Extending Graph Databases With Relational Concepts

E

Kornelije Rabuzin
Faculty of Organization and Informatics, University of Zagreb, Croatia

Mirko Čubrilo
University North, Croatia

Martina Šestak
Faculty of Electrical Engineering and Computer Science, University of Maribor, Slovenia

INTRODUCTION

Data science is a topic that uses different scientific methods, techniques, and algorithms for extracting useful information and knowledge from data. There are several reasons to use data science; for explaining what happened in the past, what is happening in the present, and even try to predict what is most likely to happen in the future. Data science is an interdisciplinary field, which includes data analysis, machine learning, business intelligence, quantitative methods, data visualization, data preparation, statistics, etc. Due to the heterogeneity of the subjects inside data science, it is not easy to be an expert in all of them, and it takes a large amount of time to master the different subjects. Because of that, some people predict that educational institutions will not be able to produce enough data scientists in the near future, which was already recognized by some countries, including the EU.

The core component of data science is data, and without it, data scientists would not be able to perform their work. Another important topic is the data source, which can be categorized as un-, semi- and structured sources. As a brief explanation, unstructured data can be for example an email - a collection of words without any structure behind it - while structured data can be found in a database where data is grouped into tables, and each row of some table shares the same structure with the rows that come before or after. Regarding semi-structured sources, a good example are XML documents, since XML nodes in the same document can have different structures. A relational database, which is a type of structured data source, can be a good data source, since people are familiar with them.

During the past decade, computer systems had to store and manage large amounts of data coming from different sources. For example, the Internet is being used on a daily basis by millions of users, and the size of generated text, messages, searches, posts, images, videos, etc. is enormous. Furthermore, many Internet of Things (IoT) devices are connected to the Internet and they also tend to generate large amounts of different types of data. While in the past one would easily talk about gigabytes and terabytes of data, today it is usual to deal with petabytes and exabytes, even though this scale is expected to grow to zettabytes and yottabytes in the future. It is clear that turning to this new field (Big Data), there is a constant need to find scalable solutions for efficient data storage and management. When the frequency and volume of data generation started to increase 15 years ago, the goal was to rethink and find new ways of handling and storing large amounts of data. The solutions that were used in early 2000s were just not suitable anymore for efficiently handling the huge amounts of generated data. Relational databases,

DOI: 10.4018/978-1-7998-9220-5.ch027

which were efficient solutions for everyday business transactional applications, are not appropriate for extremely large amounts of data, with possibly different structures and schema. These databases were not capable to store and manage the data in a satisfactory manner, leading to difficulties in real time data processing, as well as ad-hoc data querying.

In order to store and manage the data, NoSQL database systems were introduced. These are used today by many companies for different purposes, and they can be categorized into four main NoSQL database types: document-oriented, column-oriented, key-value and graph databases. In this chapter, the focus is put on graph databases, as they turned out to be an excellent choice to store and query large amounts of interconnected data, often generated by modern information systems. In general, graph databases represent a database solution based on a graph data structure, where data is stored in the form of nodes connected with relationships, where both elements can have properties as attributes, which describe real-world objects. The other NoSQL databases types also have their own advantages. For instance, key-value databases can quickly retrieve the value for the specified key; document-oriented systems can store all the important data for an entity in a single document; column-store systems are similar to relational databases with an increased flexibility to the schema. Thus, it can be concluded that each type is suitable for different application and can be interchangeably used to resolve different challenges.

As graph databases have certain advantages over relational databases, they have been increasingly used as a source of information in data science projects. There are already some datasets with a graph database structure available for data science projects ("Awesome Public Datasets as Neo4j Graph," n.d.). One author of this chapter was included in a previous data science project that used graph databases in the telecommunication industry as well, although the use of graph databases in telecommunication industry is not entirely new (Lehotay-Kéry & Kiss, 2020). The developed solution detected, in real time, potential problems that could significantly reduce the quality of the service. Health data science can also benefit from graph databases. In (Liu et al., 2021) it was proposed a graph database called EpiGraphDB, whose purpose was to store biomedical and epidemiological relationships between data, with the goal of using the solution as a database and data mining platform for health data science.

In this chapter, the idea is to briefly explain and analyse what concepts that characterize relational databases are still missing in graph databases, and how they could be implemented. Some of the reasons to do this are related to data quality, which is an important part of data science. If data quality is low, the conclusions that are drawn from the analysis of the data could be potentially wrong, especially when handling large amounts of data. Data quality in graph databases could be increased, if the missing concepts of relational databases were implemented. NoSQL systems were introduced to tackle the data-related problems in practice and because of that, it is important to understand that NoSQL databases do not have a solid mathematical foundation, as relational databases, nor a standardized query language, as SQL. This fact also applies to graph databases. For example, graph databases use query languages different from SQL, where triggers, node inheritance, advanced integrity constraints, among others, are not supported. Although graph databases have some advantages, as a high speed of reading of interconnected data when compared to relational databases, they still have some weaknesses that are not present in relational databases.

A few years ago, the authors received several university grants to explore what is missing in graph databases and what could be further implemented in order to extend the list of functional characteristics. First, query languages for graph databases were examined, and it was determined that the majority of people used the Cypher Query Language (CQL), or Gremlin. However, there are several languages, each with its own syntax, which implies a certain learning curve for its users. The authors' idea was that a language similar to SQL could help the improvement of this learning process. Further on, the sup-