Chapter 5
Bridging Relational and NoSQL Worlds

ABSTRACT

The chapter discusses the fact that the development and use of NoSQL databases showed that neither everything was good in NoSQL nor everything was so bad in relational databases. Namely, when operating with data, NoSQL databases have identical requirements for entering, updating, deleting or searching data, or for the data manipulation that SQL already resolved long ago. Therefore, it is not surprising that further development of many NoSQL databases shifted towards supporting SQL, which is one of the topics of this chapter. Namely, database users are generally not concerned with details about how data is stored. Rather, they want to have the possibility to view and analyze data together, regardless of whether the data is stored in relational or NoSQL databases. Therefore, vendors of relational databases were forced to look for solutions that would allow them to work with data stored in NoSQL databases as well.

INTRODUCTION

The first part of the book (Chapters 1, 2, 3, and 4) describes the reasons behind the appearance of NoSQL databases, as well as their main characteristics. When developing the first NoSQL databases, it was imperative to depart from relational databases as much as possible (elimination of ACID transactions,

DOI: 10.4018/978-1-5225-3385-6.ch005

SQL, etc.). First NoSQL databases were promoted as databases developed for coders, with one of the primary goals being to avoid the eternal impedance mismatch between the object-oriented approach to programming and relational databases. That is why SQL was avoided, and the focus was on the development of specific APIs and programming languages. However, such an approach soon turned out to be a limiting factor of wider implementation of NoSQL databases because it lacked the critical mass of well-trained developers (see Chapter 2), and a plethora of different APIs and programming languages was not beneficial for their faster training. On the other hand, the advent of many unstandardized APIs and abandonment of SQL were not welcomed with open arms by the SQL community that consisted of a large number of well-trained developers.

Further development and use of NoSQL databases showed that neither everything was good in NoSQL nor everything was so bad in relational databases. Namely, when operating with data, NoSQL databases have identical requirements for entering, updating, deleting, or searching data, or for the data manipulation that SQL already resolved long ago. Therefore, it is not surprising that further development of many NoSQL databases shifted toward supporting SQL, which is the topic of "From NoSQL Toward SQL."

As already emphasized, the first NoSQL databases were created out of necessity because it was not possible to resolve Big Data challenges using relational databases (see Chapter 2). Also, one of the basic postulates in the development of NoSQL databases is that there is not just one solution for all (data) problems. Practice shows that transaction-oriented requirements can be better solved using relational databases (ACID, SQL), whereas NoSQL databases are better suited to specific Big Data demands. However, these problems are not always clearly separated, and in everyday life they are often intertwined. Database users are generally not concerned with details about how data is stored: they want to have the possibility to view and analyze data together, regardless of whether the data is stored in relational or NoSQL databases. Therefore, producers of relational databases were forced, partly for the sake of solving customers' requirements, partly for the sake of keeping market share, to look for solutions within relational databases that would allow them to work with data stored in NoSQL databases as well. "Extending Relational Databases" describes how that was accomplished by the three largest database vendors: Oracle, Microsoft, and IBM.