Lenguajes de consulta para xml: un análisis comparativo

Por **Adelaida Delgado Domínguez** y **Ricardo Baeza-Yates**

Adelaida Delgado Domínguez, licenciada en informática, es profesora de "Tecnologías de la información" en la Universitat de les Illes Balears. Desde 1992 ha participado en la realización de diversos cd-roms multimedia/hipermedia, en la creación de webs de servicios de información y actualmente colabora en proyectos de recuperación de información y bibliotecas digitales.

Resumen: *Un lenguaje de consulta para xml debería ser lo suficientemente flexible para cubrir el espectro de fuentes de información que xml permite etiquetar, incluyendo bases de datos y documentos distribuidos a través del web. En este artículo se presenta un análisis comparativo de diversos lenguajes de consulta que han surgido para xml. Se estudian conjuntos de características deseables tanto desde la perspectiva de la comunidad de datos semiestructurados (la cual pone el énfasis principalmente en los grandes almacenes de datos, la integración de fuentes heterogéneas y la transformación de datos en formatos comunes de intercambio), como desde la perspectiva de la comunidad de recuperación de información (que pone el énfasis principalmente en las búsquedas de texto, la manipulación de conjuntos de resultados, las relaciones de inclusión, las distancias y la ordenación por relevancia de los documentos resultado).*

Palabras clave: *Xml, Lenguajes de consulta, Recuperación de información, Datos semiestructurados.*

Title: Query languages for xml: a comparative analysis

Abstract: *A query language for xml should be flexible enough to cover the whole range of information sources that can be labelled by xml, including databases and web documents. In this article we present a comparative analysis of several query languages that have been created for xml. We study sets of desirable traits both from the point of view of the semi-structured data community (that emphasise especially very large databases, the integration of heterogeneous sources and data transformation into common interchange formats), and from the point of view of the information retrieval community (that emphasise especially full-text searches, manipulation of the sets of results, inclusion relationships, distances, and ranking of the result documents).*

Keywords: *Xml, Query languages, Information retrieval, Semi-structured data.*

***Delgado Domínguez, Adelaida; Baeza-Yates, Ricardo**. "Lenguajes de consulta para xml: un análisis comparativo". En: El profesional de la información, 2002, marzo-abril, v. 11, n. 2, pp. 84-90.*

Ricardo Baeza-Yates, catedrático en el Departamento de Ciencias de la Computación, Universidad de Chile, Ph. D. en Computer Science (University of Waterloo, Canadá, 1989). Co-autor de "Modern information retrieval" (Addison-Wesley, 1999) entre otros libros. Actualmente, además de otros cargos, es presidente del Clei (Centro Latinoamericano de Estudios en Informática) y coordinador internacional del Subprograma de informática y electrónica de Cyted (Programa Iberoamericano de Ciencia y Tecnología para el Desarrollo).

Introducción

Desde la aparición de xml se ha debatido ampliamente cuál debe ser la función de un lenguaje de consulta (ver las diversas posturas presentadas en el *Workshop QL'98* del *W3C* sobre lenguajes de consulta [2] y en particular las cuestiones planteadas en [5]), existiendo varias propuestas de desideratas como por ejemplo [30] y [33], que el *W3C* intenta reflejar en el borrador de su documento de requerimientos (modelo de datos, álgebra y lenguaje) para la consulta de xml [17].

Tradicionalmente las búsquedas de texto se han venido aplicando a documentos como un todo mientras que la consulta de datos se ha aplicado a registros compuestos por varios campos. Dado que xml representa datos semiestructurados es importante que un lenguaje de consulta para xml pueda seleccionar datos de la parte estructurada del documento xml, buscar información en el texto y además permitir consultas mixtas de contenido y estructura. En cuanto a los resultados, tradicionalmente las búsquedas de texto devuelven una lista de documentos con cierta información acerca de ellos y ordenados según algún criterio de relevancia (*ranking*), mientras que una consulta de datos arroja generalmente un determinado campo, registro o conjunto de éstos, que pueden pasar por un proceso de cálculo, transformación, combinación, etc.

Artículo recibido el 03-12-01
Aceptación definitiva: 04-02-02

En síntesis, la comunidad de bases de datos pone el énfasis principalmente en los grandes almacenes de datos, la integración de fuentes heterogéneas y la transformación de datos en formatos comunes de intercambio, mientras que en el contexto de la recuperación de información se pone el énfasis principalmente en las búsquedas de texto, la manipulación de conjuntos de resultados, las relaciones de inclusión, las distancias y la ordenación por relevancia de los documentos obtenidos.

