Almacenamiento de datos XML en Oracle 11g

Lic. Silvina Migani, Lic. Cristina Vera, Ing. Carlos Correa, Lic. Liliana Romera

Departamento de Informática / Facultad de Ciencias Exactas, Físicas y Naturales / UNSJ
Av. Ignacio de la Roza 590 (O)
Teléfonos: 4260353 - 4260355
silvina.migani@gmail.com; civera2@yahoo.com.ar; carcorrea@hotmail.com.ar; liliromera@yahoo.com.ar

Resumen

Sin duda la aparición del lenguaje de marcas extensible, XML (eXtensible Markup Language), generó muchos cambios en el mundo de la informática. Por un lado, la introducción de muchas tecnologías asociadas, y por otro, la transformación o adecuación de las ya existentes. El mundo de las bases de datos no quedó fuera de tal situación. Los fabricantes de los SGBDs tradicionales se vieron en la necesidad de extender sus estructuras habituales para poder manipular datos XML, dando origen a los SGBDs extendidos (SGBDR_XML). Conjuntamente, surgieron los SGBDs XML nativos (NXDB), que adoptaron el tipo XML como estructura de datos intrínseca.

Este trabajo tiene como finalidad, presentar y explicar algunos aspectos significativos en cuanto a la elección del modo de almacenamiento de datos XML en el Sistema de Gestión de Bases de Datos Oracle 11g.

Palabras clave: Bases de Datos – XML

Contexto

Este artículo expone parte del trabajo realizado dentro de las actividades planificadas en el Proyecto de Investigación “XML: TÉCNICAS DE GESTIÓN E INTERCAMBIO DE DATOS” - 21/ E915, aprobado por el CICITCA en Octubre de 2011. De esta manera, contribuye al objetivo general de dicho proyecto: “Estudiar y experimentar la tecnología XML en distintos tipos de motores de bases de datos”.

Introducción

XML, al igual que el lenguaje de marcas de hipertexto HTML (Hyper Text Markup Language) sobre el que está basado World Wide Web, tiene sus raíces en la gestión de documentos y está derivado de un lenguaje para estructurar grandes documentos, conocido como lenguaje estándar generalizado de marcas SGML (Standard Generalized Markup Language) [1] [3]. Sin embargo, a diferencia de SGML y HTML, XML puede representar datos estructurados usados en aplicaciones de negocios.

Un archivo escrito en XML además de proporcionar información se describe asimismo, constituyéndose en un formato ideal para materializar el intercambio de datos entre aplicaciones. El receptor puede entender la información recibida y, por lo tanto, procesarla. Tiene el formato de un archivo de texto plano, lo que facilita enormemente la transferencia de información, logrando además independencia con respecto a diferentes plataformas.

La utilización masiva de esta nueva forma de representación de datos en muchos contextos, provocó la necesidad de gestionarlos [2]:

- Los SGBDs tradicionales extendieron sus capacidades para poder soportar datos XML [4][6][9]. En consecuencia, el lenguaje estándar utilizado por los motores de bases de datos relacionales, SQL, se amplió con el objetivo de manejar estructuras XML, dando lugar a la aparición del SQL/XML (ISO/IEC 9075-14:2006) [5].
- Surgió un nuevo tipo de motor de bases de datos, los SGBDs XML nativos. La estructura de datos subyacente en estos motores es precisamente XML. Además, proveen lenguajes XML específicos, tales como XPath y XQuery [2].

Este trabajo aborda específicamente aspectos relacionados al primer grupo, los SGBDR_XML. Particularmente se eligió Oracle 11g dado su posicionamiento aventajado en el mercado y al importante conjunto de tecnologías relacionadas al almacenamiento y recuperación de datos XML que incluye desde su versión 9iR2, llamado Oracle XML DB [7].

Oracle permite almacenar datos XML fuera de la base de datos, en un repositorio, y acceder a sus datos a través de técnicas basadas en el lenguaje XPath.

Asimismo, permite almacenar documentos XML dentro la base de datos, específicamente en tablas. Para ello provee un tipo de datos nativo llamado XMLType.

