

Stephen Buxton

Dir. of Product Management

XML and Text

Oracle Corporation

Querying XML

SQL, SQL/XML and XQuery in Context

Proposition

“XQuery and SQL/XML are not *competing* technologies for querying XML, they are *complementary*.”

Querying XML

- Kinds of Data
- Data Representation
- Data Storage
- Querying XML
 - XPath
 - XQuery
 - SQL/XML
 - Oracle and XML
- Summary

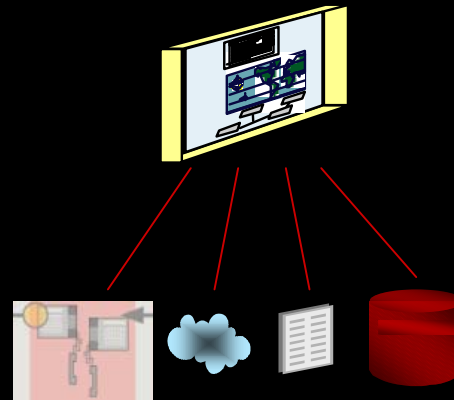
Kinds of Data

XML, (Object-)Relational data

Storing Content and Data
(Data Consolidation)



Integration
(Access Consolidation)



Messaging



Querying XML

- Kinds of data
- **Data Representation**
- Data Storage
- Querying XML
 - XPath
 - XQuery
 - SQL/XML
 - Oracle and XML
- Summary

Data Representation

- An abstraction of the data – *not* the data itself
- The same data could be represented in many ways
- Consider:
 - Natural Language
 - XML
 - SQL

A resolution presented to the U.S. House of Representatives, entitled “Welcoming the accession of Bulgaria, Estonia, Latvia, Lithuania, Romania, Slovakia, and Slovenia to the North Atlantic Treaty Organization (NATO). It was created 11th March 2004, sponsored by Bereuter, and it “welcomes with enthusiasm the accession of Bulgaria, Estonia, Latvia, Lithuania, Romania, Slovakia, and Slovenia to the North Atlantic Treaty Organization (NATO)”.

XML Representation

```
<resolution dms-id="42" public-private="public">
  <congress>108</congress>
  <session>2</session>
  <legis-num>558</legis-num>
  <action>
    <action-date>20040311</action-date>
    <action-desc>
      <sponsor>Bereuter</sponsor>
      <cosponsor>Wexler</cosponsor>
      <cosponsor>Gillmor</cosponsor>
      <cosponsor>Shimkus</cosponsor>
      <committee-name>Committee on International Relations
    </committee-name>
    </action-desc>
  </action>
  <official-title>Welcoming ... </official-title>
  <resolution-body>
    <paragraph> ... </paragraph>
    <paragraph> ... </paragraph>
    <paragraph> ... </paragraph>
  </resolution-body>
</resolution>
```

XML Gives you ...

- Distinct, named data fields (elements)
- Document order (except attributes)
- Hierarchical, named structure (sub-elements)
- Flexibility+

```
<official-title>Welcoming ...  
  </official-title>
```

```
<paragraph> ... </paragraph>  
<paragraph> ... </paragraph>  
<paragraph> ... </paragraph>
```

```
<action>  
  <action-date>20040311</action-date>  
  <action-desc>  
    <sponsor>Bereuter</sponsor>  
    <cosponsor>Wexler</cosponsor>  
    <committee-name>Committee on  
International Relations  
  </committee-name>  
  </action-desc>  
</action>
```

Typed XML Adds ...

- Simple types
 - double, string, ...
 - Facets (e.g. max, min)
 - Enumeration
- Complex types
 - Sequences
 - Occurrences
- Expressed as XML in an XML Schema document

SQL Representation

Id	Congress	Session_num	Legis_num	Action_date	Committee_name	Official_title
42	108	2	558	20040311	Committee on International Relations	Welcoming the
Etc.						

Resolution_id	Sponsor	Sponsor_type
42	Bereuter	primary
42	Wexler	Co-sponsor
42	Gillmor	Co-sponsor
Etc.		

Resolution_id	P_order	paragraph
42	1	Welcomes with ...
42	2	Reaffirms that ...
42	3	Agrees that ...
Etc.		

SQL Gives You ...

