SQL for NoSQL Databases: Déjà Vu

Christoph Bussler
Oracle Corporation
500 Oracle Parkway
Redwood Shores, CA 94065

Christoph.Bussler@oracle.com

1. INTRODUCTION

Most NoSQL databases are extremely painful for their users when querying data: in order to retrieve data, users must write a program in their favorite programming language and execute it (rinse and repeat for every query). Most NoSQL databases expose only a limited programming language interface, but usually not a declarative query language like SQL.

Existing relational database technology experience tells us that declarative database query languages are beneficial for different reasons, including application independence from access path considerations and decades of sophisticated query optimization technologies [2]. In context of most NoSQL databases, however, users are forced to implement complex query algorithms manually themselves using low-level programming language interfaces that provide only limited data access functionality.

There is an ongoing debate on the usefulness and the technical feasibility of a SQL query interface for NoSQL databases. The debate can be kept short: SQL in context of NoSQL databases is of course technically possible and unquestionable useful: a total no-brainer.

An argument can be made that SQL is actually going to be an important success factor ('kingmaker') for NoSQL databases [1]. At the end of the day it is important to be able to query data easily, reliably, accurately and declaratively.

2. THE KNOWN BIT: SQL FOR NF2 RELATIONS

Going back far enough in database history reveals that SQL was proposed on NF2 relations [3]. This fundamentally means that querying hierarchical data sets (or "documents") is possible with the corresponding SQL syntax and execution semantics. Clearly, SQL is not restricted to a single-valued relational model.

The key elements of SQL and operators for NF2 relations are:

- Queries inside projections for composite values. For example, 'select a, (select * from b limit 5) from c' selects columns 'a' and 'b' in relation 'c', whereby 'b' is a composite data type and only the first 5 elements of 'b' are selected for each row.
- 'nest' and 'unnest' operators in order to 'flatten' or 'unflatten' NF2 relations. For example, 'select a, b from unnest c on b' creates a row for each element in 'b' with the value of 'a' corresponding to 'b' before the unnest invocation (Cartesian product of 'a' of a row and every element in 'b' of the same row).

3. THE TRICKY BIT: DOCUMENT-SPECIFIC SCHEMA

The early efforts of defining SQL on NF2 relations assumed a fixed and defined schema, which is usually not supported in Document NoSQL databases. In Document NoSQL databases each document has its own schema and as a consequence there might be a heterogeneous document set in a database (in the extreme case each document has a different schema). A query across schema varying documents encounters schema differences that it has to be able to deal with (not only in projection, but also selection, grouping, etc.) in a well-defined semantics.

For example, querying against a missing property that is used in a query's join criteria must be well-defined. Is a missing property interpreted as NULL? Or does its absence mean that there is no join possible for this document? Arrays or subdocuments can be used as join criteria as well and this requires well-defined comparison operations on complex types.

The same property can be of different type in different documents. If used as a join or selection criteria, type casting as well as the query behavior must be well-defined when types are incompatible and cannot be cast to each other.

4. THE WAY FORWARD: JUST DO IT

Implementing an extended SQL for NoSQL databases is possible and is extremely useful (e.g. [4]). This requires extensions to SQL like those proposed in context of NF2 relations without assuming a global schema. This is not against the nature of SQL at all and outlines a clear path forward for databases to provide a declarative query language implementation for NoSQL data. Users can then actually query NoSQL data easily and without having to write programs against low-level database interfaces.

5. REFERENCES

[1] http://realprogrammer.wordpress.com/2014/03/04/sqltheking makerofnosqldatabases/

[2] Mohan, C.: History Repeats Itself: Sensible and NonsenSQL Aspects of the NoSQL Hoopla. In: Proceedings of EDBT/ICDT '13, March 18–22, 2013, Genoa, Italy.

[3] https://www.google.com/search?q=sql+nf2

[4] https://docs.oracle.com/database/121/ADXDB/json.htm#ADXDB6246

7th Biennial Conference on Innovative Data Systems Research (CIDR '15) January 4-7, 2015, Asilomar, California, USA.