Journal of Database Management
Volume 27 • Issue 3 • July-September 2016

Path Expressions in SQL:
A User Study on Query Formulation

Marko Junkkari, School of Information Sciences, University of Tampere, Tampere, Finland
Johanna Vainio, School of Information Sciences, University of Tampere, Tampere, Finland
Kati Iltanen, School of Information Sciences, University of Tampere, Tampere, Finland
Paavo Arvola, School of Information Sciences, University of Tampere, Tampere, Finland
Heidi Kari, School of Information Sciences, University of Tampere, Tampere, Finland
Jaana Kekäläinen, School of Information Sciences, University of Tampere, Tampere, Finland

ABSTRACT

This article focuses on testing a path-oriented querying approach to hierarchical data in relational databases. The authors execute a user study to compare the path-oriented approach and traditional SQL from two perspectives: correctness of queries and time spent in querying. They also analyze what kinds of errors are typical in path-oriented SQL. Path-oriented query languages are popular in the context of object-orientation and XML. However, relational databases are the most common paradigm for storing data and SQL is most common for manipulating data. When querying hierarchical data in SQL, the user must specify join conditions explicitly between hierarchy levels. Path-oriented SQL is a new alternative for expressing hierarchical queries in relational databases. In the authors' study, the users spent significantly less time in writing path-oriented SQL queries and made fewer errors in query formulation.

KEYWORDS

Hierarchical Data, Path-Orientation, PathSQL, Query Language, Relational Databases, SQL, User Test

INTRODUCTION

Query languages are focused on accessing data and relationships in databases. A database is organized based on the underlying data model. Relational databases are the most common database paradigm and are based on the relational data model (Codd, 1983). In the relational data model, the data is organized in relations (tables) and the data records in different relations are associated with each other by foreign key constraints. SQL (Chamberlin et al., 1974) is the most common query language for relational databases. In it, the user specifies the target of the query, source relations, conditions among relations, and possible join operations among relations. Relational databases are general purpose databases and all relationships are represented similarly.

In many domains, the data has a hierarchical nature, meaning that the data can be viewed through several hierarchical levels (Bernauer, 1996; Burrough and McDonnell, 1998; Niemi and Järvelin, 1995; Motschnig-Pitrik and Kaasböll, 1999; Pazzi, 1999; Urtado and Oussalah, 1998). For example, a physical assembly contains components that in turn contain smaller components, etc. (Junkkari, 2005). In relational databases, the components and the composites they form are represented in separate relations and part-of relationships are represented by foreign key constraints (David, 2003). In SQL, the part-of relationships are manipulated like any association among data. This means that

DOI: 10.4018/JDM.2016070101

the user must explicitly specify join operations or foreign key constraints between a composition and its components. In large hierarchies, the user must specify a great number of join operations or equal conditions for foreign key constraints. In the research literature, join operations are shown to cause difficulties for users in query writing (Chan, Lu and Wei, 1993; Chan, 2007; Rho and March, 1996; Smelcer, 1995).

Vainio and Junkkari (2014) have recently proposed an alternative way to formulate hierarchical queries, i.e. they embed path expressions into SQL. We call their language PathSQL. Path expressions are widely used in the context of hierarchical data in early data models (Elmasri and Navathe, 1989) and later on in the context of semi-structured data models XML (Bray et al., 1998; Clark and DeRose, 1999; Deutsch et al., 1999) and JSON (Ong et al. 2014). Path-oriented query languages are also used in object-oriented databases (Cattell and Barry, 2000; Cluet, 1998; Alashqur et al., 1989). However, in spite of the popularity of path-oriented query languages, they have not been tested from the perspective of how easy they are to learn and use compared with SQL, the most popular query language. Now, the work by Vainio and Junkkari enables the testing of the path-oriented query approach.

Vainio and Junkkari (2014) demonstrated that queries with path expressions are shorter and more compact. Nevertheless, they do not provide experimental evidence about the usefulness of their approach. We agree with Vainio and Junkkari that path expressions shorten some queries, but this does not ensure that the path-oriented query language would be easier or faster to use. In the present paper, we test traditional SQL and PathSQL languages with users having experience in SQL and minor experience in path-oriented query languages. We compare the time spent in formulating queries in both languages and the number of correct and erroneous queries per user and language. We also analyze what types of errors are common in formulating path-oriented queries.

The rest of the paper is organized as follows. Next, query test approaches are surveyed. Then, the SQL and PathSQL languages are introduced briefly. The main focus is on PathSQL because it is a new querying approach and thus not as well-known as the original SQL. After that, the test setting and query topics are introduced. Again, we focus on introducing of PathSQL queries; related SQL queries are given in an appendix. Then, the test is analyzed and its results are given. We compare the correctness of the queries and the time spent in writing them in PathSQL and traditional SQL. We also classify and analyze the errors of the PathSQL queries. Then, the results and possible extensions of PathSQL are discussed. Finally, the conclusions are given.

RELATED WORK

Relational database query languages have been tested with users since the 1970s. Welty (1990) gives a valuable overview of studies involving SQL from 1971–1989. These tests, e.g., evaluate the ease-of-use of a query language or compare two or more query languages. Reisner (1981) has studied user tests of query languages, and states that six different tasks and six different tests have been used to measure ease-of-use. These tasks are query writing, query reading, query interpretation, question comprehension, memorization, and problem solving. The most popular tests are the final exam of learning, immediate comprehension, and retention. Some of the earlier user tests on relational database query languages are briefly represented below. The most common task is query writing and the most common test is a final exam.

The ease-of-use of SQUARE and SQL was tested by Reisner et al. (1975). Four groups of students participated in the test, 61 students in all. Two of the groups were programmers and the other two groups were non-programmers. One group of programmers and one group of non-programmers where taught SQL and the others where taught SQUARE. Non-programmers were taught for 14 academic hours whereas the programmers were taught for 12 academic hours. A final exam was executed after the instruction and a retention test was conducted a week later. The main task was query writing. The authors' conclusions were that programmers were able to learn SQL "more completely" than non-