SQL/XML and the SQLX Informal Group of Companies

Andrew Eisenberg
IBM, Westford, MA 01886
andrew.eisenberg@us.ibm.com

Jim Melton
Oracle Corp., Sandy, UT 84093
jim.melton@acm.org

Introduction

For many years now, the SQL standard has been maintained and enhanced by NCITS Technical Committee H2 on Database in the US and by the ISO/IEC JTC 1/SC32/WG3 Database Languages Working Group internationally. After the publication of SQL:1992, the groups began to publish SQL as a base document (SQL/Foundation) and a number of independent parts. These parts began with SQL/CLI (Call-Level Interface) and SQL/PSM (Persistent Stored Modules) in 1995 and 1996, respectively.

Interest in XML has been growing in the last several years among software vendors, user companies of all sizes, and within the standards community. NCITS H2 and SC32 both approved a project for a new part of SQL, part 14, XML-Related Specifications (SQL/XML), in the second half of 2000.

After this new project was approved, a small number of companies began to meet in order to explore the technology and develop proposals to begin fleshing out this new part of SQL. This informal group of companies has come to be known as The SQLX Group.

In roughly a year's time, several pieces of what might be termed "infrastructure" have been progressed and are included in the initial working draft of SQL/XML [2].

SQL/XML Subproject

This new part of SQL, SQL/XML, began with the approval by both bodies of a subproject proposal document [1]. This document provided the justification for this subproject request and a suggested program of work. The justification states, in part, the following:

> "One of the most intriguing and urgent requirements to arise from the appearance of XML is a well-defined relationship between XML and SQL. Vast quantities of business data are currently stored in SQL database systems and great demand exists for the ability to present that data in XML form to various client applications. By contrast, increasing amounts of less-traditional data ("documents") are being produced in XML formats and there is tremendous pressure to allow that data to be queried concurrently with traditional ("object-relational") data. In addition, the growing need to allow disparate systems to exchange data has caused significant attention to be paid to the use of XML as a "canonical data format" between such systems (e.g., on the Web)."

The program of work enumerates several capabilities that may be included in SQL/XML. This list is as follows:

- Specifications for the representation of SQL data (specifically rows and tables of rows, as well as views and query results) in XML form, and vice versa.
- Specifications associated with mapping SQL schemata to and from XML schemata. This may include performing the mapping between existing arbitrary XML and SQL schemata.
- Specifications for the representation of SQL Schemas in XML.
- Specifications for the representation of SQL actions (insert, update, delete).
- Specifications for messaging for XML when used with SQL.
- Specifications of the (perhaps "a") manner in which SQL language can be used with XML.

If all of these capabilities were attempted as a single effort, then SQL/XML would not be published for a very long time. We feel confident that NCITS H2 and SC32/WG3 will all agree to some subset of these capabilities that will allow for publication in a timely manner. NCITS H2 and WG3 have a long history of adding features to their standards incrementally.

XML itself [3] is a recommendation of the W3C. SQL/XML is attempting to build upon the foundation provided by XML, XML Namespaces [4],

and XML Schema [5]. NCITS H2 and SC32/WG3 will endeavor to make sure that they neither duplicate nor conflict with these recommendations.

The SQLX Group

The level of interest in SQL/XML is so high that several of the companies participating in NCITS H2 have chosen to work together on developing proposals for this new part. These proposals, when they are ready, will be submitted to both NCITS H2 and SC32/WG3 for approval.

This informal group of companies has chosen SQLX as its name. SQLX meets roughly every 3 weeks, alternating teleconferences with face-to-face meetings. The group does not have formal rules or voting, choosing instead to move forward when a rough consensus of its members has been achieved.

One thing that cannot be said too often is that SQLX is an open group. Members of SQLX have, in several public forums, invited members of the broad technical community to participate. Our public web page can be found at www.sqlx.org. There are no fees or other requirements for membership. Membership in NCITS H2, SC32/WG3, or W3C is not a requirement for membership in SQLX.

Some of the largest DBMS vendors are participating in SQLX (our employers, IBM and Oracle, certainly are). We have some members that represent smaller vendors and some that represent potential end-users of this technology.

Initial SQL/XML Working Draft

In this section, we describe the features provided in the recently published initial SQL/XML Working Draft.

Mapping SQL Character Sets to and from Unicode

SQL allows the use of one or more named character sets. The choice of which character sets are supported and their internal representation are *implementation-defined*. That is to say that the vendor of each product specifies them. The characters in XML documents, in contrast, are Unicode characters.

Each product that supports SQL/XML will have to define a mapping between strings of each of its character sets and Unicode strings. It will also have to provide the reverse mapping, from Unicode strings to strings of each of its character sets.