1. Modelos de almacenamiento XMLType

Este nuevo tipo de dato abstracto permite guardar datos en formato XML. Brindando la posibilidad de utilizar esquemas XML [10], XPath, XQuery [3], XSLT [11], indexación y particionamiento de documentos XML.

En las versiones Oracle Database 9i y 10g los documentos XML se almacenan como CLOB o como objetos. A partir de la versión Oracle Database 11g se agregó una nueva posibilidad, almacenarlos en formato binario (Binary XML), que de hecho es el formato por defecto a partir de la versión 11.2.0.2 [7][8].

a) *Almacenamiento no estructurado*

Los datos XMLType se almacenan como un CLOB (Character Large Object). Los documentos XML son almacenados preservando el documento original, inclusive los espacios en blanco. Así, se mantiene fidelidad de contenido o de documento. Además, Oracle brinda la posibilidad de asociar un esquema en esta forma de almacenamiento. Provee un desempeño muy bueno en las operaciones de inserción y recuperación de los documentos completos.

Sin duda es una opción que proporciona una gran flexibilidad, pero por otro lado, requiere una sobrecarga en el procesamiento cuando se necesita consultar su contenido, como por ejemplo cuando se usan las funciones XMLType.Extract y XMLType.ExistsNode. Evaluar estas funciones requiere construir el árbol XML DOM en memoria y sobre él evaluar las expresiones XPath. Además, toda operación de actualización debe realizarse a nivel de documento [7].

b) Almacenamiento estructurado

En este caso, los datos XMLType se almacenan como un conjunto de objetos. Los documentos XML mantienen fidelidad DOM (Documento Object Model). Esta alternativa también es mencionada como almacenamiento Objeto-Relacional (OR) [7].

Es común y conveniente almacenarlos con un esquema asociado, puesto que permite acelerar las consultas y las actualizaciones de granularidad fina. Esto provoca que las aplicaciones que lo utilizan deban ajustarse a un esquema de datos bien estructurado y rígido.

Por otro lado, presenta varias ventajas en comparación con el almacenamiento no estructurado. Se logra una optimización del manejo de memoria, se reducen los requerimientos de almacenamiento, y se pueden hacer actualizaciones a nivel de detalle.

Concluyendo, las mejoras de esta alternativa tiene su contrapartida, el incremento de la sobrecarga durante la inserción y la recuperación de los datos, además de la reducción de la flexibilidad en términos de estructura.

c) Almacenamiento binario

Esta nueva forma de almacenamiento provista por la versión 11g se introduce con la intención de compensar ventajas y desventajas de las dos alternativas anteriores. Así, provee una gran flexibilidad en cuanto a la estructura sin deteriorar el desempeño.

Concretamente, mantiene un buen rendimiento en la actualización, indexación y extracción de fragmentos; así como en las consultas puesto que parsea el documento antes de almacenarlo [7][13].

La Figura 1[13] muestra una comparación bastante ilustrativa entre las tres opciones de almacenamiento descriptas en Oracle 11g.

2. Elección de la estructura de almacenamiento

	CLOB	OR	Binary XML
Query	poor	excellent	good/excellent
DML	poor	good/excellent	excellent
document retrieval	excellent	good/excellent	excellent
schema flexibility	good	poor	excellent
document fidelity	excellent	poor	good/excellent

Figura 1: Comparación de los diferentes modelos de almacenamiento [13]

Cada modelo de almacenamiento posee un conjunto de características que lo hacen más o menos apropiado según el tipo de datos a manipular. Sin embargo, aunque la naturaleza de los datos es un aspecto fundamental a considerar, el uso que se les dará a esos datos también constituye un factor decisivo [12].

En cuanto a los datos, se pueden distinguir:

a) *Datos Estructurados*

Corresponde a aquellos datos que tienen una estructura regular y granularidad fina. Los datos contenidos en facturas de ventas y en cuentas de un banco son ejemplos de esta categoría.

b) *Datos Semiestructurados o No Estructurados*

Se caracterizan por tener una estructura menos regular, una granularidad más gruesa y con contenidos mezclados; tales como los datos contenidos en correos electrónicos, libros, currículos y advertencias. En general, los datos están organizados para ser usados por personas.

