Chapter 13

Document SQL (DSQL): A Conservative Extension to SQL as an Ad-hoc Querying Frontend for XQuery

Arijit Sengupta
Wright State University, USA

V. Ramesh
Indiana University, USA

ABSTRACT

This chapter presents DSQL, a conservative extension of SQL, as an ad-hoc query language for XML. The development of DSQL follows the theoretical foundations of first order logic, and uses common query semantics already accepted for SQL. DSQL represents a core subset of XQuery that lends well to query optimization techniques; while at the same time allows easy integration into current databases and applications that use SQL. The intent of DSQL is not to replace XQuery, the current W3C recommended XML query language, but to serve as an ad-hoc querying frontend to XQuery. Further, the authors present proofs for important query language properties such as complexity and closure. An empirical study comparing DSQL and XQuery for the purpose of ad-hoc querying demonstrates that users perform better with DSQL for both flat and tree structures, in terms of both accuracy and efficiency.

INTRODUCTION

XQuery, the query language for XML, originally proposed as early as 2001 (Don Chamberlin, Clark, et al., 2001), was ratified as a candidate W3C recommendation late 2005, and became an official W3C recommendation in January 2007 (Boag et al., 2007). With the increase in the popularity of XML as the next generation of documentation representation language for the hyped Web 2.0 (O’Reilly, 2005), the need for a standard way of retrieving information from XML documents was considered a critical issue, which resulted in the design and eventual recommendation of XQuery. XQuery came from a marriage of two directions of querying: (i) pattern-based languages based on the tree structure of XML documents such as XPath (Clark & DeRose, 1999) and XQL (Robie, Lapp, & Schach, 1998), and (ii) a more logic-oriented approach with conditions and output specifications such as XML-QL (Deutsch, Fernandez, Florescu, Levy, & Suciu, 1998). Interestingly, however,
both of these approaches used a syntactic convention significantly different from the predominant database query language SQL (Structured Query Language). Although there are some attempts towards including XML querying support in SQL, including SQLX (or SQL/XML), an effort by the International Standards Organization - SQL-03 (ANSI/ISO, 2003) from the International Standards Organization (ISO) to incorporate XML support directly into SQL, a common decision among XML query language designers was to create a completely new language for the purpose of querying XML data. One motivation for such a decision may have been the idea that the query language itself would use XML syntax. XQuery is defined as a “full programming language, and supports user-defined functions, with support for arbitrary levels of recursion and arbitrarily large memory usage” (Boag et al., 2007). This is a direction away from previous query language research, which tended to ensure the complexity of SQL stayed within reasonable complexity bounds. The problem is that having a full programming language may not be suitable for ad-hoc querying, which will explain why after 7 years of the development of XQuery, it is still not close to the level of popularity as an ad-hoc query language for XML.

Typically, a declarative (as opposed to a procedural) language is one that can specify an expression by declaring the structure and conditions of the intended result, instead of explicitly providing the steps necessary to obtain those results. For example, an SQL query need only specify the output attributes, the input relations, and properties of the output. The advantage of a declarative language is that the query engine can decide what steps to take to generate the output, by considering all query optimization possibilities. Some characteristics of XML schema make it possible to write queries using a declarative language. Although XML documents have a complex hierarchical structure, the strong presence of meta-data in XML documents makes it fairly intuitive to write declarative queries based purely on logical combinations of the properties of the intended results. Declarative query languages where the primary focus is on the properties of the result, rather than the process of extracting the result itself, are very suitable for structured data, because they allow the possibility of letting the system optimize the queries instead of relying on the users’ capabilities for writing an efficient query. We present a declarative query language, Document SQL (DSQL) that has the same look and feel as SQL and was designed by updating the semantics of SQL operations in the structured document domain. At the same time, DSQL was designed such that all queries written in it have equivalent counterparts in XQuery. Thus, by using such a language, users can take advantage of their existing SQL knowledge when writing ad-hoc queries without losing the expressive power of XQuery. In addition to describing the syntax and semantics of the language we also present the results of an experiment that investigates whether a language like DSQL make it possible for users to write more accurate and efficient queries than XQuery.

The rest of the paper is organized as follows. We start by reviewing some current research in this area. We then illustrate a data model for representing XML documents. We then develop the DSQL query language, and provide a comparison between DSQL and XQuery. Next, we describe a study comparing DSQL with XQuery, discuss the findings of this study and finally, provide some concluding remarks.

LITERATURE REVIEW

Query languages for documents and other hierarchical structures have been a topic of research ever since XML became a force in document representation. W3C (http://www.w3.org) initiated a working group for XML Query, for the purpose of designing query languages for docu-