- Distinct, named data fields (elements)
- Relations between data fields
- A set-based algebra over the data, relations
- Type and relation information, in a data dictionary
- No default order
- In addition:
 - Constraints
 - Indexing, tools, ... built up over decades

SQL is not SQL-92

SQL has a number of ways to represent data that is not “table-shaped”

- Mater-detail tables
- Variable-length Arrays (VARRAYS)
- Nested tables
- Objects (SQL:1999)
- SQL/XML
- Plus extensions

SQL/XML

- SQLX working group, <http://sqlx.org>
 - Formed to generate, query XML in SQL
- Emerging part of the ANSI/ISO SQL standard
- Defines mappings, types and functions
 - Data type mappings XML Schema <-> SQL
 - XML Type
 - “publishing” functions: *XMLAgg*, *XMLConcat*, *XMLElement*, *XMLForest*
- Oracle has added extensions (anticipating future versions of the standard)
 - Functions: *existsNode*, *extract*, *extractValue*

SQL/XML adds ...

- XMLType

Table Resolutions_xml

ID	Official-title	Resolution
42	Welcoming the accession of Bulgaria, Estonia, Latvia, Lithuania, Romania, Slovakia, and Slovenia to the North Atlantic Treaty Organization (NATO), and for other purposes.	<pre><resolution dms-id="42" public-private="public"> <congress>108</congress> <session>2</session> <legis-num>558</legis-num> <action> <action-date>20040311</action-date> <action-desc> <sponsor>Bereuter</sponsor> <cosponsor>Wexler</cosponsor> <cosponsor>Gillmor</cosponsor> ... </action-desc> <paragraph> ... </paragraph> </resolution-body> </resolution></pre>

SQL/XML Publishing func's

- XML *view* of SQL

```
CREATE OR REPLACE VIEW resolutions_xml_view AS
(SELECT
  r.id AS id ,
  XMLElement ("resolution" ,
    XMLElement ("congress" , congress) , ...
    XMLElement ("action" ,
      XMLElement ("action-date", to_char(action_date, 'YYYYMMDD')),
      XMLElement ("action-desc" , ...
    (SELECT (XMLAGG(XMLElement("cosponsor", sponsor)))
      FROM resolutions_sponsors s
      WHERE s.resolution_id = r.id
      AND s.sponsor_type = 'cosponsor'
    ) , ...
  )
) AS resolution
FROM resolutions r
)
```

Oracle adds

- SQL *view* of XML

```
CREATE OR REPLACE VIEW resolutions_view AS (  
  SELECT  
    id AS id ,  
    extractValue( resolution, '/resolution/congress' ) AS congress ,  
    to_number( extractValue( resolution, '/resolution/session' ) ) AS session_num ,  
    extractValue( resolution, '/resolution/legis-num') AS legis_num ,  
    to_date( extractValue( resolution, '/resolution/action/action-date'), 'YYYYMMDD' ) AS  
action_date,  
    extractValue( resolution, '/resolution/action/action-desc/committee-name') AS  
committee_name ,  
    extractValue( resolution, '/resolution/official-title') AS official_title  
FROM resolutions_xml  
)
```

SQL XML Duality

- Representation Duality
 - SQL \leftrightarrow XML
 - SQL \leftrightarrow SQL'
 - XML \leftrightarrow XML'
- Flip between representations
 - For publishing
 - For migration
- Note: “Storage” is an abstraction ...

Representation Choice

- Interfaces
 - Where is the data coming from ?
 - going to ?
- Tools
- Query, Storage choices are *de-coupled* from Representation

Querying XML

- Kinds of data
- Data Representation
- **Data Storage**
- Querying XML
 - XPath
 - XQuery
 - SQL/XML
 - Oracle and XML
- Summary

Data Storage

- LOB storage
 - Collection of data stored in a single Large Object
- Object-relational storage
 - Tables, nested tables, objects
 - XML can be “shredded”
- XMLType
 - is a Native SQL data type
 - is an abstraction over several physical storage models
 - Query (and other operations) on XMLType, though storage model may change

XML Type Benefits

- Native XML Storage – ORX database
- Performance and Simplicity for XML operations (query, update, insert, delete)
- Typed or Untyped (XML Schema)
- Range of Indexes:
 - XML Path,
 - Full-Text,
 - B-Tree indexes
 - *etc. ...*