«Tradicionalmente las búsquedas de texto se han venido aplicando a documentos como un todo, mientras que la consulta de datos se ha aplicado a registros compuestos por varios campos»

En la siguiente sección de este artículo se presentarán las características deseables en un lenguaje de consulta, tanto desde la perspectiva de datos semiestructurados como desde el punto de vista de recuperación de información. Con posterioridad se presentan los lenguajes de consulta para xml estudiados, y por último, en el punto 4 se hará un análisis comparativo de diferentes lenguajes de consulta diseñados según las características enunciadas en la sección 2.

2. Características deseables en un lenguaje de consulta

2. 1. Desde la perspectiva de datos semiestructurados. Para atender las necesidades de esta comunidad, un lenguaje de consulta para xml debería poseer las cualidades comunes de los lenguajes de consulta de datos semiestructurados, tanto las puramente relacionales (operaciones de selección, filtrado, reducción, reestructuración y combinación) como aquellas similares a las de los lenguajes de consulta de bases de datos orientados a objeto, tales como la navegación y el anidamiento. A continuación se explican con mayor detalle las principales particularidades deseables para la consulta de datos:

a. Operación de selección: elige un documento o elemento basándose en el contenido, estructura o atributos que satisfagan una condición específica. Estas consultas constan generalmente de 3 partes o cláusulas:

—Patrón: equipara elementos anidados en el documento de entrada y les asocia variables. Equivale a "from" en sql, donde los patrones son tablas, vistas o alias. En el caso de xml, un patrón es un elemento xml en el cual algunos de los ítems (nombres de etiqueta, contenido, nombres de atributo o valores de atributo) son eventualmente sustituidos por variables.

—Filtro: testea que las variables asociadas cumplan las condiciones establecidas ("where" en sql). El significado de los patrones y filtros en los lenguajes para datos semiestructurados es una relación (producto cartesiano de todas las variables asignadas que satisfacen los patrones y filtros) que tiene una estructura plana y sin orden.

—Constructor: especifica el resultado en términos de las variables asociadas, es decir qué formato ha de tener la respuesta. Se corresponde a la cláusula "select" de sql. Las construcciones deberían poder incluir consultas anidadas, así como poder crear nuevos elementos.

b. Operación de filtrado: extrae determinados elementos de los documentos conservando la jerarquía y secuencia.

c. Operación de reducción: proyecta como salida la poda de los elementos especificados en la selección que satisfacen las condiciones, en vez de devolver un subárbol con todos los elementos y atributos.

d. Acción de reestructuración, como por ejemplo la agrupación de datos relacionados y la ordenación.

e. Operación de combinación de datos de diferentes porciones de documentos (correspondiente al "join" relacional) o combinación de diferentes partes del mismo documento ("semi-join").

f. Uso de funciones de agregación.

g. Utilización de la cuantificación existencial y universal.

h. Operaciones de inserción, borrado y modificación.

Además de estas características genéricas, para consultar documentos en xml sería también deseable disponer de:

—Variables etiqueta o expresiones de camino para permitir peticiones sin conocimiento preciso de la estructura del documento y acceso a datos anidados de forma arbitraria. El lenguaje de consulta debe poder usarse aun cuando no se conozca un esquema (dtd o xml schema) a priori.

—Operadores de navegación que simplifiquen el manejo de datos con referencias [atributos id, idref(s)].

—Manejo de tipos de datos, en particular los del xml schema [11].

2.2. Desde la perspectiva de recuperación de información. Los documentos xml pueden considerarse como una colección de documentos de texto con eti-

quetas adicionales que guardan cierta relación. Las técnicas de recuperación tradicionales han de adaptarse a la búsqueda por estructura y contenido de tales documentos (ver [9], [32] y [40] para un análisis de diferentes enfoques sobre esta cuestión) y hay que plantearse cómo ordenar por relevancia los resultados teniendo en cuenta que en una consulta xml las unidades de información devueltas pueden ser tanto documentos xml enteros, un fragmento de documento (por ejemplo un elemento y su contenido), así como una combinación de documentos o de fragmentos.

