XML コンソーシアム XMLDB 勉強会
株式会社サイバーテック 福田 寛明

1.1.2 Sample Data (bib.xml)

bib.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE bib SYSTEM "schema/bib.dtd">
<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>
```

1.1.9.1 Q1

a. Question

bib.xml から「Addison-Wesley 社より 1991 年より後(1991 年を含まない)に出版された書籍」の一覧表を作成しなさい。
一覧表は発行年(year)と書籍名(title)を含むこと。

b. XQuery 記述例

```
1:      <bib>
2:          {
3:              for $b in doc("bib.xml")/bib/book
4:              where $b/publisher = "Addison-Wesley" and $b/@year > 1991
5:              return
6:                  <book year="{ $b/@year }">
7:                      { $b/title }
8:                  </book>
9:          }
10:     </bib>
```

c. XQuery 概説

```
3:      ドキュメント bib.xml について、
        /bib/book ノードを順に$b に格納しつつ、以下の処理を繰り返す。
4:          $b/publisher ノードが“Addison-Wesley”、かつ$b/@year ($b の year 属性)が 1991 より大きければ、
5:          結果を下記形式に整形して出力する。
6:              <book year="{ $b/@year }">
7:                  { $b/title }
8:              </book>
```

d. 期待される結果

```
<bib>
    <book year="1994">
        <title>TCP/IP Illustrated</title>
    </book>
    <book year="1992">
        <title>Advanced Programming in the Unix environment</title>
    </book>
</bib>
```

e. XQuery 詳説

･考え方

bib.xml から、子要素 publisher の値が"Addison-Wesley"で、かつ year 属性の値が 1991 より大な book 要素のみを残す。次に、残った各 book 要素に対して、属性 year と子要素 title のみを残す。

まず、bib.xml から、各 book 要素の属性 year、および子要素 title のみ残した結果を求める。

```
<bib>
{
for $b in doc("bib.xml")/bib/book
    return
        <book year="{ $b/@year }">
            { $b/title }
        </book>
}
</bib>
```

厳密に言うと、bib.xml 内の bib および book 要素と、上記 XQuery 式で返される bib および book 要素とは同一ではない。

book 要素の属性 year および子要素 title のみ残すとは言っても、XQuery ではその他の内容(author, editor, publisher, price)を「切り離す」ことはできない。

そこで、bib および book 要素は XQuery 式の中で生成し、「残したい」year 属性および title 要素を元の book 要素から抜き出して、生成した book 要素に追加する形をとる。

次に、子要素 publisher の値が"Addison-Wesley"で、かつ year 属性の値が 1991 より大な book 要素に限定するには、for 節の後に以下の where 節を加えればよい。

```
where $b/publisher = "Addison-Wesley" and $b/@year > 1991
```

最終的に題意を満たす XQuery 式は下記のようになる。

```
<bib>
{
    for $b in doc("bib.xml")/bib/book
    where $b/publisher = "Addison-Wesley" and $b/@year > 1991
    return
        <book year="{ $b/@year }">
            { $b/title }
        </book>
}
</bib>
```

・クエリ実行経過

3 行目

for $b in doc("bib.xml")/bib/book

得られるシーケンス:

($b) = {
(/bib/book[1]),
(/bib/book[2]),
(/bib/book[3]),
(/bib/book[4])
}

3〜4 行目

for $b in doc("bib.xml")/bib/book
where $b/publisher = "Addison-Wesley" and $b/@year > 1991

得られるシーケンス:

($b) = {
(/bib/book[1]),
(/bib/book[2])
}

1.1.9.2 Q2

a. Question

bib.xml から書籍名(title), 著者(author)の全ての組み合わせを抽出し、一覧表にしなさい。
組み合わせは各々result 要素で囲むこと。

b. XQuery 記述例

```
1:    <results>
2:        {
3:            for $b in doc("bib.xml")/bib/book,
4:                $t in $b/title,
5:                $a in $b/author
6:            return
7:                <result>
8:                    { $t }
9:                    { $a }
10:               </result>
11:       }
12:   </results>
```

c. XQuery 概説