Por otra parte, como se comentó con anterioridad, las aplicaciones que usan los datos también pueden ser de diferente naturaleza:

c) *Aplicaciones centradas en datos*

Estas aplicaciones necesitan conocer la estructura de los datos. En general, los datos son altamente estructurados. Luego, la aplicación puede tomar ventaja en este sentido. Es común que los datos se ajustan a un esquema XML.

d) *Aplicaciones centrada en documentos*

En este caso, es común que las aplicaciones necesiten mantener una copia exacta de los documentos, como sucede por ejemplo en el ámbito judicial, médico, etc.

Luego, bajo la hipótesis de contar con datos estructurados, por un lado cabe la situación que las aplicaciones no necesiten conocer esa estructura, y por otro, que sí lo demanden. En el primer caso, el modelo de almacenamiento como CLOB será la alternativa más aconsejable, mientras que en el segundo, el modelo apropiado sería el binario.

3. Ejemplo

En cuanto a las experiencias realizadas en el uso de datos XML en Oracle, éstas tienen relación con algunas tareas realizadas desde el proyecto marco. Concretamente se tomó como escenario la situación presentada en el Departamento de Informática de la Facultad de CEFN de la UNSJ, a raíz del proceso de acreditación de sus carreras.

El primer objetivo del ensayo realizado fue capturar datos relacionales del Sistema SIU Guaraní de la FCEFN de la UNSJ, para posteriormente almacenarlos como datos XML, ofreciendo así una de las cualidades de este tipo de estructura, la conveniencia para el intercambio de datos.

En esta sección se expone una porción exigua del trabajo mencionado en los párrafos anteriores, ya que el objetivo, en este caso, es sólo ilustrar algunos aspectos relacionados al almacenamiento de datos XML en Oracle 11g, finalidad de este artículo.

- Se generó una base de datos llamada SIU_XML, la cual contendrá algunos datos SIU Guaraní versión 2.00.0, a Junio de 2004, por supuesto almacenados como datos XML.
- Se creó la tabla EXAMEN_MATERIA, ilustrada en la Figura 2. Como se puede visualizar en este caso el modo de almacenamiento optado fue el binario [8]. Dicha elección se debió a que los datos a almacenar son estructurados, además que el uso que se les dio posteriormente necesita conocer su estructura, hacer consultas sobre ellos, y tener cierta flexibilidad.

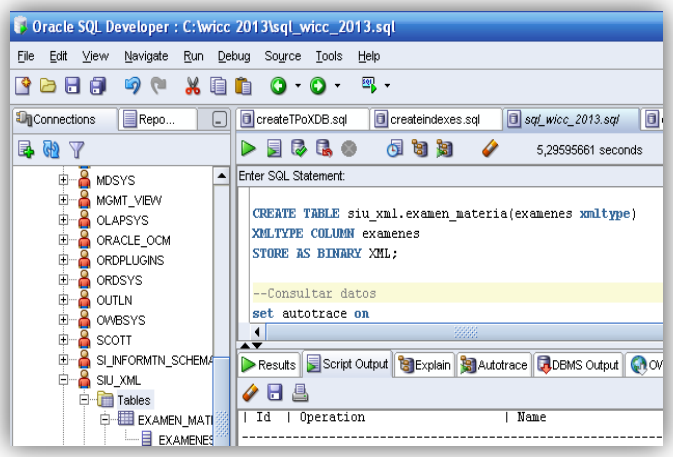

Figura 2: Creación de la tabla EXAMEN_MATERIA (esquema SIU_XML)

- Posteriormente se insertaron las tuplas en la tabla EXAMEN_MATERIA. Los datos agregados corresponen a los Exámenes de la carrera Licenciatura en Ciencias la de Información, período académico 2000-2002. Uno de los documentos XML insertados se muestra en la Figura 3.

