XQuery

XQuery

- FLWR expressions
  - FOR and LET expressions
  - Collections and sorting

Resource

W3C recommendation: www.w3.org/TR/xquery/

Symbols

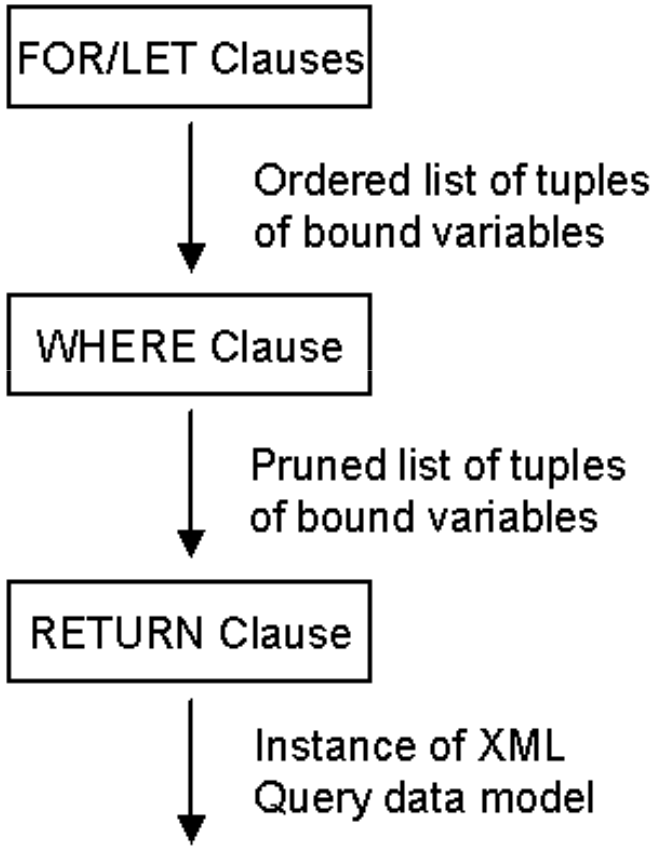
.	Denotes the current node.
..	Denotes the parent of the current node.
/	Denotes the root node, or a separator between steps in a path.
//	Denotes descendants of the current node.
@	Denotes attributes of the current node.
*	Denotes "any" (node with unrestricted name).
[ ]	Brackets enclose a Boolean expression that serves as a predicate for a given step.
[n]	When a predicate consists of an integer, it serves to select the element with the given ordinal number from a list of elements.

FLWR (“Flower”) Expressions

FOR ... LET... FOR... LET...

WHERE...

RETURN...



FOR v.s. LET

FOR

- Binds *node variables* → iteration

LET

- Binds *collection variables* → one value

WHERE - continued

- AND, OR, and NOT usually contain references to bound variables
- Variables bound in FOR clause usually contain scalar predicates

  $p/color = "Red"

- Variables bound in LET clause usually used in list predicates

  avg($p/price) > 100

XML - Example Schema

- <!ELEMENT bib (book*)>
- <!ELEMENT book (title, (author+ | editor+), publisher, price)>
- <!ATTLIST book year CDATA #REQUIRED >
  <!ELEMENT author (last, first)>
- <!ELEMENT editor (last, first, affiliation)>
  <!ELEMENT title (#PCDATA)>
- <!ELEMENT last (#PCDATA)>
- <!ELEMENT first (#PCDATA)>
- <!ELEMENT affiliation (#PCDATA)>
  <!ELEMENT publisher (#PCDATA)>
  <!ELEMENT price (#PCDATA)>

XML - Example

```
<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>
    <author><last> Gerbarg</last>
        <first>Darcy </first>
        <affiliation>CITI</affiliation>
    </author>
      <publisher> Kluwer Academic Publishers
    </publisher>
      <price> 129.95  </price>
  </book>
</bib>
```

Xquery – Example 1

Find all book titles published after 1995:

```
FOR $x IN document("bib.xml")/bib/book
WHERE $x/year > 1995
RETURN $x/title
```

Result:

<title> abc </title>
<title> def </title>
<title> ghi </title>

XQuery - Example 2

For each author of a book by Morgan Kaufmann, list all books she published:

```
FOR $a IN distinct(document("bib.xml")
                   /bib/book[publisher="Morgan Kaufmann"]/author)
RETURN <result>
            $a,
            FOR $t IN /bib/book[author=$a]/title
            RETURN $t
        </result>
```

distinct = a function that eliminates duplicates

Xquery -Example 2

Result:

```
<result>
    <author>Jones</author>
    <title> abc </title>
    <title> def </title>
</result>
<result>
    <author> Smith </author>
    <title> ghi </title>
</result>
```

