <title>Advanced Programming in the Unix environment</title>
</results>
```

XQuery

```
<results>
  {
    let $c := for $b1 in doc("http://bstore1.example.com/bib.xml")//book
              where contains( $b1/title/text(), "Data on the Web")
              return $b1
    for $b2 in doc("http://bstore1.example.com/bib.xml")//book
      
    return $b2/title
  }
</results>
```

このクエリーは「Data on the Web」を含むタイトルの本が、XML 文書内に1つ存在する場合のみ有効です。

解答選択肢

ア)

```
where $b2 << $c
```

イ)

```
where $b2 >> $c
```

ウ)

```
where before($b2, $c)
```

エ)

```
where after($b2, $c)
```

ヒント

XML Query Use Cases 1.1.9.12 Q12

正解

ア

解説

XQuery にはノード比較演算子として `<<`, `>>`, `is` があります。 *where* `$a << $b` と使うことで `$a` の要素が `$b` の要素の前にある場合、とすることができます。

問題 10 重複値の削除 (Distinct-Values 関数)

prices.xml (XML Query Use Cases 1.1.8 Sample Data for Q10)のデータから、そこに記載されている本のタイトルの一覧 (重複は除いたもの) を作成しようとしています。

prices.xml

```
<prices>
  <book>
    <title>Advanced Programming in the Unix environment</title>
    <source>bstore2.example.com</source>
    <price>65.95</price>
  </book>
  <book>
    <title>Advanced Programming in the Unix environment</title>
    <source>bstore1.example.com</source>
    <price>65.95</price>
  </book>
  <book>
    <title>TCP/IP Illustrated</title>
    <source>bstore2.example.com</source>
    <price>65.95</price>
  </book>
  <book>
    <title>TCP/IP Illustrated</title>
    <source>bstore1.example.com</source>
    <price>65.95</price>
  </book>
  <book>
    <title>Data on the Web</title>
    <source>bstore2.example.com</source>
    <price>34.95</price>
  </book>
  <book>
    <title>Data on the Web</title>
    <source>bstore1.example.com</source>
    <price>39.95</price>
  </book>
</prices>
```

以下の「欲しい結果」になるように「XQuery」の空欄 A に入る文字の中で適切なものを以下の選択肢ア～エの中から選択してください。

欲しい結果

```
<results>
  <title>Advanced Programming in the Unix environment</title>
  <title>TCP/IP Illustrated</title>
  <title>Data on the Web</title>
</results>
```

XQuery

```
<results>
{
  

|   |
|---|
| A |
|---|


  return <title>{$a2}</title>
}
</results>
```

解答選択肢

ア)

```
for $a2 in distinct-values(doc("prices.xml")//title)
```

イ)

```
for $a in doc("prices.xml")//title
let $a2 := distinct-values($a)
```

ウ)

```
for $a in doc("prices.xml")//title
let $a2 := deep-equal($a)
```

エ)

```
for $a2 in deep-equal(doc("prices.xml")//title)
```

ヒント

XML Query Use Cases 1.1.9.4 Q4

正解

ア

解説

distinct-values 関数を使うことによって重複した値を取り除くことが出来ます。イの記述の仕方では、distinct-values 関数にはこの title 要素しか渡すことができず、すべての title 要素の比較を行うことが出来ません。

従って正解はアになります。

ちなみに deep-equal 関数は、指定された要素以下の全ての要素と値が一致するものを確認するための関数です。

問題 11 応用問題

prices.xml (XML Query Use Cases 1.1.8 Sample Data for Q10)のデータから、書籍をタイトル別に一つにまとめ、ソースと価格はショップの子要素とし、タイトルの後ろに列挙(順番はソースの昇順とする)しようとしています。

以下の「欲しい結果」になるような「XQuery」を以下の選択肢ア～エの中から選択してください。

欲しい結果

```
<prices>
  <book>
    <title>Advanced Programming in the Unix environment</title>
    <shop>
      <source>bstore1.example.com</source>
      <price>65.95</price>
    </shop>
    <shop>
      <source>bstore2.example.com</source>
      <price>65.95</price>
    </shop>
  </book>
  <book>
    <title>TCP/IP Illustrated</title>
    <shop>
      <source>bstore1.example.com</source>
      <price>65.95</price>
    </shop>
    <shop>
      <source>bstore2.example.com</source>
      <price>65.95</price>
    </shop>
  </book>
  <book>
    <title>Data on the Web</title>
    <shop>
      <source>bstore1.example.com</source>
      <price>34.95</price>
    </shop>
    <shop>
      <source>bstore2.example.com</source>
      <price>39.95</price>
    </shop>
  </book>
</prices>
```

解答選択肢

ア)