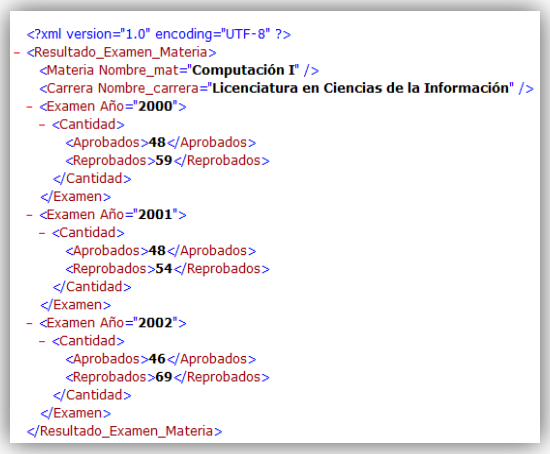

Figura 3: Ejemplo de Documento XML insertado en la tabla EXAMEN_MATERIA

Cabe mencionar que se generaron índices específicos[1] que permitieron optimizar la manipulación de los datos XML[12][7][8]. En este caso el tipo de índice creado fue XMLIndex, mostrado en la Figura 4.

[1] Por razones de espacio, en este trabajo no se abordó el tema de índices. Sólo se presenta una breve acotación dada su gran y provechosa utilización.

```
create index idx_examen ON
siu_xml.EXAMEN_MATERIA(EXAMENES) INDEXTYPE IS
XDB.XMLINDEX PARAMETERS
('PATHS (INCLUDE
(/Resultado_Examen_Materia/Nombre_mat))');
```

Figura 4: Creación del índice

Específicamente el índice se construyó sobre el elemento Nombre_mat. Si en la sentencia de creación del índice no se especifica el parámetro PATHS, se crea un índice para cada uno de los elementos del documento XML. Es necesario ser cuidadoso y analizar el uso que se le dará a posteriori a esos datos para decidir sobre qué elementos conviene hacer la indexación, ya que por supuesto tiene asociado un costo, fundamentalmente en cuanto al espacio de almacenamiento.

La Figura 5 muestra una consulta planteada que emplea la función extractValue para recuperar el nombre de las materias.

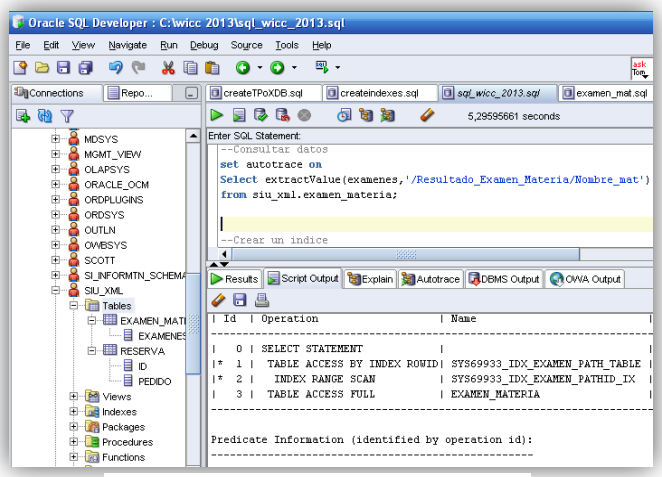

Figura 5: Consulta documento XML

A continuación, en la Figura 6 se puede observar el resultado de dicha consulta.

```
AutoTrace Enabled
EXTRACTVALUE(EXAMENES,'/RESULTADO_EXAMEN_MATERIA/NOMBRE_MAT')
--------------------------------------------------------------------------------
Sistemas de Datos I
Análisis Comparativo de Lenguajes
Modelos y Simulación
Introducción a los Sistemas Digitales
Computación III
Investigación Operativa
Integración Cultural I
Sistemas Administrativos III
Integración Cultural II
Integración Cultural III
Base de Datos con Orientación a Objetos
Optativa I  (LIC)
Optativa II (LIC)
Razonamiento Temporal y Programación Lógica con Restricciones
Informática Industrial
Requisito de Idioma
Sistemas de Datos II
Sistemas de Información I
Sistemas de Información II
Informática Legal
Sistemas de Información III
Computación IV
Matemática Básica y Lógica
Introducción a los Sistemas de Información
```

Figura 6: Resultado de la consulta efectuada

Como la consulta de la Figura 5 selecciona los nombres de materias, elemento por el cual se generó el índice, Oracle lo utiliza para acelerar el tiempo de respuesta. La Figura 7 muestra el plan de ejecución que permite observar dicha situación.