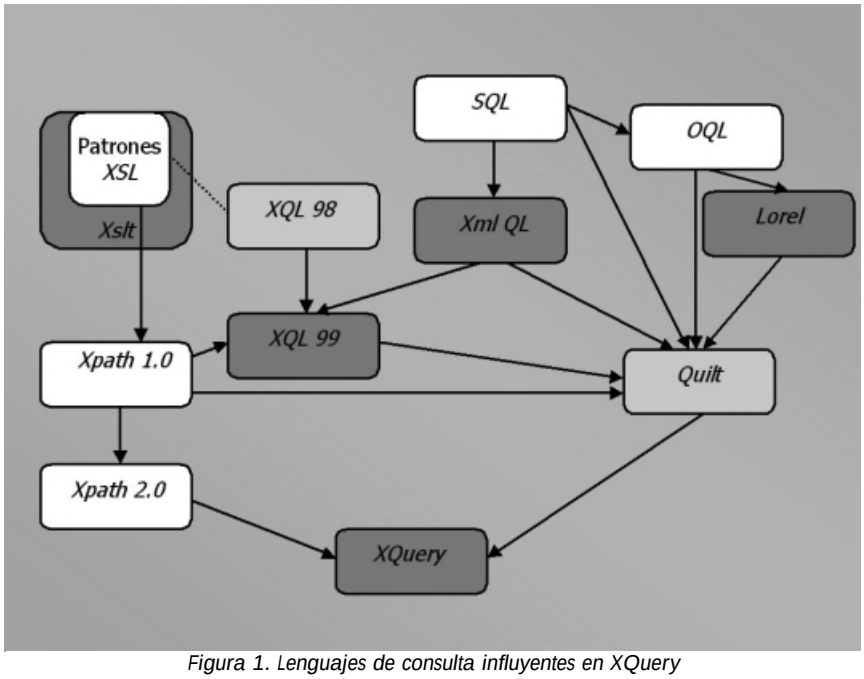

Figura 1. Lenguajes de consulta influyentes en XQuery

Analizaremos los diferentes tipos de búsquedas formuladas normalmente a los sistemas de recuperación de información clásicos [10]:

a. Basadas en palabras-clave. Se refiere a consultas de contenido, tanto para buscar datos sin estructura como para datos estructurados pero en los que la estructura es desconocida por el usuario, o consultar múltiples documentos xml con la misma ontología pero diferentes dtds. Consultas básicas:

—De una sola palabra. La búsqueda de texto puede ser tan simple como determinar si una palabra concreta está o no presente. En este caso el resultado serían los documentos o elementos que la contienen, ordenados arbitrariamente. Los lenguajes de consulta para xml generalmente suavizan el problema del desconocimiento de la estructura exacta mediante la utilización de expresiones regulares de camino, si bien en general requieren un conocimiento parcial de la misma del que un usuario puede carecer totalmente.

«Un lenguaje de consulta para xml debería poseer las cualidades comunes de los lenguajes de consulta de datos semiestructurados, tanto las relacionales como aquellas de los lenguajes de consulta de bases de datos orientados a objetos»

—Consulta difusa. Consta de una o más palabras, siendo el resultado un conjunto de documentos o elementos que contienen al menos una de ellas. El material recuperado será más o menos relevante a la consulta y se ordenará (*ranking*) por similitud, aplicando una función generalmente ligada a la frecuencia de un término (número de veces que una palabra aparece en un documento o elemento) y a la frecuencia inversa de documento o elemento (número de documentos o elementos en los cuales aparece el término).

—Consultas de contexto. Implican distancias textuales, restricciones que se pueden imponer respecto a las posiciones de los términos, como que formen una secuencia (frase) o que estén próximos. La proximidad textual también es uno de los factores a considerar para efectuar la ordenación de resultados por relevancia.

—Booleanas: constan de una combinación de varias palabras ligadas con operadores booleanos; los documentos o elementos recuperados dependerán de los operadores utilizados.

b. Consultas de concordancia de patrones, para recuperar fragmentos de texto que concuerden con el patrón especificado (cadenas, prefijos, sufijos, subcadenas o en general expresiones regulares).

c. Consultas estructurales. Básicamente podemos resaltar:

—Relaciones de inclusión: se refieren a la selección de elementos que incluyan o estén incluidos en otros (por ejemplo seleccionar capítulos que contengan una figura, o seleccionar figuras comprendidas en una sección). Este tipo de relación abarca también la selección de ancestros o descendientes directos de un elemento. También consideraremos la inclusión posicional: seleccionar el i-ésimo elemento (o un rango de elementos) de los incluidos por un elemento de otro conjunto (por ejemplo seleccionar el tercer párrafo de los capítulos en donde se encuentra una figura).