```
<prices>
{
  let $books := doc("prices.xml")/prices/book
  for $title in distinct-values($books/title)
  return
    <book>
      <title>{ $title }</title>
      {
        for $book in $books
        where
          $book/title = $title
        order by $book/source
        return
          <shop>
            { $book/source, $book/price }
          </shop>
      }
    </book>
}
</prices>
```

イ)

```
<prices>
{
  let $books := doc("prices.xml")/prices/book
  for $title in distinct-values($books/title),
    $book in $books[title = $title]
  order by $book/source
  return
    <book>
      <title>{ $title }</title>
      <shop>
        { $book/source, $book/price }
      </shop>
    </book>
}
</prices>
```

ウ)

```
<prices>
{
  let $books := for $book in doc("prices.xml")/prices/book
                order by $book/source
                return $book
  for $title in distinct-values($books/title)
  let $shops := let $shop := $books
                return
                  <shop>
                    { $books[./title = $title]/source}
                    { $books[./title = $title]/price }
                  </shop>

  return
    <book>
      <title>{ $title }</title>
      { $shops }
    </book>
}
</prices>
```

エ)

```
<prices>
{
  let $books := for $book in doc("prices.xml")/prices/book
                order by $book/source
                return $book
  for $title in distinct-values($books/title)
  let $shops := for $book in $books
                where
                  $book/title = $title
                return
                  <shop>
                    { $book/source, $book/price }
                  </shop>

  return
    <book>
      <title>{ $title }</title>
      { $shops }
    </book>
}
</prices>
```

ヒント

問題 4、問題 10、XML Query Use Cases 1.1.9.4 Q4

正解

ア、エ

解説

選択肢ア()

```
let $books := doc("prices.xml")/prices/book
for $title in distinct-values($books/title)
```

によって、重複のない書籍名のシーケンス

```
(Advanced Programming in the Unix environment,
TCP/IP Illustrated,
Data on the Web)
```

の項目が一つ一つ変数\$title にバインドされて return 以下が実行されます。

return では、タイトルが\$title に等しい書籍をソースの値でソートし、書籍毎にソースと価格を取り出し、ショップでグルーピングして、目的の結果を得ることができます。

選択肢イ(×)

```
let $books := doc("prices.xml")/prices/book
for $title in distinct-values($books/title),
    $book in $books[title = $title]
```

によって、重複のない書籍名のシーケンス

```
(Advanced Programming in the Unix environment,
TCP/IP Illustrated,
Data on the Web)
```

の項目が一つ一つ変数\$title にバインドされます、さらに、各\$title に対して書籍名が\$title に等しい書籍が変数\$book にバインドされます。結果として、次のタプルが作られます。

```
($title, $book) = (Advanced Programming in the Unix environment, $books[1])
                  (Advanced Programming in the Unix environment, $books[2])
                  (TCP/IP Illustrated, $books[3])
                  (TCP/IP Illustrated, $books[4])
                  (Data on the Web, $books[5])
                  (Data on the Web, $books[6])
```

上記タプルは、

```
order by $book/source
```

により、次の順に並べ替えられ、タプル毎に return が実行されます。

```
{ $title, $book } = { Advanced Programming in the Unix environment, $books[2] }
                   { TCP/IP Illustrated, $books[4] }
                   { Data on the Web, $books[6] }
                   { Advanced Programming in the Unix environment, $books[1] }
                   { TCP/IP Illustrated, $books[3] }
                   { Data on the Web, $books[5] }
```

実行結果は、以下のようになります。