```
3:    bib.xml について、
      下記の通り$b, $t, $a の組み合わせを作りつつ、以下の処理を繰り返す。
        $b  : /bib/book ノードを順に格納する。
4:      $t  : $b に含まれる title ノード($b/title)を順に格納しする。
5:      $a  : $b, $t の組み合わせに対し、$b/author ノードを順に格納する。
6:          結果を下記形式に整形して出力する。
7:              <result>
8:                  { $t }
9:                  { $a }
10:             </result>
```

※ 3〜5 行目で実際に生成されるノードの組み合わせ(タプル)については「e. XQuery 詳説」の「クエリ実行経過」を参照。

d. 期待される結果

```
<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>
```

e. XQuery 詳説

・考え方

bib.xml の各 book 要素に対して、子要素 title と author の全組合せ(タプル)を作る。
各タプルは、result 要素の内容とし、全 result 要素は results 要素の内容とする。

```
<results>
{
    for $b in doc("bib.xml")/bib/book,
        $t in $b/title,
        $a in $b/author
    return
        <result>
            { $t }
            { $a }
        </result>
}
</results>
```

($b, $t, $a)の組合せを構成できる book, title, author の組合せのみが、return 節で使用される。
/bib/book[4]は子要素に author を持たない。すなわち、($b, $t, $a)の組合せを構成する$a が存在しない。当然、/bib/book[4]/title が return 節で使用されることはない。

・クエリ実行経過

3 行目

```
for $b in doc("bib.xml")/bib/book
```

得られるシーケンス:

```
($b) = {
        (/bib/book[1]),
        (/bib/book[2]),
        (/bib/book[3]),
        (/bib/book[4])
}
```

3〜4 行目

```
for $b in doc("bib.xml")/bib/book,
    $t in $b/title
```

得られるタプル･シーケンス:

```
($b, $t) = {
        (/bib/book[1], $b/title[1]),
        (/bib/book[2], $b/title[1]),
        (/bib/book[3], $b/title[1]),
        (/bib/book[4], $b/title[1])
} = {
        (/bib/book[1], /bib/book[1]/title[1]),
        (/bib/book[2], /bib/book[2]/title[1]),
        (/bib/book[3], /bib/book[3]/title[1]),
        (/bib/book[4], /bib/book[4]/title[1])
}
```

3〜5 行目

```
for $b in doc("bib.xml")/bib/book,
    $t in $b/title,
    $a in $b/author
```

得られるタプル･シーケンス:

```
($b, $t, $a) = {
        (/bib/book[1], $b/title[1], $b/author[1]),
        (/bib/book[2], $b/title[1], $b/author[1]),
        (/bib/book[3], $b/title[1], $b/author[1]),
        (/bib/book[3], $b/title[1], $b/author[2]),
        (/bib/book[3], $b/title[1], $b/author[3])
} = {
        (/bib/book[1], /bib/book[1]/title[1], /bib/book[1]/author[1]),
        (/bib/book[2], /bib/book[2]/title[1], /bib/book[2]/author[1]),
        (/bib/book[3], /bib/book[3]/title[1], /bib/book[3]/author[1]),
        (/bib/book[3], /bib/book[3]/title[1], /bib/book[3]/author[2]),
        (/bib/book[3], /bib/book[3]/title[1], /bib/book[3]/author[3])
}
```

タプル($b, $t) = (/bib/book[4], /bib/book[4]/title[1]) のケースについては /bib/book[4]/author が空シーケンスであるためタプル($b, $t, $a) を構成できない。従って for 句で撥ねられ return 句に適用されない。

1.1.9.3 Q3

a. Question

bib.xml から書籍名(title)と著者(author)の一覧表を作成しなさい。
書籍名と著者の纏まりを result 要素で囲むこと。

b. XQuery 記述例

```
1:  <results>
2:  {
3:      for $b in doc("bib.xml")/bib/book,
4:      return
5:          <result>
6:              { $b/title }
7:              { $b/author }
8:          </result>
9:  }
10: </results>
```

c. XQuery 概説