XML Type Storage Examples

“shredded”: object-relational

- DOM fidelity
- Partial Update/Insert/Delete
 - LOB later
- Constraints, Triggers
- Indexes
 - Full range of SQL indexes
 - Includes
 - XML Path index
 - Full-Text index

LOB: single Large Object

- Document fidelity
- XML in, XML out, little processing
- Mixed/changing structures
- Indexes
 - XML Path index
 - Full-Text index

Any Representation, Any Storage

<i>Representation:</i>	SQL	XML	Mixed
<i>Storage:</i> Relational	Relational	XML View	Relational + XML View
XMLType	SQL View (Forest, ...)	XMLType	SQL View + XMLType
Mixed	Relational + SQL View	XML View + XMLType	SQL View + XML View + Relational + XMLType

Querying XML

- Kinds of data
- Data Representation
- Data Storage
- Querying XML
 - XPath
 - XQuery
 - SQL/XML
 - Oracle and XML
- Summary

Querying

- *“to ask questions of, especially with a desire for authoritative information”* (Merriam-Webster)
- *“... retrieving records or parts of records and performing various calculations before displaying the results ...”* (Brittanica)

Querying XML

- Query Storage or Representation
- Choice of query language:
 - XPath
 - XQuery
 - SQL/XML
- *not* mutually exclusive – can use all three
- Oracle adds extensions to all three

Querying XML

A query language must provide:

- **Selection**: choose only some parts of the data, according to some criteria
 - In SQL, you select rows from a table
 - When querying XML, you select nodes or node-sequences (from the XML tree)
- **Projection**: once you have the data that is interesting, return only some parts of it
 - In SQL, you project columns from a table (set of tuples)
 - When querying XML, you project nodes or node-sequences⁺ (from the XML tree)

XPath

- XPath 1.0 – W3C Recommendation Nov-1999
- Used extensively in XSLT (and elsewhere)
- Describes a path to the data, with predicates
- Assumes a *context*

XPath example, showing **Selection** and **Projection**:

```
/resolutions/resolution[action/action-desc/sponsor="Bereuter"]/official-title
```

XQuery

- XQuery 1.0 – W3C *Last Call* Jan-2005?
- Includes XPath (2.0)

XQuery adds:

- Expressions
 - notably the FLWOR expression
- Variables
- Joins
- Strong typing, based on XML Schema
- Constructors
 - construct XML fragments as (intermediate) results

XQuery Example

```
for $i in /resolutions/resolution
```

```
  let $action := $i/action
```

```
  where $action/action-date = "20040311"
```

```
  order by $i/@dms-id
```

```
return
```

```
  <result>
```

```
    {$action/action-desc/sponsor}
```

```
  </result>
```

- *Assumes a context*
 - ... or input function such as doc(), collection()

SQL

```
SELECT id, Official-title  
FROM Resolutions  
WHERE Official-title = 'Welcoming the accession ...'
```

To unextended SQL, this is a “black box”



ID	Official-title	Resolution
42	Welcoming the accession of Bulgaria, Estonia, Latvia, Lithuania, Romania, Slovakia, and Slovenia to the North Atlantic Treaty Organization (NATO), and for other purposes.	<pre><resolution dms-id="42" public-private="public"> <congress>108</congress> <session>2</session> <legis-num>558</legis-num> ... </resolution></pre>

XPath in SQL/XML⁺

3 SQL extension functions:

`extract(XML Instance, XPath expression)`

Returns node*

`extractValue(XML Instance, XPath expression)`

Returns value

`existsNode(XML Instance, XPath expression)`

Returns 1 or 0

+ anticipating the functions XMLQuery, XMLEExists, XMLCast, expected in SQL:2005

XPath in SQL/XML+

```
SELECT id, extractValue(Resolution, '/resolution/congress')
FROM resolutions_xml
WHERE Official_title = 'Welcoming the accession ...'
AND existsNode(Resolution, '/resolution[legis-num=558]')=1
```

ID	Official-title	Resolution
42	Welcoming the accession of Bulgaria, Estonia, Latvia, Lithuania, Romania, Slovakia, and Slovenia to the North Atlantic Treaty Organization (NATO), and for other purposes.	<resolution dms-id="42" public-private="public"> <congress>108</congress> <session>2</session> <legis-num>558</legis-num> ... </resolution>

XQuery in SQL/XML

2 SQL extension functions:

XMLQuery (*XQuery* passing ... as ... returning ...)
Returns node* (or ...)