—Relaciones de distancia (medida en función del número de arcos que hay que atravesar desde un nodo

a otro) para poder expresar un orden de precedencia o una proximidad estructural, la cual se podría tener en cuenta para calcular la relevancia.

Los tipos de consulta concretos que se puedan formular dependerán en parte del modelo de recuperación que el sistema adopte (ver [29] para un análisis de técnicas de indización para documentos xml y modelos de recuperación, y [8] para un estudio de modelos de datos en cuanto a la manipulación de documentos estructurados a nivel de usuario, conceptual y físico).

Además de permitir algunos de los tipos de consulta mencionados anteriormente, un lenguaje para xml también sería deseable que posibilitara:

—La asignación de pesos a los términos de la consulta.

—La utilización de metadatos (rdf).

—El soporte de *XLink* y *XPointer*. Dado que generalmente las consultas están orientadas al web deben poder atravesar las referencias internas y externas para proporcionar documentos relacionados.

—La manipulación de conjuntos: unión, diferencia e intersección de resultados.

3. Los lenguajes de consulta para xml

En febrero de 2001 el *W3C* publicó el primer borrador de *XQuery*, el que se pretende que constituya el lenguaje de consulta estándar para xml. Pero hasta llegar a este proceso de estandarización actual han surgido otros, de los cuales centraremos nuestro estudio sólo en aquellos que han tenido influencia en el diseño de *XQuery* [15]: *Lorel* [1, 6] (representativo de los lenguajes para datos semiestructurados), *xml-QL* [22, 23, 24] (primer lenguaje que utiliza sintaxis xml), *Xslt* [20] (lenguaje de transformación de árboles xml) y *XQL 99* [34] (aporta la navegación por documentos jerárquicos). El análisis del lenguaje *Quilt* [18, 35] resulta redundante ya que *XQuery* deriva directamente de él y su estudio engloba todas sus características. La figura 1 muestra las relaciones entre distintos lenguajes, apareciendo rellenos los recuadros correspondientes a los lenguajes de consulta para xml, y resaltando en intensidad los analizados en este trabajo.

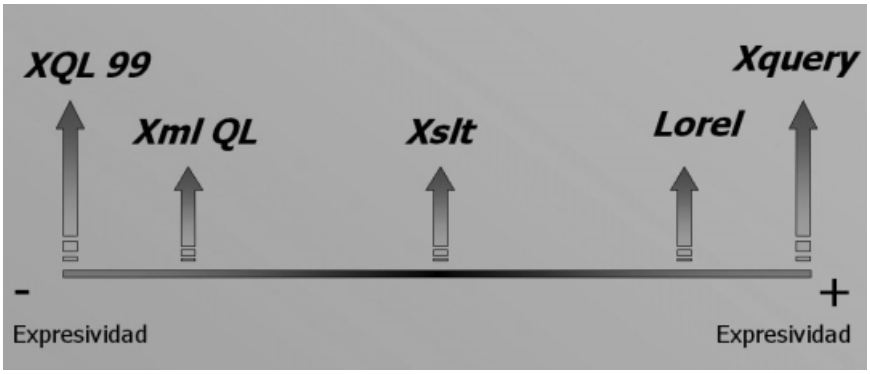

Figura 2. Ordenación de los lenguajes de consulta estudiados según su expresividad ligada a la consulta de datos

«Los lenguajes de consulta para xml suavizan el problema del desconocimiento de la estructura exacta mediante la utilización de expresiones regulares de camino»

Existen otros análisis de consulta como [28 y 34] que comparan *XQL* y *xml-QL*; [25] que analiza *xml-QL*, *YaTL*, *Lorel* y *XQL* a través de 10 ejemplos de consultas[1], y [12] que analiza *Lorel*, *xml-QL*, *xml-GL*, *Xslt*, *XQL* y *XQuery*, si bien todos estos estudios proceden más bien de la perspectiva de la comunidad de bases de datos y se deja de lado la problemática inherente a la perspectiva de recuperación de información. Esta cuestión sin embargo es analizada en [29], aunque éste no presenta un examen de los diferentes lenguajes de consulta.

4. Análisis comparativo

El modelo de datos utilizado por un lenguaje de consulta determina su entrada/salida. *Lorel* y *xml-QL* poseen uno propio (grafo); *XQL* adopta el implícito de xml (árbol); *Xslt* utiliza el mismo que *XPath 1.0* (árbol), mientras que *XQuery* se basa en el modelo de *XPath 2.0* (secuencia de nodos, cada uno de los cuales a su vez puede contener secuencias anidadas de nodos) y todavía se encuentra en desarrollo.