```
3:  ドキュメント bib.xml について、
    /bib/book ノードを順に$b に格納しつつ、以下の処理を繰り返す。
4:      結果を下記形式に整形して出力する。
5:          <result>
6:              { $b/title }
7:              { $b/author }
8:          </result>
```

d. 期待される結果

```
<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>
        <author>
            <last>Buneman</last>
            <first>Peter</first>
        </author>
        <author>
            <last>Suciu</last>
            <first>Dan</first>
        </author>
    </result>
    <result>
        <title>The Economics of Technology and Content for Digital TV</title>
    </result>
</results>
```

e. XQuery 詳説

・考え方

bib.xml の各 book 要素に対して、子要素 title と author の全てを result 要素の内容とする。
さらに、全 result 要素は results 要素の内容とする。

```
<results>
{
    for $b in doc("bib.xml")/bib/book
    return
        <result>
            { $b/title }
            { $b/author }
        </result>
}
</results>
```

Q2 と異なり、author 要素を子要素として持たない/bib/book[4]も return 節で使用される。
/bib/book[4]/author は空シーケンス( )である。また、/bib/book[3]/author は 3 つの author 要素から成るシーケンスである。

・クエリ実行経過

3 行目

```
for $b in doc("bib.xml")/bib/book
```

得られるシーケンス:

```
($b) = {
    (/bib/book[1]),
    (/bib/book[2]),
    (/bib/book[3]),
    (/bib/book[4])
}
```

1.1.9.4 Q4

a. Question

bib.xml の著者(author)毎に、全ての著書(title)を列挙した一覧表を作成しなさい。
著者と著書の纏まりを result 要素で囲むこと。

b. XQuery 記述例

```
1:  <results>
2:      {
3:          let $a := doc("bib.xml")//author
4:          for $last in distinct-values($a/last),
5:              $first in distinct-values($a[last=$last]/first)
6:          order by $last, $first
7:          return
8:              <result>
9:                  <author>
10:                     <last>{ $last }</last>
11:                     <first>{ $first }</first>
12:                 </author>
13:                 {
14:                     for $b in doc("bib.xml")/bib/book
15:                     where some $ba in $b/author
16:                         satisfies {$ba/last = $last and $ba/first = $first}
17:                     return $b/title
18:                 }
19:             </result>
20:     }
21: </results>
```

c. XML Query 概説

```
3:      bib.xml に含まれる著者(author)ノードを全て抽出し、そのシーケンスを$a に格納する。
4~5:    $a から著者の重複を取り除き、出力用に著者名のタプル ( $last, $first ) のシーケンスを構成する。
6:      シーケンスは著者名の昇順にソートしておく。
7:      下記形式で結果を出力する。
8:          <result>
9:              <author>
10:                 <last>{ $last }</last>       (: 著者のラストネーム :)
11:                 <first>{ $first }</first>    (: 著者のファーストネーム :)
12:             </author>
13:             {
14:                 bib.xml の書籍(/bib/book)ノードのシーケンスを$b に格納し、各々に対して下記の処理を行う。
15~16:                 書籍の著者名($ba/last, $ba/first)に($last, $first)と一致するものが1つ以上存在する場合、
17:                 その書籍の書籍名($b/title)ノードを出力する。
18:             }
19:         </result>
```

d. 期待される結果

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

e. XQuery 詳説

・考え方

bib.xml の author 要素毎にその著者の title 要素全てを含め、result 要素の内容とする。さらに、全 result 要素は results 要素の内容とすることで結果を求める。

author 要素は同一の内容をもつものがあるため、比較して重複を除く必要がある。
author 要素には子要素 last と first があるため、author 要素同士の比較には、last、first 要素についての比較が必要である。

```
<results>
{
    let $a := doc("bib.xml")//author

    for $last in distinct-values($a/last),                                    (: last の値を重複を除いて取り出す :)
        $first in distinct-values($a[last=$last]/first)(: last と同一の親要素(author)をもつ first の値を重複を除いて取り出す :)
    order by $last, $first
    return
        <result>
            <author>
                <last>{ $last }</last>
                <first>{ $first }</first>
            </author>

