



XML Query Use Casesを使用した XQuery入門

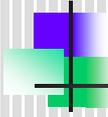
2007年5月18日

XMLDB勉強会 技術系サブグループ XMLDB実践チーム
(株)日立システムアンドサービス 藤春康弘



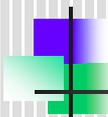
XMLDB実践チーム活動取り纏め者(敬称略)

- XQueryにこれから取り組む人のためのFAQ作成
伊藤(凸版印刷)
- XML Query Use Casesの解説作成
福田(サイバーテック)
千種(日立製作所)
本山(NTTデータ)
山崎(サイバーテック)
中村(ジャステック)
- XML Query Use Cases理解のための確認問題作成
宇都木(エクサ)
藤春(日立システム)



アジェンダ

1. XML実践チーム発足経緯
2. 発表テーマ選定経緯(1)
3. 発表テーマ選定経緯(2)
4. 成果物
5. 成果物ご紹介(1) -- FAQ
6. 成果物ご紹介(2) -- Use Cases解説
7. 成果物ご紹介(3) -- 確認問題
8. 活動のまとめ



1. XML実践チーム発足経緯

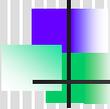
第1回XMLDB勉強会

2006年10月4日開催。

マーケティング系と技術系の2グループで活動することに決定。

第2回XMLDB技術系勉強会

技術系グループは、XMLDBを使ってみようというチームと、XMLDBに関するドキュメント整備をするチームの2本立てで活動することに決定。



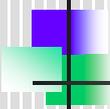
2. 発表テーマ選定経緯(1)

第2回XMLDB技術系勉強会

3つのXMLDB(ネイティブ2, ハイブリッド1)を動かしてみることにする。

第3回XMLDB技術系勉強会

結果を持ち寄って、ディスカッション。
大きな問題はなく動かせた。
XMLDBの機能、適用業務に関する意見交換。



3. 発表テーマ選定経緯(2)

第4回XMLDB技術系勉強会

XMLDBのパフォーマンス、設計/運用、および技術的可能性にまで及んで議論。

しかし、XQueryが書けないことには始まらない。

XQueryの「詳細で本格的」なテキストがない。
これ以降、W3C公開の「XML Query Use Cases」
を題材に、XQueryに取り組んで行くことに決定。

4. 成果物

これからXQueryに取り組もうとされる方々への手引きとなるものを残す。

FAQ作成

XQueryとは、学習環境、情報源、基本事項紹介。

Use Cases解説

最初の12例について、解説を入れる。

確認問題

Use Casesの理解確認のための問題。

5. 成果物ご紹介(1) -- FAQ

- Q1. XQueryを使うと何が出来ますか？
- Q2. XQueryの構文はどのようなものですか？
- Q3. XQueryを学ぶにはどのような方法がありますか？
- Q4. XQueryにて処理を行うのと、XPathで処理を行うのでは何が違いますか？
- Q5. XMLDB製品毎にXQueryの実装に差(方言)はありますか？
- Q6. FLWOR式について詳しく教えて下さい。
- Q7. XMLDBにおいてデータ抽出(検索)はXQueryが標準になりつつあるようですが、データ投入や更新に対する標準の言語はありますか？
- Q8. XQueryの他に、XMLDBに関連したW3C規格にはどのようなものがありますか？

6. 成果物ご紹介(2) -- Use Cases解説

XML Query Use Casesの冒頭に以下の記載。

この文書のクエリに付随する説明はとても単純で、クエリの機能の説明はそれぞれのクエリの結果例自体を注意深く見て把握する必要があります。
(ドキュメンテーションチームの翻訳「訳注」より)

以下、W3C公開の「XML Query Use Cases」に記載されている最初の4例を題材にして紹介いたします。

6-1 Q1 (1)

```
<bib>
{
  for $b in doc("http://bstore1.example.com/bib.xml")/bib/book
  where $b/publisher = "Addison-Wesley" and $b/@year > 1991
  return
    <book year="{ $b/@year }">
      { $b/title }
    </book>
}
</bib>
```

6-1 Q1 (2)



[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>
  .....
</bib>
```

[XQuery実行結果]

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

6-1 Q1 (3)



bib.xmlと実行結果を比較すると、「各book要素が、year属性とtitle要素のみ残して取り出されている」というイメージ。

しかし、XQueryでは、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要素に追加するという形をとる。

6-2 Q2 (1)



タイトルと著者、双方を持つ書籍に対して、そのタイトルと著者のペアを返す。

```
<results>
{
  for $b in doc("http://bstore1.example.com/bib.xml")/bib/book,
    $t in $b/title,
    $a in $b/author
  return
    <result>
      { $t }
      { $a }
    </result>
}
</results>
```

6-2 Q2 (2)



for節では、bib.xmlの各book要素に対して、子要素titleとauthorとの全組合せ(タプル)を作る。