En cualquier lenguaje las consultas se pueden aplicar tanto a un solo documento como a una colección de ellos. Como información de salida generalmente se construye un nuevo documento xml (al cual se le puede volver a aplicar una búsqueda) a partir de los fragmentos xml seleccionados por la consulta. Tan sólo *Xslt*, *XQL* y *XQuery* pueden producir como resultado múltiples documentos, a los que se les puede envolver en un nodo raíz común para crear un documento xml bien formado. En el caso de *XQuery* no es preciso ya que su modelo de datos permite que el resultado sea una secuencia de nodos, sin que tengan que tener una raíz común.

Desde el punto de vista de consulta de datos, todos los lenguajes estudiados poseen cláusulas de selección con patrón, filtro y constructor, excepto *XQL* que no posee esta última y no permite la creación de nuevos elementos. Todos toleran consultas

anidadas y también expresiones de camino más o menos complejas.

XQuery es el más expresivo de todos los lenguajes si bien carece de las operaciones de reducción y de manipulación (inserción borrado y modificación). *Lorel*, al provenir de la comunidad de bases de datos, reúne la mayoría de características estudiadas (incluidas las operaciones de manipulación), si bien no presenta las operaciones de reducción y filtrado, no soportando todos los tipos de datos de xml-schema.

Por su parte, *Xslt* no posee operaciones de reestructuración de resultados (a excepción de la ordenación de nodos), de cuantificación universal y no permite el manejo de tipos de datos (si bien esto está en estudio para la próxima versión).

Xml-QL es menos expresivo que los anteriores al carecer de operaciones de filtrado, reducción, agrupación de resultados, cuantificación universal, manejo de tipos de datos y de las operaciones de inserción, borrado y modificación.

XQL 99 añadió a *XQL 98* la importante característica de realizar uniones inter-documentos. Sin embargo todavía está falto de otras como la posibilidad de ordenación de resultados, la creación de nuevos elementos y el manejo de tipos de datos, además de no presentar la posibilidad de utilizar variables etiqueta. Sin embargo *XQL* es más sencillo que *xml-QL* para consultas que requieran preservar la jerarquía, secuencia y anidamiento original, y por otra parte, algunas de las carencias de *XQL* mencionadas son extensiones del lenguaje propuestas en [34] para asemejarlo más a los lenguajes de datos semi-estructurados. En la figura 2 se puede ver la ordenación de los lenguajes analizados según su expresividad.

En cuanto a las características ligadas a recuperación de información, los cinco lenguajes estudiados permiten realizar consultas teniendo en cuenta la inclusión estructural (indicada mediante expresiones de camino) y poseen índices posicionales que pueden utilizarse para consultas de inclusión posicional y de orden estructural. Algunos de los lenguajes poseen operadores específicos para expresar el orden estructural: *preceding, preceding-siblings, following, following-siblings*, en *Xslt*, y operadores *before* y *after* en *XQL 99* y *XQuery*. Todos permiten también operaciones sobre conjuntos, si bien *XQL* no puede expresar transformaciones de los resultados de la consulta.

A pesar de estas características, los lenguajes presentan serias limitaciones para la recuperación de información ya que incluso la posibilidad más sencilla de búsqueda de una palabra en el texto se ha de realizar utilizando las expresiones regulares de camino, sin que exista una función u operador específico para ello. Tampoco permiten la búsqueda por similitud y por tanto ninguno ordena los resultados por relevancia, y también carecen de operadores de adyacencia, a excepción de ";" (inmediatamente precede) de *XQL 99* para proximidad estructural.

En cuanto a la búsqueda de patrones, *Lorel* y *xml-QL* son los únicos que incorporan el operador "like". *Xql* y *Xslt* proporcionan muchos operadores y funciones de texto así como la posibilidad de comparar la posición de diferentes palabras. *XQuery* hereda estas propiedades de *XQL* soportando además el uso de funciones externas. Sin embargo ninguno permite puntos de concordancia (*match points*) para presentar los puntos concretos del documento donde aparece el patrón buscado y permitir además visualizarlo en su contexto para asignarles una semántica correcta. Este aspecto es muy importante para investigadores del área de filología y lingüística y sí es considerado en otro lenguaje de consulta para documentos xml denominado *Tequila-TX* [7].