            (:
                for 節の各($last, $first)に対して、子要素 last の値が$last、first の値が$first である book を求め、
                その title 要素を返す。
            :)
            {
                for $b in doc("bib.xml")/bib/book

                (:
                    $b/author シーケンスの各ノードを順に代入した$ba について
                    条件 "$ba/last = $last and $ba/first = $first" で判定し、これを満たすケースが
                    1 つでも存在する場合に真となる。
                :)
                where some $ba in $b/author
                    satisfies ($ba/last = $last and $ba/first=$first)
                return $b/title
            }
        </result>
}
</results>
```

・クエリ実行経過

3 行目

let $a := doc("bib.xml")//author

得られるシーケンス:

($a) = {
(/bib/book[1]/author[1], /bib/book[2]/author[1], /bib/book[3]/author[1], /bib/book[3]/author[2], /bib/book[3]/author[3])
}

3〜4 行目

let $a := doc("bib.xml")//author
for $last in distinct-values($a/last)

得られるシーケンス:

($last) = {
Stevens,
Abiteboul,
Buneman,
Suciu
}

(/bib/book[2]/author[1]/last[1]) と (/bib/book[1]/author[1]/last[1]) の値(テキスト)は同じ"Stevens"のため、distinct-values 関数により重複が除かれます。

3〜5 行目

let $a := doc("bib.xml")//author
for $last in distinct-values($a/last),
$first in distinct-values($a[last=$last]/first)

得られるシーケンス:

($last, $first) = {
(Stevens, W.),
(Abiteboul, Serge),
(Buneman, Peter),
(Suciu, Dan)
}

3〜6 行目

```
let $a := doc("bib.xml")//author
for $last in distinct-values($a/last),
    $first in distinct-values($a[last = $last]/first)
order by $last, $first
```

シーケンス①: bib.xml に含まれる全ての著者について、重複を除いて名前の昇順でソートしたシーケンス

```
($last, $first) = {
    (Abiteboul, Serge),     "Abiteboul", "Serge"  タプル①-1
    (Buneman, Peter),       "Buneman", "Peter"  タプル①-2
    (Stevens, W.),          "Stevens", "W."  タプル①-3
    (Suciu, Dan)            "Suciu", "Dan"  タプル①-4
}
```

14〜16 行目

```
for $b in doc("bib.xml")/bib/book
where some $ba in $b/author
      satisfies ($ba/last = $last and $ba/first = $first)
return $b/title
```

シーケンス②:

```
($b, $ba/last, $ba/first) = {
    (/bib/book[1], Stevens, W.),                "Stevens", "W."  タプル②-1
    (/bib/book[2], Stevens, W.),                "Stevens", "W."  タプル②-2
    (/bib/book[3], Abiteboul, Serge),           "Abiteboul", "Serge"  タプル②-3
    (/bib/book[3], Buneman, Peter),             "Buneman", "Peter"  タプル②-4
    (/bib/book[3], Suciu, Dan)                  "Suciu", "Dan"  タプル②-5
}
```

「著者シーケンス①の著者名($last, $first)」と「シーケンス②の著者名($ba/last, $ba/first)」を、satisfies 句で突き合わせる。
下記のケースが残るので、マッチした時点のタプル②の書籍名($b/title)を出力する。

タプル①-1 → タプル②-3 **"Data on the Web"**
タプル①-2 → タプル②-4 **"Data on the Web"**
タプル①-3 → タプル②-1, **"TCP/IP Illustrated"**
タプル②-2 **"Advanced Programming in the Unix Environment"**
タプル①-4 → タプル②-5 **"Data on the Web"**

f. 別解

```
<results>
{
    let $a := doc("bib.xml")//author
    for $last in distinct-values($a/last),
        $first in distinct-values($a/first[../last=$last])
    order by $last, $first
    return
        <result>
            <author>
                <last>{ $last }</last>
                <first>{ $first }</first>
            </author>

            {
                for $b in doc("bib.xml")/bib/book[auther]
                where
                    $b/auther/last = $last and
                    $b/auther/first = $first
                return $b/title
            }
        </result>
}
</results>
```