SQL_TEXT is a character set that contains all of the characters that are used in the SQL language itself, as well as the characters in all of the character sets that an implementation supports. The mapping between SQL_TEXT and Unicode is known as the *plain text mapping* from SQL to XML.

Mapping SQL <identifier>s to XML Names

The range of characters that can be used in an SQL identifier is much greater than the range of characters that can be used in an XML Name. While this might not seem so at first, remember that SQL supports delimited identifiers (identifiers that are delimited by double-quote characters) such as "Max % ESPP".

XML Names that begin with the characters “XML” (in any case combination) are reserved by W3C for use in future recommendations. The XML Namespace recommendation uses the “:” character to separate the namespace prefix from the local part of the name (“xsd:string”, for example).

In order to bridge this gap, SQL/XML provides a mapping between the characters of SQL <identifier>s and XML Names. This mapping begins by mapping the characters of the SQL <identifier> to Unicode. SQL <identifier> characters that are valid characters in an XML Name are not changed. SQL <identifier> characters that are not valid XML Name characters are replaced with either “_xHHHH_” or “_xHHHHHHHH”, where “H” is an upper case hexadecimal digit.

In addition to these simple rules, there are some special cases that must be dealt with. The SQL <identifier> character “_” will always be represented by “_X005F_”. A leading “:” in an SQL <identifier> will be represented by “_x003A_”.

SQL/XML defines two flavors of this mapping. We have just illustrated the *partially-escaped* mapping.

The *fully-escaped* mapping has some additional rules beyond those we just mentioned. It maps all occurrences of “:”, not just as the initial character of the string, to “_x003A_”. Also, an SQL <identifier> that begins with the characters “X”, “M”, and “L” in any mixture of cases will be prefixed by “_xFFFF_”.

The following table shows how some SQL <identifier>s are mapped to XML Names:

SQL <identifier>	fully-escaped XML Name	partially-escaped XML Name
`employee`	`EMPLOYEE`	`EMPLOYEE`
`"employee"`	`employee`	`employee`
`"hire date"`	`hire_x0020_date`	`hire_x0020_date`
`"comp_plan"`	`comp_x005F_plan`	`comp_x005F_plan`
`"dept:id"`	`dept_x003A_id`	`dept:id`
`xmlcol`	`_xFFFF_xmlcol`	`xmlcol`

One might reasonably expect the partially-escaped mapping to be used for SQL <identifier>s that have been chosen with the mapping to XML Names in mind. The fully-escaped mapping allows the mapping of SQL <identifier>s that were chosen without any cognizance at all of XML.

Mapping XML Names to SQL <identifier>s

The mapping from an XML Name to an SQL <identifier> is very straightforward. The characters of an XML Name are examined from left to right. If the sequence "_xHHHH_" or "_xHHHHHHHH_" is encountered, then it is mapped to the Unicode character identified by that codepoint. If the XML Name begins with "_xFFFF_", then these initial characters are ignored.

The mapping between Unicode characters and SQL_TEXT is then applied. If there are any Unicode characters that cannot be mapped to SQL_TEXT, then an exception is raised.

These two mappings allow an SQL <identifier> to be mapped to an XML Name, and later to have the XML Name mapped back into the same SQL <identifier>. This fully-reversible property does not hold for the mapping of an XML Name to an SQL <identifier> and back again.

Mapping SQL data types to XML Schema data types

XML Schema Part 2 defines simple data types for XML and lexical representations for the values of these types. SQL/XML provides a mapping for each of SQL's scalar data types to an XML Schema data type.

The approach SQL/XML has taken is to select the closest possible XML Schema data type for each SQL data type. By "closest", we mean that it must allow all SQL values of this type to be represented and that it must allow the fewest values that are not SQL values. XML Schema *facets* are used to restrict the acceptable XML values to as closely as possible match the legal values SQL values. In the case of SQL's INTEGER type, for example, the XML Schema type of integer is used. The minInclusive and maxInclusive facets are set to the values of the implementation's smallest and largest INTEGER value, respectively.

The XML Schema annotation mechanism is used by SQL/XML to preserve SQL type information that would otherwise be lost in the mapping to an XML Schema data type. The use of this annotation mechanism is not required by SQL/XML, so an implementation is free to omit this part of the mapping if it chooses to.

In our examples, the "xsd" namespace prefix is used to indicate the XML Schema namespace. The "sqlxml" namespace prefix is used to indicate the SQL/XML namespace. A URI has not yet been chosen for the SQL/XML namespace, but we can report that work in this area has already been started. "xsd:annotation" is the name of an element defined by XML Schema. "sqlxml:sqltype" is the name of an element defined by SQL/XML to metadata specific to SQL.