«Todos los lenguajes estudiados poseen cláusulas de selección con patrón, filtro y constructor, excepto *XQL*»

Tampoco ninguno de estos lenguajes soporta rdf, si bien en [14] y [13] se estudia cómo generar descripciones tabulares a partir de rdf usando *Xslt*. La idea es aplicable a la presentación de resultados de una consulta en la web, donde las descripciones de los recursos proporcionadas ayudarían al usuario a decidir su relevancia particular sin necesidad de descargarlos. En cuanto a consultas que puedan seguir enlaces, sí pueden ser expresadas en *XQL 99* usando el método *document()*.

Dada la poca adecuación en general de los lenguajes estudiados para la recuperación de información, han surgido otros, algunos basados en ellos, que permiten expresar el tipo de consultas requeridas por esta comunidad. La simple búsqueda por palabra clave, sin conocimiento alguno de la estructura de los documentos, es posible de manera muy sencilla con lenguajes como el de [26] (ampliación de *xml-QL*) que incorpora el predicado *contains* para testear la existencia de un término en un elemento xml, como una alternativa más práctica al uso de expresiones regulares de camino.

La búsqueda por similitud la integran lenguajes como *XXL* [39] y *Elixir* [19] (derivados de *xml-QL*), o *Xirql* [27] y *ApproXQL* [37] (procedente de *XQL*). Tan sólo uno de ellos, *Xirql*, permite la asignación de pesos a los términos de la consulta. Está en estudio integrar las búsquedas por similitud en *Lorel*, lo cual implica-

ría combinar la presentación de resultados basada en conjuntos con la ordenación por relevancia.

Desde que se celebró el *workshop* sobre lenguajes de consulta *QL'98*, la *Library of Congress* de los EUA ha estado representando y defendiendo la perspectiva de la comunidad de biblioteconomía en la definición de un lenguaje de consulta para la web, aportando por un lado una lista de las capacidades que debe proporcionar un protocolo de recuperación de información [21] —capacidades que son soportadas por el Z39.50—, y por otro una lista de escenarios de uso para *XQuery* [4] mucho más extensiva que la ofrecida por el *W3C* [16], así como una lista de operadores y funciones de texto [3] asociados a los correspondientes escenarios de uso.

Los operadores propuestos permitirían expresar orden en los términos de consulta (*order*) —de momento solucionado con *before* y *after* de *XQuery*—, proximidad (*within*), distancia (*window*), unidades lingüísticas que aparezcan un número determinado de veces o menos (*atleast*) o más (*atmost*), relevancia en función de algún criterio como pesos de etiquetas o atributos, frecuencia, proximidad, tamaño del documento o registro (*relevant*)*, expresiones regulares (*like*)*, y una negación moderada para buscar una unidad lingüística excepto si aparece en un contexto específico (*ignore*).

Por el momento, el grupo de trabajo sobre consultas XML, del *W3C*, se plantea si sería necesario definir una función para la búsqueda de texto libre (*issue 37* de la lista de funciones y operadores de *XQuery* [31]) y si se debería permitir consultar por palabras que estuvieran próximas a otras (*issue 94* de la lista de funciones y operadores de *XQuery* [31]). Tales cuestiones están todavía pendientes de cerrar y desde la lista de distribución *www-ql@w3.org* se hace una llamada a participar en dicho grupo de trabajo a las organizaciones interesadas en el tema de procesamiento de consultas de texto completo en el contexto de xml.

También desde esa lista se recomienda tener en cuenta el estándar *ISO/IEC 13249-2:2000 "Information technology (database languages) SQL multimedia and application packages, Part 2: full-text"* que, aunque no es específico para xml, define una interfaz estandarizada para búsquedas de texto completo en general y ha sido implementada por conocidos vendedores de bases de datos como *Oracle* e *IBM*.

Nota

1. En: **Sahuguet**, **A**. *Xslt as a query language for xml*, se puede ver cómo se resuelven estos mismos ejemplos con *Xslt.*
http://db.cis.upenn.edu/XSLT/

Bibliografía

1. *Lore* (sistema de gestión de bases de datos para xml)
http://www-db.stanford.edu/lore/

2. *QL'98, the query languages workshop*. 1998, W3C: Boston, Massachusetts.
http://www.w3.org/TandS/QL/QL98/pp.html

3. *Xml query language operators and functions recommended by the Library of Congress*. Consultado en: 22-01-02.
http://www.loc.gov/crsinfo/xml/lc_core.html

4. *Xml query use cases from the Library of Congress*. 2001. Consultado en: 22-01-02.
http://lcweb.loc.gov/crsinfo/xml/lc_usecases.html