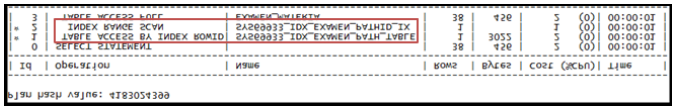

Figura 7: Plan de Ejecución

Líneas de Investigación y Desarrollo

La línea de investigación es la tecnología XML en el area de bases de datos, tanto desde el aspecto teórico como también del práctico. En tal sentido se ha trabajado con herramientas que soportan estas tecnologías de maneras muy diferentes:

Editores XML:

- Altova XMLSpy. Se usó una versión de prueba durante 30 días. Sitio de descarga es http://www.altova.com.
- XML Copy Editor: Licencia GPL. Sitio de descarga: http://xml-copy-editor.sourceforge.net/.

Sistemas de gestión de bases de datos que manipulan datos XML:

- El SGBD Oracle 11 g R1 con licencia OTN. Sitio de descarga http://www.oracle.com.ar
- El SGBD SQL Server 2012
- El SGBD XML nativo eXist 1.4.0, con licencia GPL. Sitio de descarga http://www.exist-db.org/download.html

Resultados y Objetivos

A continuación se detallan los resultados obtenidos hasta el momento:

- Se ha estudiado y experimentado la mayoría de los temas que estaban planificados. También se ha investigado el motor SQL Server 2012, aspecto no contemplado inicialmente en el proyecto.
- Se dictó el curso: "Base de Datos Relacional-XML" dirigido a empleados de cómputos de

OSSE (Obras Sanitarias Sociedad del Estado, San Juan), de 70 horas presenciales.

- Se dirigió un trabajo final de la carrera Licenciatura en Ciencias de la Información, "Tecnología XML y Bases de Datos: Un ejemplo de aplicación con datos del SIU". C. Vera y L. Romera.
- Se está asesorando un trabajo final de la carrera LSI, "XML y Bases de datos: Un caso de aplicación". M. Avendaño.

Formación de Recursos Humanos

Se propusieron y aceptaron dos Becas de Investigación del CICITCA:

- Beca Interna de Investigación en la CATEGORÍA INICIACIÓN, convocatoria 2012, en el marco de la Ordenanza Nro. 10/05-CS: Licenciada Cristina Vera.
- Beca Interna de Investigación en la CATEGORÍA ALUMNO AVANZADO, convocatoria 2012, en el marco de la Ordenanza Nro. 10/05-CS: Mauro Avendaño, alumno de la carrera LCI.

Referencias

1. Benz, Durant. "XML Programming Bible". Wiley Publishing Inc.
2. Chaudhri, Rashid, Zicari. "XML Data Management: Native XML and XML-Enabled Database Systems". Addison Wesley.
3. Evjenet. "Professional XML". Wrox Press 2007.
4. Elmasri, Navathe. FUNDAMENTALS OF DATABASE SYSTEMS. Cuarta Edición. Adisson Wesley.
5. ISO/IEC 9075-14:2003 Information technology – Database languages – SQL – Part 14: XML-Related Specifications, (SQL/XML)
6. Molina, Ullman, Widom. DATABASES SYSTEMS: THE COMPLETE BOOK. Prentice Hall.
7. Oracle. "Oracle® XML DB Developer's Guide 11*g*".
8. Oracle. "SQL Reference 11g".
9. Silberschatz, Korth, Sudarshan. FUNDAMENTOS DE BASES DE DATOS, Quinta Edición, Mc Graw Hill.
10. w3c. "XML Schema Part 1: Structures W3C Recommendation 2 May 2001". Fecha de consulta: 10 de Marzo de 2010. URL://www.w3.org/TR/2001/REC-xmlschema-1-20010502.
11. w3c. "XSL Transformations (XSLT) " URL: http://www.w3.org/TR/xslt.html
12. Williams. "Performance of Relational Databases versus Native XML Databases"
13. Zhang, Agarwal. "Binary XML Storage and Query Processing in Oracle 11g".