XQuery

- FOR $x in expr -- binds $x to each element in the list expr

- LET $x = expr -- binds $x to the entire list expr
  - Useful for common subexpressions and for aggregations

Xquery -Example 3

```
<big_publishers>
    FOR $p IN distinct(document("bib.xml")//publisher)
    LET $b := document("bib.xml")/book[publisher = $p]
    WHERE count($b) > 100
    RETURN $p
</big_publishers>
```

count = a (aggregate) function that returns the number of elms

Xquery - Example 4

Find books whose price is larger than average:

```
LET $a=avg(document("bib.xml")/bib/book/@price)
FOR $b in document("bib.xml")/bib/book
WHERE $b/@price > $a
RETURN $b
```

FOR versus LET

```
FOR $x IN document("bib.xml")/bib/book

RETURN <result> $x </result>
```

Returns:

```
<result> <book>...</book></result>
<result> <book>...</book></result>
<result> <book>...</book></result>
...
```

```
LET $x := document("bib.xml")/bib/book

RETURN <result> $x </result>
```

Returns:

```
<result> <book>...</book>
         <book>...</book>
         <book>...</book>
         ...
</result>
```

Collections in XQuery

- Ordered and unordered collections
  - /bib/book/author = an ordered collection
  - Distinct(/bib/book/author) = an unordered collection
- LET $a = /bib/book → $a is a collection
- $b/author → a collection (several authors...)

```
RETURN <result> $b/author </result>
```

Returns:

```
<result> <author>...</author>
         <author>...</author>
         <author>...</author>
         ...
</result>
```

Collections in XQuery

What about collections in expressions ?

- $b/@price → list of n prices
- $b/@price * 0.7 → list of n numbers
- $b/@price * $b/@quantity → list of n x m numbers ??
- $b/@price * ($b/@quant1 + $b/@quant2) ≠ $b/@price * $b/@quant1 + $b/@price * $b/@quant2 !!

Sorting in XQuery

```
<publisher_list>
   FOR $p IN distinct(document("bib.xml")//publisher)
   RETURN <publisher> <name> $p/text() </name> ,
                  FOR $b IN
document("bib.xml")//book[publisher = $p]
                  RETURN <book>
                                 $b/title ,
                                 $b/@price
                           </book> SORTBY(price
DESCENDING)
               </publisher> SORTBY(name)
</publisher_list>
```

If-Then-Else

```
FOR $h IN //holding

RETURN <holding>

            $h/title,

            IF $h/@type = "Journal"

                  THEN $h/editor

            ELSE $h/author

      </holding> SORTBY (title)
```

Existential Quantifiers

```
FOR $b IN //book
WHERE SOME $p IN $b//para SATISFIES
   contains($p, "sailing")
   AND contains($p, "windsurfing")
RETURN $b/title
```

Universal Quantifiers

```
FOR $b IN //book
WHERE EVERY $p IN $b//para SATISFIES
   contains($p, "sailing")
RETURN $b/title
```

Group-By in Xquery

```
FOR $b IN document("http://www.bn.com")/bib/book,
    $y IN $b/@year
WHERE $b/publisher="Morgan Kaufmann"
RETURN    GROUPBY $y
          WHERE count($b) > 10
          IN <year> $y </year>
```

← with GROUPBY

Equivalent SQL →

```
SELECT year
FROM Bib
WHERE Bib.publisher="Morgan Kaufmann"
GROUPBY year
HAVING count(*) > 10
```

Query - Example 6

- Example: Return a flat list of supplier names and their part descriptions for the parts that are actually supplied, in alphabetic order.