5. **Aberer**, **K**.; **Fankhauser**, **P**.; **Neuhold**, **E**. "Xml is the data model, what is the function of the query language?". En: *QL'98, the query languages workshop*. 1998.
http://www.w3.org/TandS/QL/QL98/pp/gmd.html

6. **Abiteboul**, **S**.; **Quass**, **D**.; **McHugh**, **J**.; **Widom**, **J**.; **Wiener**, **J**. **L**. "The Lorel query language for semi-structured data". En: *International journal on digital libraries*, v. 1, n. 1, pp. 68-88.
http://www-db.stanford.edu/pub/papers/lorel96.ps

7. **Albano**, **A**.; **Colazzo**, **D**.; **Ghelli**, **G**.; **Manghi**, **P**.; **Sartiani**, **C**.; **Lini**, **L**.; **Paoli**, **M**. "A text retrieval query language for xml documents". En: *Jasis*, en prensa.

http://www.cribecu.sns.it/analisi_testuale/settore_informatico/attivita/TequylaTX.pdf

8. **Arnold-Moore**, **T**.; **Sacks-Davis**, **R**. "Models for structured document database systems". En: *Markup technologies'98*. 1998.
http://www.mds.rmit.edu.au/~tja/papers/mt98.html

9. **Baeza-Yates**, **R**.; **Navarro**, **G**. "Integrating contents and structure in text retrieval". En: *ACM Sigmod record*, v. 25, n. 1, pp. 67-69.
ftp://sunsite.dcc.uchile.cl/pub/users/gnavarro/sigmod96.ps.gz

10. **Baeza-Yates**, **R**.; **Navarro**, **G**. "Query languages". En: *Modern information retrieval*. **Baeza-Yates**, **R**.; **Ribeiro-Neto**, **B**., 1999, pp. 99-116.

11. **Biron**, **P**. **V**.; **Malhotra**, **A**. *Xml schema part 2: datatypes*. W3C, 2001. W3C recommendation.
http://www.w3.org/TR/xmlschema-2/

12. **Bonifati**, **A**.; **Lee**, **D**. "Technical survey of xml schema and query languages". 2001.
http://www.cobase.cs.ucla.edu/tech-docs/dongwon/vldbjCom.pdf

13. **Cawsey**, **A**. "Presenting tailored resource descriptions: will Xslt do the job?". En: *9th International conference on the world wide web*. 2000.
http://www.cee.hw.ac.uk/~alison/www9/paper.html

14. **Cawsey**, **A**.; **Bental**, **D**.; **Eddy**, **B**.; **McAndrew**, **P**. "Generating resource descriptions from metadata to support relevance assessments in retrieval". En: *Riao 2000 content-based multimedia information access*. 2000.
http://www.cee.hw.ac.uk/~alison/publications/riao.doc

15. **Chamberlin**, **D**.; **Clark**, **J**.; **Florescu**, **D**.; **Robie**, **J**.; **Siméon**, **J**.; **Stefanescu**, **M**. *XQuery 1.0: an xml query language*. W3C, 2001. Working draft.
http://www.w3.org/TR/xquery/

16. **Chamberlin**, **D**.; **Fankhauser**, **P**.; **Marchiori**, **M**.; **Robie**, **J**. *Use case "text": full-text search*. W3C, 2001. W3C Working draft.
http://www.w3.org/TR/xmlquery-use-cases#text

17. **Chamberlin**, **D**.; **Fankhauser**, **P**.; **Marchiori**, **M**.; **Robie**, **J**. *Xml query requirements*. W3C, 2001. W3C Working draft.
http://www.w3.org/TR/xmlquery-req

18. **Chamberlin**, **D**.; **Robie**, **J**.; **Florescu**, **D**. *Quilt: an xml query language for heterogeneous data sources*. Consultado en: 17-04-01.
http://www.almaden.ibm.com/cs/people/chamberlin/quilt_lncs.pdf

19. **Chinenyanga**, **T**.; **Kushmerick**, **N**. *Elixir: an expressive and efficient language for xml information retrieval*. Consultado en: 20-04-01.
http://www.smi.ucd.ie/elixir/

20. **Clark**, **J**. *XSL transformations (Xslt). Version 1.0*. W3C, 1999. W3C recommendation.
http://www.w3.org/TR/xslt

21. **Denenberg**, **R**. "The library perspective". En: *QL'98, the query languages workshop*. 1998.
http://www.w3.org/TandS/QL/QL98/pp/denenberg.html