```
<prices>
  <book>
    <title>Advanced Programming in the Unix environment</title>
    <shop>
      <source>bstore1.example.com</source>
      <price>65.95</price>
    </shop>
  </book>
  <book>
    <title>TCP/IP Illustrated</title>
    <shop>
      <source>bstore1.example.com</source>
      <price>65.95</price>
    </shop>
  </book>
  <book>
    <title>Data on the Web</title>
    <shop>
      <source>bstore1.example.com</source>
      <price>39.95</price>
    </shop>
  </book>
  <book>
    <title>Advanced Programming in the Unix environment</title>
    <shop>
      <source>bstore2.example.com</source>
      <price>65.95</price>
    </shop>
  </book>
  <book>
    <title>TCP/IP Illustrated</title>
    <shop>
      <source>bstore2.example.com</source>
      <price>65.95</price>
    </shop>
  </book>
  <book>
    <title>Data on the Web</title>
    <shop>
      <source>bstore2.example.com</source>
      <price>34.95</price>
    </shop>
  </book>
</prices>
```

選択肢ウ(×)

```
let $books := for $book in doc("prices.xml")/prices/book
              order by $book/source
              return $book
for $title in distinct-values($books/title)
```

によって、重複のない書籍名のシーケンス

```
(Advanced Programming in the Unix environment,
TCP/IP Illustrated,
Data on the Web)
```

の項目が一つ一つ変数\$title にバインドされます。さらに、各\$title に対して、

```
let $shops := let $shop := $books
              return
                <shop>
                  { $books[./title = $title]/source }
                  { $books[./title = $title]/price }
                </shop>
```

により、書籍名が\$title に等しい2冊の書籍のソースと価格をショップでグルーピングしたシーケンスが、変数\$shops にバインドされます。結果として、次のタプルが作られます。

```
{ $title, $shops } =
{Advanced Programming in the Unix environment,
 <shop>
  <source>bstore2.example.com</source><source>bstore1.example.com</source>
  <price>65.95</price><price>65.95</price>
</shop> }
{TCP/IP Illustrated,
 <shop>
  <source>bstore2.example.com</source><source>bstore1.example.com</source>
  <price>65.95</price><price>65.95</price>
</shop> }
{Data on the Web,
 <shop>
  <source>bstore2.example.com</source><source>bstore1.example.com</source>
  <price>34.95</price><price>39.95</price>
</shop> }
```

上記タプル毎に return が実行され、結果は以下ようになります。


```

<prices>
  <book>
    <title>Advanced Programming in the Unix environment</title>
    <shop>
      <source>bstore2.example.com</source>
      <source>bstore1.example.com</source>
      <price>65.95</price>
      <price>65.95</price>
    </shop>
  </book>
  <book>
    <title>TCP/IP Illustrated</title>
    <shop>
      <source>bstore2.example.com</source>
      <source>bstore1.example.com</source>
      <price>65.95</price>
      <price>65.95</price>
    </shop>
  </book>
  <book>
    <title>Data on the Web</title>
    <shop>
      <source>bstore2.example.com</source>
      <source>bstore1.example.com</source>
      <price>39.95</price>
      <price>34.95</price>
    </shop>
  </book>
</prices>

```

選択肢工()

```

let $books := for $book in doc("prices.xml")/prices/book
              order by $book/source
              return $book
for $title in distinct-values($books/title)

```

によって、重複のない書籍名のシーケンス

```

(Advanced Programming in the Unix environment,
TCP/IP Illustrated,
Data on the Web)

```

の項目が一つ一つ変数\$title にバインドされます。さらに、各\$title に対して、タイトルが\$title に等しい書籍からソースと価格を取り出しショップでグルーピングしています。

```

let $shops := for $book in $books
              where
                $book/title = $title
              return
                <shop>
                  { $book/source, $book/price }
                </shop>

```

\$books は既にソースでソートされており、結果として、以下のタプルが作られます。

```
{ $title, $shops } =  
  { Advanced Programming in the Unix environment,  
    (<shop><source>bstore1.example.com</source><price>65.95</price></shop>,  
      <shop><source>bstore2.example.com</source><price>65.95</price></shop>) }  
  { TCP/IP Illustrated,  
    (<shop><source>bstore1.example.com</source><price>65.95</price></shop>,  
      <shop><source>bstore2.example.com</source><price>65.95</price></shop>) }  
  { Data on the Web,  
    (<shop><source>bstore1.example.com</source><price>39.95</price></shop>,  
      <shop><source>bstore2.example.com</source><price>34.95</price></shop>) }
```

上記タプル毎に return が実行され、目的の結果を得ることができます。