XMLTable(*XQuery* columns [*col-name col-type* PATH
XPath]*)
Returns Table

XMLQuery - repository

SELECT

XMLQUERY(

'for \$v in doc("/public/xml2004/vote.xml")/rollcall-vote

let \$d := \$v/vote-data

return

<vote-count>

<total>

{count(\$d/recorded-vote[vote="Yea"])}

</total>

<republican>

{count(\$d/recorded-vote[legislator/@party="R" and vote="Yea"])}

</republican>

<democrat>

{count(\$d/recorded-vote[legislator/@party="D" and vote="Yea"])}

</democrat>

</vote-count>' returning content)

AS result FROM DUAL

XMLQuery - repository

RESULT

```
<vote-count>
  <total>7</total>
  <republican>4</republican>
  <democrat>3</democrat>
</vote-count>
```

XMLQuery - table

```
SELECT
  XMLQUERY(
    'for $v in $Vote/rollcall-vote
     let $d := $v/vote-data
     return
<vote-count>
  <total>
    {count($d/recorded-vote[vote="Yea"])}
  </total>
  <republican>
    {count($d/recorded-vote[legislator/@party="R" and vote="Yea"])}
  </republican>
  <democrat>
    {count($d/recorded-vote[legislator/@party="D" and vote="Yea"])}
  </democrat>
</vote-count>' passing vote as "Vote" returning content)
  AS result FROM votes_xml
```

XMLQuery - table

RESULT

```
<vote-count>
  <total>7</total>
  <republican>4</republican>
  <democrat>3</democrat>
</vote-count>
```

XMLQuery – ora:view

```
SELECT
  XMLQUERY(
    'for $v in
      ora:view("votes_xml")/ROW/VOTE/rollcall-vote
    let $d := $v/vote-data
    return
<vote-count>
  <total>
    {count($d/recorded-vote[vote="Yea"])}
  </total>
  <republican>
    {count($d/recorded-vote[legislator/@party="R" and vote="Yea"])}
  </republican>
  <democrat>
    {count($d/recorded-vote[legislator/@party="D" and vote="Yea"])}
  </democrat>
</vote-count>' returning content)
  AS result FROM DUAL
```

XMLQuery – ora:view

RESULT

```
<vote-count>  
  <total>7</total>  
  <republican>4</republican>  
  <democrat>3</democrat>  
</vote-count>
```

XMLTable - simple

```
SELECT * FROM
XMLTABLE( 'for $r in
  doc("/public/xml2004/resolution.xml")/resolution
  let $a := $r/action
  where $a/action-date="20040311"
  order by $r/legis-num ascending
  return
    <all-sponsors>
      {$a/action-desc/sponsor}
      {$a/action-desc/cosponsor}
    </all-sponsors>')
```

Returns a table with 1 column of type
XMLType

XMLTable - simple

COLUMN_VALUE

```
<all-sponsors>
  <sponsor>Bereuter</sponsor>
  <cosponsor>Wexler</cosponsor>
  <cosponsor>Gillmor</cosponsor>
  <cosponsor>Shimkus</cosponsor>
</all-sponsors>
```

XMLTable - columns

```
SELECT resolution_id, sponsor, sponsor_type
FROM XMLTABLE( 'for $r in doc("resolution.xml")/resolution
let $a := $r/action/action-desc
where $r/action/action-date="20040311"
return ( <sponsor>
  <resid>{$r/@dms-id}</resid>
  <type>primary</type>
  <name>{$a/sponsor/text()}</name>
</sponsor>,
for $j in $a/cosponsor
return <sponsor>
  <resid>{$r/@dms-id}</resid>
  <type>cosponsor</type>
  <name>{$j/text()}</name>
</sponsor> )'
```

COLUMNS

```
resolution_id number PATH 'resid',
sponsor        varchar2(4000) PATH 'name/text()',
sponsor_type   varchar2(200) PATH 'type'
```

```
) resolutions_sponsors
```

XMLTable - columns






```
RESOLUTION_ID SPONSOR          SPONSOR_TYPE
-----
              42 Bereuter      primary
              42 Wexler       cosponsor
              42 Gillmor      cosponsor
              42 Shimkus      cosponsor
```

4 rows selected.