```
{ book[1], book[1]/title, book[1]/author }
{ book[2], book[2]/title, book[2]/author }
{ book[3], book[3]/title, book[3]/author[1] }
{ book[3], book[3]/title, book[3]/author[2] }
{ book[3], book[3]/title, book[3]/author[3] }
```

タプル内の各値が変数(\$b, \$t, \$a)にバインドされ、順次return節が実行される。

もし、order byの指定があれば(Q2にはない)、タプル毎にソートされる。

6-2 Q2 (3)



(\$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。

book[4]は子要素にauthorを持たない(代わりにeditorを持つ)。すなわち、(\$b, \$t, \$a)の組合せを構成するauthorが存在しない。

当然、book[4]/titleがreturn節で使用されることはない。

6-3 Q3 (1)



各書籍に対して、その書籍が持つタイトルと全ての著者を返す。

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

6-3 Q3 (2)



Q2と異なり、author要素を子要素として持たないbook[4]もreturn節で使用される。

book[4]にバインドされる変数\$bに対して、\$b/authorは空シーケンス()。
したがって、{ \$b/author }は何も返さない。

book[3]にバインドされる変数\$bに対して、\$b/authorは3つのauthor要素から成るシーケンス。
したがって、{ \$b/author }は3つのauthor要素を返す。

6-4 Q4 (1)



著者毎に、その著作のタイトルをグルーピングして返す。

```
<results>
{
  let $a := doc("http://bstore1.example.com/bib/bib.xml")//author
  for $last in distinct-values($a/last),
    $first in distinct-values($a[last=$last]/first)
  order by $last, $first
  return
  <result>
  <author>
  <last>{ $last }</last>
  <first>{ $first }</first>
  </author>
  {
    for $b in doc("http://bstore1.example.com/bib.xml")/bib/book
    where some $ba in $b/author
      satisfies ($ba/last = $last and $ba/first = $first)
    return $b/title
  }
  </result>
}
</results>
```

6-4 Q4 (2)



要は、author要素毎にその著者のtitle要素全てを求める。

ただし、author要素は、同一の内容をもつものがある。
比較して重複を除く必要がある。

さらに、author要素は、子要素lastとfirstをもつ。
author要素同士の比較には、last、first要素についての比較が必要。

以下、Q4のXQuery式を上から順に追っていきます。

6-4 Q4 (3)



```
let $a := doc("http://bstore1.example.com/bib/bib.xml")//author
```

変数\$aには、次のシーケンスがバインド。

```
$a = (  
  <author><last>Stevens</last><first>W.</first></author>,  
  <author><last>Stevens</last><first>W.</first></author>,  
  <author><last>Abiteboul</last><first>Serge</first></author>,  
  <author><last>Buneman</last><first>Peter</first></author>,  
  <author><last>Suciu</last><first>Dan</first></author>  
)
```

6-4 Q4 (4)



```
for $last in distinct-values($a/last)
```

```
$a/last = (<last>Stevens</last>, <last>Stevens</last>,  
          <last>Abiteboul</last>, <last>Buneman</last>,  
          <last>Suciu</last>)
```

distinct-values関数により重複を除いたlast要素の値からなるシーケンスは次のようになる。

```
(Stevens, Abiteboul, Buneman, Suciu)
```

変数\$lastには、上記シーケンスの先頭から1つずつバインドされる。

6-4 Q4 (5)



```
for $last in distinct-values($a/last),  
   $first in distinct-values($a[last=$last]/first)
```

\$a[last=\$last]/firstは、\$last毎に、last要素の値が\$lastに等しいauthor要素を求め、その子要素であるfirst要素からなるシーケンスを示す。

したがって、変数\$lastと\$firstにバインドされるタプルは次のようになる。

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

6-4 Q4 (6)



```
for $last in distinct-values($a/last),  
  $first in distinct-values($a[last=$last]/first)  
order by $last, $first
```

求めたタブルを、\$last, \$firstの順でソート。

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

6-4 Q4 (7)



先に求めた、タブル{ \$last, \$first }毎に下記が実行。

```
return  
  <result>  
    <author>  
      <last>{ $last }</last>  
      <first>{ $first }</first>  
    </author>  
    {  
      .....  
    }  
  </result>
```

最後に、次のスライドで、{ }の内側を説明いたします。

6-4 Q4 (8)



先に求めた、タプル{\$last, \$first}毎に下記が実行。

```
for $b in doc("http://bstore1.example.com/bib.xml")/bib/book
where some $ba in $b/author
    satisfies ($ba/last = $last and $ba/first = $first)
return $b/title
```

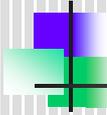
各book要素に対して、last要素の値が\$lastに等しく、first要素の値が\$firstに等しいような子要素authorが1つでも存在すれば、そのbook要素の子要素titleを返す、というもの。

これで、著者別に著書のタイトルを全て求めることができる。

7. 成果物ご紹介(3) -- 確認問題



- 問題1, 2, 3 FLWOR表現式の確認
- 問題4 くり返し(Iteration)
- 問題5 結合(JOIN)
- 問題6 要素数の扱い
- 問題7, 8 文字列の検索
- 問題9 ノード比較演算子
- 問題10 重複値の削除(Distinct - Values関数)
- 問題11 応用問題



8. 活動のまとめ

- (1) XMLDBに接してみようとしたきっかけ
- (2) XMLDBを実際に動かしてみた上での所感
- (3) XML Query Use Casesを題材に学習した上での所感
- (4) XMLDBやXQueryについて普及・啓蒙して行く上で必要と思われる事
- (5) 勉強会に参加した上での所感