XML Query Languages

XPATH

XQUERY

1

XPATH and XQUERY

- Two query language to search for features in XML documents
  - XPATH
  - XQUERY

2

XPATH

- XPATH is a language for describing paths in XML documents.
  - To be more precise it describes semistructured data graph and *its* paths.
  - XML documents documents can be described as semi-structured data graphs
    - Each subobject as a node of a graph with its subobjects as its children

3

XQUERY

- XQUERY is a full query language for XML documents with power similar to OQL.

4

Example DTD

```
<!DOCTYPE Rests [
   <!ELEMENT RESTS (REST*, SODA*)>
   <!ELEMENT REST (PRICE+)>
      <!ATTLIST REST name = ID>
   <!ELEMENT PRICE (#PCDATA)>
      <!ATTLIST PRICE theSoda = IDREF>
   <!ELEMENT SODA ()>
      <!ATTLIST SODA name = ID, soldBy = IDREFS>
]>
```

RESTS have 0 or more REST & SODA

Every REST has 1 or more PRICE and also an attr name

PRICE has data for price & the Soda with that price.

SODA name for ID and IDREFs for the rest that sell the soda.

5

Example Document

```
<RESTS>
   <REST name = "JoesRest">
      <PRICE theSoda ="Dew">1.50</PRICE>
      <PRICE theSoda = "Slice">1.75</PRICE>
   </REST> ...
   <SODA name = "Dew", soldBy = "JoesRest,
      SuesRest,...">
   </SODA> ...
</RESTS>
```

6

XPATH Path Descriptors

- Queries are really path descriptors
  - Look like UNIX path description with tags instead of directories and files
  - Tags are separated by /
- Simple path descriptors are sequences of tags separated by slashes (/).

7

XPATH Path Descriptors

- If the descriptor begins with /, then the path starts at the root and has those tags, in order.
- If the descriptor begins with //, then the path can start anywhere.

8

Example: /RESTS/REST/PRICE

```
<RESTS>
  <REST name = "JoesRest">
     <PRICE theSoda ="Dew">1.50</PRICE>
     <PRICE theSoda = "Slice">1.75</PRICE>
  </REST> ...
  <SODA name = "Dew", soldBy = "JoesRest,
     SuesRest,...">
  </SODA> ...
</RESTS>
```

/RESTS/REST/PRICE describes the set with these two PRICE objects as well as the PRICE objects for any other bars.

9

Example: //PRICE

```
<RESTS>
  <REST name = "JoesRest">
     <PRICE theSoda ="Dew">1.50</PRICE>
     <PRICE theSoda = "Slice">1.75</PRICE>
  </REST> ...
  <SODA name = "Dew", soldBy = "JoesRest,
     SuesRest,...">
  </SODA> ...
</RESTS>
```

//PRICE describes the same PRICE objects, but only because the DTD forces every PRICE to appear within a RESTS and a REST.

10

Wild-Card *

- A star (*) in place of a tag represents any one tag.
  - Acts as a "wildcard"
- Example: /*/*/PRICE represents all price objects at the third level of nesting.

11

Example: /RESTS/*

```
<RESTS>
  <REST name = "JoesRest">
     <PRICE theSoda ="Dew">1.50</PRICE>
     <PRICE theSoda = "Slice">1.75</PRICE>
  </REST> ...
  <SODA name = "Dew", soldBy = "JoesRest,
     SuesRest,...">
  </SODA> ...
</RESTS>
```

/RESTS/* captures all REST and SODA objects, such as these.

12

Attributes

- We may refer to attributes in addition to tags.
- In XPATH, we refer to attributes by prepending @ to their name.
- Attributes of a tag may appear in paths as if they were nested within that tag.

13

Example: /RESTS/*/@name

```
<RESTS>
  <REST name = "JoesRest">
     <PRICE theSoda ="Dew">1.50</PRICE>
     <PRICE theSoda = "Slice">1.75</PRICE>
  </REST> ...
  <SODA name = "Dew", soldBy = "JoesRest,
     SuesRest,...">
  </SODA> ...
</RESTS>
```

/RESTS/*/@name selects all name attributes of immediate subobjects of the RESTS object.