Querying XML

<i>Representation:</i>	SQL	XML	Mixed
<i>Query:</i> SQL	Relational	Black Box	Black Box
XQuery Engine	Black Box	XPath (+ XQuery)	Black Box
SQL/XML	Relational	XPath (+XQuery)	Relational + XPath (+ XQuery)

Querying XML

	<i>Representation:</i>		
	SQL	XML	Mixed
<i>Query:</i> SQL	Relational 	Text Only	Text Only
XQuery Engine	Must Convert	XPath (XQuery) 	Must Convert
SQL/XML	Relational 	XPath (XQuery) 	Relational + XPath (XQuery) 

Querying XML

	<i>Representation:</i>		
	SQL	XML	Mixed
<i>Query:</i>			
SQL	Relational	Black Box	Black Box
XQuery Engine		(+ XQuery)	X
SQL/XML	Relational	XPath (+XQuery)	Relational + XPath (+ XQuery)

SQL/XML allows you to mix'n'match:

- Storage/representation
- Selection/projection using SQL, XQuery

Query Choices

- Tools
- Skills
- APIs (e.g. JDBC)
- Expressive power, naturalness
- Representation, Storage choices are *de-coupled* from Query

Querying XML

- Kinds of data
- Data Representation
- Data Storage
- Querying XML
 - XPath
 - XQuery
 - SQL/XML
 - Oracle and XML
- **Try doing this with XQuery**
- Summary

Full-Text Search

```
SELECT id
  FROM resolutions_xml
 WHERE CONTAINS( resolution,
                '(bulgaria and slovenia) within official\title')>0
```

or ...

```
SELECT
  XMLQUERY(
    'for $r in
      $res/resolution/official-title[ora:contains(., "bulgaria and slovenia")>0]
    return
      $r' passing resolution as "res" returning content)
  AS result FROM resolutions_xml
```

Querying

- “to ask questions of, especially with a desire for authoritative information” (Merriam-Webster)
- “... retrieving records or parts of records and performing various calculations before displaying the results ...” (Brittanica)
- “... A database query can be either a select query or an action query. A select query is simply a data retrieval query. An action query can ask for additional operations on the data, such as *insertion, updating, or deletion*. ...”
(searchDatabase.com)

Update XML

```
UPDATE votes_xml
SET vote = UPDATEXML( vote,
    '/rollcall-vote/vote-data/recorded-vote[2]/vote/text()',
    'Nay' )
WHERE id = 42
```

Querying XML

- Kinds of data
- Data Representation
- Data Storage
- Querying XML
 - XPath
 - XQuery
 - SQL/XML
 - Oracle and XML
- **Summary**

Benefits of the Extended-SQL approach

- SQL is mature, robust
- Additional functionality, control
 - Constraints
 - Triggers
- Most of the world's critical data is in SQL
- Decades of SQL investment:
 - People's skills
 - Tools
 - Performance

Benefits of the Extended-SQL approach (cont'd)

- SQL/XML⁺ is a convenient bridge into XML
 - Query XML with familiar commands, tools
 - Query across XML and object-relational data
- XPath, XQuery not yet complete
 - Update
 - Full-Text retrieval
 - OLAP functions
- XPath, XQuery need a “harness”
 - Assume some context
 - SQL provides a natural harness for XML queries

All the benefits of XPath, XQuery, SQL, plus simple migration and a robust context

Proposition

“XQuery and SQL/XML are not *competing* technologies for querying XML, they are *complementary*.”

Proposition

“XQuery and SQL/XML are not *competing* technologies for querying XML, they are *complementary*.”

- XQuery can directly query XML representation (data + structure)

Proposition

“XQuery and SQL/XML are not *competing* technologies for querying XML, they are *complementary*.”

- XQuery can directly query XML representation (data + structure)
- SQL provides a robust, proven framework

Proposition

“XQuery and SQL/XML are not *competing* technologies for querying XML, they are *complementary*.”

- XQuery can directly query XML representation (data + structure)
- SQL provides a robust, proven framework
- SQL/XML⁺ provides the best of all worlds

A large, stylized graphic of the letters 'Q' and 'A' in a dark grey, serif font. A large, bright red ampersand is superimposed over the 'Q' and 'A'. In the center of this graphic, the words 'QUESTIONS' and 'ANSWERS' are written in a white, bold, sans-serif font, stacked vertically.

QUESTIONS
ANSWERS