JOINS in Relation

P (part)

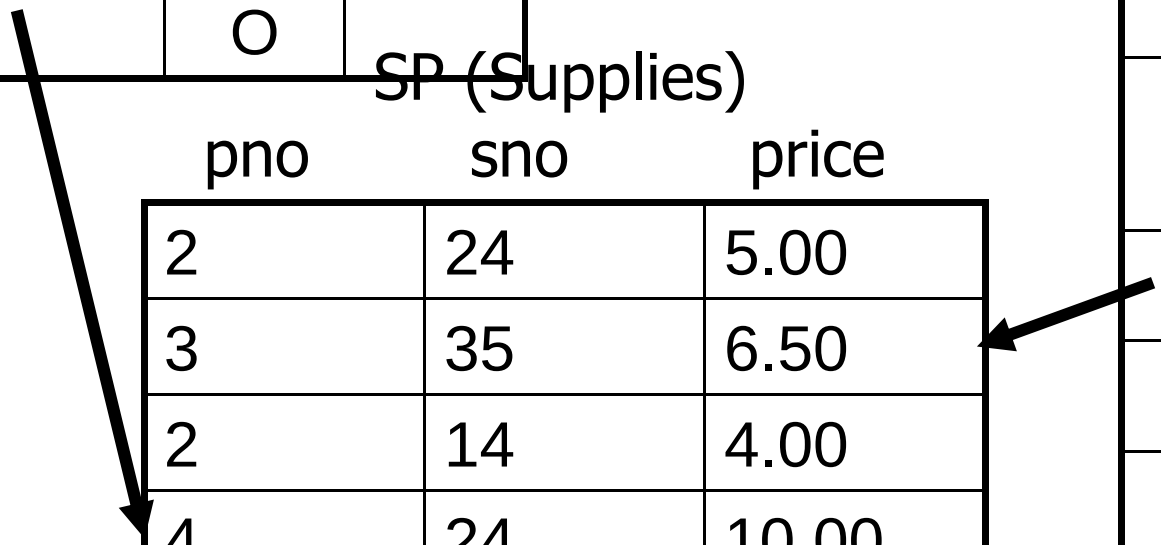
pno	descrip	qnty
1	ABC	100
2	DEF	75
3	GHI	36
4	JKL	2
5	MNO	0

S (supplier)

sno	name	locat
27	IBM	NY
35	MSFT	WSH
8	LSN	JAX
14	AMD	CA
51	AJR	BNA
24	UF	GNV

SP (Supplies)

pno	sno	price
2	24	5.00
3	35	6.50
2	14	4.00
4	24	10.00
1	27	2.25



XML documents

P.XML

```
<parts>
  <p_tuple>
    <p_no>
      1
    </p_no>
    <descrip>
      ABC
    </descrip>
    <qty>
      100
    </qty>
  </p_tuple>
</parts>
```

S.XML

```
<supplier>
  <s_tuple>
    <s_no>
      27
    </s_no>
    <name>
      IBM
    </name>
    <locat>
      NY
    </locat>
  </s_tuple>
</supplier>
```