The mapping of SQL data types to XML Schema data types is illustrated in the following examples:

```
CHAR (10) CHARACTER SET LATIN1 COLLATION
DEUTSCH
```

→

```
<xsd:simpleType>
   <xsd:restriction base="xsd:string">
      <xsd:length value="10"/>
      <xsd:annotation>
         <sqlxml:sqltype name="CHAR"
            length="10"
            characterSetName="LATIN1"
            collation="DEUTSCH"/>
      </xsd:annotation>
   </xsd:restriction>
</xsd:simpleType>
```

```
CHAR (10) CHARACTER SET LATIN1 COLLATION
DEUTSCH
```

→ (without annotation)

```
<xsd:simpleType>
   <xsd:restriction base="xsd:string">
      <xsd:length value="10"/>
   </xsd:restriction>
</xsd:simpleType>
```

```
INTEGER
```

→

```
<xsd:simpleType>
   <xsd:restriction base="xsd:integer">
      <xsd:maxInclusive value="2157483647"/>
      <xsd:minInclusive
          value="-2157483648"/>
      <xsd:annotation>
         <sqlxml:sqltype name="INTEGER"/>
      </xsd:annotation>
   </xsd:restriction>
</xsd:simpleType>
```

```
DECIMAL (8, 2)
→
<xsd:simpleType>
   <xsd:restriction base="xsd:decimal">
      <xsd:precision value="9"/>
      <xsd:scale value="2"/>
      <xsd:annotation>
         <sqlxml:sqltype name="DECIMAL"
            userPrecision="8"
            scale=" 2"/>
      </xsd:annotation>
   </xsd:restriction>
</xsd:simpleType>

REAL
→
<xsd:simpleType>
   <xsd:restriction base="xsd:float">
      <xsd:annotation>
         <sqlxml:sqltype name="REAL"
            precision="24"
            minExponent="-149"
            maxExponent="104"/>
      </xsd:annotation>
   </xsd:restriction>
</xsd:simpleType>

INTERVAL YEAR(4) TO MONTH
→
<xsd:simpleType>
   <xsd:restriction
         base="xsd:timeDuration">
      <xsd:pattern value=
         "-?P\p{Nd}{1,4}Y\p{Nd}{2}M"/>
      <xsd:annotation>
         <sqlxml:sqltype
            name="INTERVAL YEAR TO MONTH"
            leadingPrecision="4"/>
      </xsd:annotation>
   </xsd:restriction>
</xsd:simpleType>
```

We hope that these examples were self-explanatory. The careful reader may have noticed that the mapping of DECIMAL (8,2) had an XML precision of 9, and an SQL precision of 8. This is possible because an SQL implementation is allowed to use a value for precision that is greater than or equal to the precision that was specified. The XML value of precision reflects the value of precision that the implementation chose.

Mapping SQL values to XML values

The mapping of SQL values to XML values is largely determined by the mapping from the SQL data type to the XML Schema data type.

This mapping of values can be seen in the following examples:

SQL data type	SQL literal	XML value
VARCHAR (10)	'Smith'	Smith
INTEGER	10	10
DECIMAL (5,2)	99.52	99.52
TIME	TIME'12:30:00'	12:30:00
INTERVAL HOUR TO MINUTE	INTERVAL'2:15'	PT02H15M

Future Work

So far, we have only laid the groundwork for the bulk of the SQL/XML effort, and even this infrastructure is incomplete. We have not yet dealt with how null values will map to XML values. We have also not yet specified how SQL's User-Defined Types or Arrays are mapped to XML Schema data types.

In a year or so we expect to write another article that will update our readers by describing SQL/XML at a stage much nearer to its completion.

References

[1] *Subproject: "XML-Related Specs (SQL/XML)"*, ISO/IEC JTC 1/SC 32 N00575, WG3:HEL-026R2, H2-2000-331R2, Jim Melton, October 10, 2000.

[2] *XML-Related Specifications (SQL/XML) – Working Draft SQL:200n Part 14*, H2-2001-149, WG3:YYJ-012, Jim Melton (Editor), June 18, 2001, available at http://www.sqlx.org.

[3] *Extensible Markup Language (XML) Version 1.0 (second edition)*, October 2, 2000, http://www.w3.org/TR/REC-xml.

[4] Namespaces in XML, W3C Recommendation, Tim Bray, Dave Hollander, and Andrew Layman (Editors), January 14, 1999, http://www.w3.org/TR/1999/REC-xml-names-19990114.

[5] *XML Schema Part 2: Datatypes*, W3C Recommendation, Paul V. Biron and Ashok Malhotra (Editors), May 2, 2001, http://www.w3.org/TR/2001/REC-xmlschema-2-20010502.

Web References

National Committee for Information Technology Standards (NCITS)
http://www.ncits.org

NCITS H2 – Database Committee
http://www.ncits.org/tc_home/h2.htm

ISO/IEC JTC 1/SC32
http://www.jtc1sc32.org

SQLX http://www.sqlx.org

W3C http://www.w3.org