22. **Deutsch**, **A**.; **Fernández**, **M**.; **Florescu**, **D**.; **Levy**, **A**.; **Suciu**, **D**. *Xml-QL: a query language for xml*. Consultado en: 17-04-01.
http://www.w3.org/TR/1998/NOTE-xml-ql-19980819/

23. **Deutsch**, **A**.; **Fernández**, **M**.; **Florescu**, **D**.; **Levy**, **A**.; **Suciu**, **D**. "A query language for xml". En: *8th International world wide web conference*. 1999.
http://www.research.att.com/~mff/files/final.html

24. **Fernández**, **M**. *Xml-QL*. 1999, AT&T.
http://www.research.att.com/sw/tools/xmlql/

25. **Fernández**, **M**.; **Siméon**, **J**.; **Wadler**, **P**. Xml query languages: experiences and exemplars.
http://www-db.research.bell-labs.com/user/simeon/xquery.html

26. **Florescu**, **D**.; **Kossmann**, **D**.; **Manolescu**, **I**. "Integrating keyword search into xml query processing". En: *9th International world wide web conference*. 2000.
http://www.db.fmi.uni-passau.de/~kossmann/FKM00/paper.html

27. **Fuhr**, **N**. "Xirql: an extension of XQL for information retrieval". En: *Sigmod/Pods workshop on the web and databases*. 2000.
http://www.haifa.il.ibm.com/sigir00-xml/final-papers/KaiGross/sigir00.html

28. **Ives**, **Z**. **G**.; **Lu**, **Y**. "Xml query languages in practice: an evaluation". En: *Web age information management 2000*. 2000.
http://www.cs.washington.edu/homes/zives/projects/xmlquery.pdf

29. **Luk**, **R**. **W**. **P**.; **Leong**, **H**. **V**.; **Dillon**, **T**. **S**.; **Chan**, **A**. **T**. **S**.; **Croft**, **W**. **B**.; **Allan**, **J**. "Special issue on xml and IR". En: *Jasis*, en prensa, 2001.

30. **Maier**, **D**. "Database desiderata for an xml query language". En: *QL'98, the query languages workshop*. 1998.
http://www.w3.org/TandS/QL/QL98/pp/maier.html

31. **Malhotra**, **A**.; **Marsh**, **J**.; **Melton**, **J**.; **Robie**, **J**. *XQuery 1.0 and XPath 2.0 functions and operators version 1.0*. W3C, 2001. Working draft.
http://www.w3.org/TR/xquery-operators/#b2b7b6b4

32. **Navarro**, **G**.; **Baeza-Yates**, **R**. "A language for queries on structure and contents of textual databases". En: *ACM Sigir'95*, pp. 93-101.
ftp://sunsite.dcc.uchile.cl/pub/users/gnavarro/sigir95.ps.gz

33. **Quass**, **D**. "Ten features necessary for an xml query language". En: *QL'98, the query languages workshop*. 1998.
http://www.w3.org/TandS/QL/QL98/pp/quass.html

34. **Robie**, **J**. "Combining and querying xml data with XQL". En: *Xml 99, CGA conferences*. 1999.
http://www.infoloom.com/gcaconfs/WEB/philadelphia99/robie.HTM

35. **Robie**, **J**.; **Chamberlin**, **D**.; **Florescu**, **D**. *Quilt: an xml query language*. Consultado en: 17-04-01.
http://www.almaden.ibm.com/cs/people/chamberlin/robie_xml_Europe.pdf

36. **Sahuguet**, **A**. *Xslt as a query language for xml*.
http://db.cis.upenn.edu/XSLT/

37. **Schlieder**, **T**.; **Meuss**, **H**. "Result ranking for structured queries against xml documents". En: *Delos workshop on information seeking, searching and querying in digital libraries*. 2000.
http://www.inf.fu-berlin.de/~schlied/publications/delos2000.ps

38. **Theobald**, **A**.; **Weikum**, **G**. "Adding relevance to xml". En: *3rd International workshop on the web and databases* (*WebDB'00*). 2000.
http://www-dbs.cs.uni-sb.de/~weikum/webdb00-lncs.pdf

39. **Vegas**, **J**. *Un sistema de recuperación de información sobre estructura y contenido*. *Universidad de Valladolid*, 1999.

Adelaida Delgado Domínguez
adelaida@ipc4.uib.es
http://dmi.uib.es/people/adelaida

Ricardo Baeza-Yates
rbaeza@dcc.uchile.cl
http://www.dcc.uchile.cl/~rbaeza/spanish.html