14

Selection Conditions

- A condition inside [...] may follow a tag.
- If so, the only paths included in the result of a path expression are ones that
  - have that tag and
  - also satisfy the condition

15

Example: Selection Condition

- /RESTS/REST/PRICE[PRICE < 1.60]

```
<RESTS>
  <REST name = "JoesRest">
     <PRICE theSoda = "Dew">1.50</PRICE>
     <PRICE theSoda = "Slice">1.75</PRICE>
  </REST> ...
```

The condition that the PRICE be < $1.60 makes this price but not the Slice price satisfy the path descriptor.

16

Example: Attribute in Selection

- /RESTS/REST/PRICE[@theSoda = "Slice"]

```
<RESTS>
  <REST name = "JoesRest">
     <PRICE theSoda = "Dew">1.50</PRICE>
     <PRICE theSoda = "Slice">1.75</PRICE>
  </REST> ...
```

Now, this PRICE object is selected, along with any other prices for Slice.

17

Axes

- In general, path expressions allow us to start at the root and execute a sequence of steps to find a set of nodes at each step.
  - At each step, we may follow any one of several *axes*.
- The default axis is child:: --- go to any child of the current set of nodes.

18

Example: Axes

- /RESTS/SODA is really shorthand for /RESTS/child::SODA .
- @ is really shorthand for the attribute:: axis. Thus,
  - /RESTS/SODA[@name = “Dew”] is shorthand for
    /RESTS/SODA[attribute::name = “Dew”]

19

More Axes

- Some other useful axes are:
  1. parent:: = parent(s) of the current node(s).
  2. descendant-or-self:: = the current node(s) and all descendants.
     - Note: // is really a shorthand for this axis.
  3. ancestor::, ancestor-or-self, etc.

20

XQUERY

- XQUERY allows us to query XML documents, using path expressions from XPATH to describe important sets.
- Corresponding to SQL’s select-from-where is the XQUERY *FLWR* (pronounced “flower”) *expression*, standing for “for-let-where-return.”

21

XQUERY ⇔ SQL

- where ⇔WHERE
- return ⇔SELECT
- for ⇔ FROM

22

FLWR Expressions

1. One or more FOR and/or LET clauses.
2. Then an optional WHERE clause.
3. A RETURN clause.

23

FOR Clauses

FOR <variable> IN <path expression>,...

- Variables begin with $.
- A FOR variable takes on each object in the set denoted by the path expression, in turn.
- Whatever follows this FOR is executed once for each value of the variable.
  - Creates a loop

24

Example: FOR

```
FOR $soda IN /RESTS/SODA/@name
RETURN
  <SODANAME>$soda</SODANAME>
```

- $soda ranges over the name attributes of all sodas in our example document.
- Result is a list of tagged names, like <SODANAME>Dew</SODANAME> <SODANAME>Slice</SODANAME>...

25

LET Clauses

LET <variable> := <path expression>,...

- Value of the variable becomes the ***set*** of objects defined by the path expression.
- Note LET does not cause iteration; FOR does.

26

Example: LET

```
LET $sodas := /RESTS/SODA/@name
RETURN
  <SODANAMES>$sodas</SODANAMES>
```

- Returns one object with all the names of the sodas, like:

<SODANAMES>Dew, Slice,...</SODANAMES>

27

Following IDREF's

- XQUERY (but not XPATH) allows us to use paths that follow attributes that are IDREF's.
- If *x* denotes a set of IDREF's, then *x*=>*y* denotes all the objects with tag *y* whose ID's are one of these IDREF's.

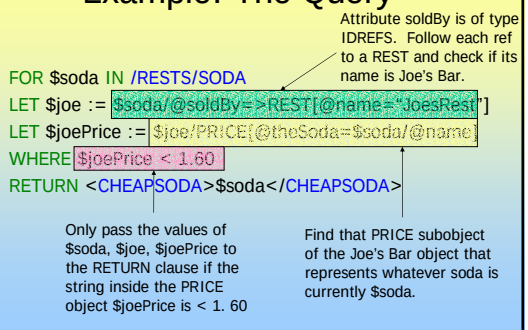
28

Example

- Find all the soda objects where the soda is sold by Joe's Rest for less than 1.60.
- Strategy:
  1. $soda will for-loop over all soda objects.
  2. For each $soda, let $joe be either the Joe's-Rest object, if Joe sells the soda, or the empty set of rest objects.
  3. Test whether $joe sells the soda for < 1.60.

29

Example: The Query



30