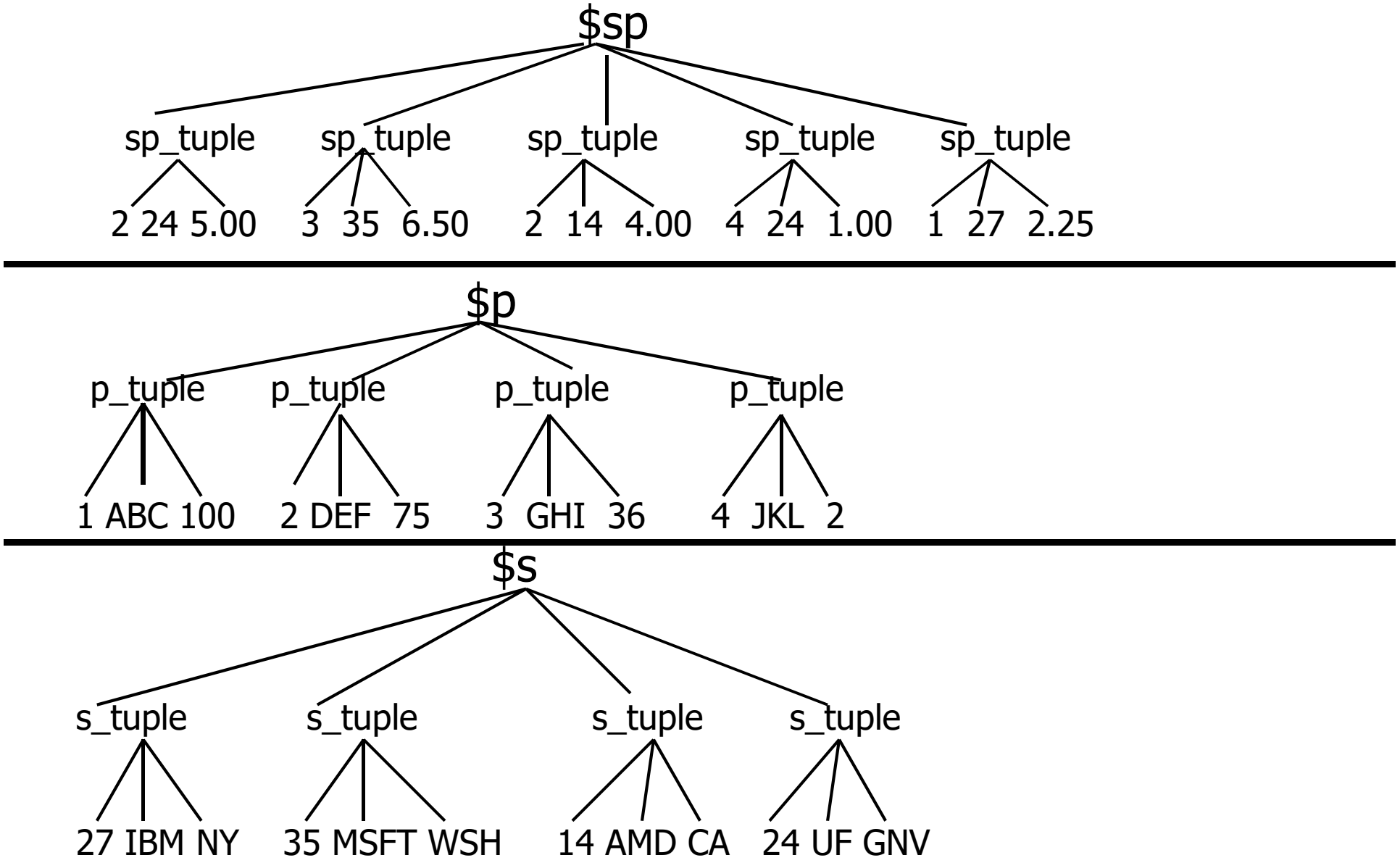
SP.XML

```
<supplies_part>
  <sp_tuple>
    <p_no>
      2
    </p_no>
    <s_no>
      24
    </s_no>
    <price>
      5.00
    </price>
  </p_tuple>
</supplies_part>
```

JOINS in XQuery

```
For $sp in document("sp.xml")//sp_tuple,
     $p  in document("p.xml")//p_tuple[
                              pno = $sp/pno]
    $s in document("s.xml")//s_tuple
                              [sno = $sp/sno]
Return <sp_pair> {
         $s/name, $p/descrip }
         </sp_pair> sortby(sname, descrip)
```

Binding of Joins in Xquery



More Examples

```
<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="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>
```

List books published by Addison-Wesley after 1991, including their year and title.

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

Expected Result

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

More Examples - 2

- Create a flat list of all the title-author pairs, with each pair enclosed in a "result" element.

```
<results> {
for $b in
    document("http://www.bn.com")/bib/book, $t
    in $b/title,
$a in $b/author
return <result>
{ $t }
{ $a } </result>
}
</results>
```

Expected Results

<results>
<result> <title>TCP/IP Illustrated</title>
<author>
<last>Stevens</last>
<first>W.</first>
</author>

More Example - 3

- For each book found at both bn.com and amazon.com, list the title of the book and its price from each source.

```
<books-with-prices>
{ for $b in document("www.bn.com/bib.xml")//book,
$a in
   document("www.amazon.com/reviews.xml")//entry
where $b/title = $a/title
return <book-with-prices>
{ $b/title }
<price-amazon>{ $a/price/text() }</price-amazon>
   <price-bn>{ $b/price/text() }</price-bn>
</book-with-prices> }
</books-with-prices>
```

More Examples -4

- For each book that has at least one author, list the title and first two authors, and an empty "et-al" element if the book has additional authors.

- <bib>

```
{ for $b in document("www.bn.com/bib.xml")//book
    where count($b/author) > 0
return <book> { $b/title }
{ for $a in $b/author[position()<=2]
return $a }
{ if (count($b/author) > 2) then
 <et-al/> else () }
</book> }
</bib>
```

```
<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>
```

More Examples - 5

- List the titles and years of all books published by Addison-Wesley after 1991, in alphabetic order.
- <bib>

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

More Examples

- http://www-106.ibm.com/developerworks/xml/library/x-xquery.html