Graph Database Management Systems: The Past, the Present, and the Future

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

Martina Šestak
Faculty of Organization and Informatics, University of Zagreb, Croatia

INTRODUCTION

Nowadays, the increased amount and complexity of connected data stimulated by the appearance of social networks has shed a new light on the importance of managing such data, especially handling information about the connections. The most natural way of representing connected data is to represent them as nodes connected with relationships forming a graph. The concepts of graph theory have been used in many occasions to handle connected data in databases over the years.

The idea of storing data as a set of nodes and edges comprising a graph was implemented in various forms in data models used in the past. The network data model, developed in late 1960s, can be considered as the first data model, which most accurately incorporated this idea. However, it was not long before the relational data model appeared, and took over the entire database market for years, which it dominates even nowadays.

Even though several solutions have been introduced, which can be used to store graph-like data in available relational database management systems (RDBMSs), such as Oracle Graph or MariaDB OQGraph, NoSQL graph databases are considered to be the most efficient solution for storing such data, since the idea of storing highly connected data is their primary goal and purpose.

The possibilities of using graph databases in various application domains continue to grow. For instance, the most popular graph databases are Facebook Social Graph developed by Facebook to view connections between friends, or Amazon's graph-based recommendation system (Amazon Neptune service). Sieger discusses the possibilities of introducing concepts of graph databases to modern businesses, and how graph databases could be used for Supply Chain Management or finding deeper connections between patients with similar diseases (Sieger, 2016). Moreover, according to D. Woods, due to their ability to efficiently gain insight into connections between pieces of information, graph databases play (and will play) an important role in transforming modern businesses to data-driven organizations, which make use of their data through the concept of Master Data Management (Woods, 2015).

Since their beginnings, there have been many graph database management systems (GDBMSs) available on the database market (e.g., Neo4j, TitanDB, AllegroGraph, FlockDB, InfiniteGraph, etc.). Recently, new emerging trends can be observed on the GDBMS market; aside from Neo4j, which is still the most popular and constantly developed GDBMS, other „native" GDBMSs are starting to be replaced and „outgrown" by multi-model databases, such as OrientDB, ArangoDB, etc.

The aforementioned trends and solutions indicate that it is worth exploring how graph-like data has been stored and manipulated over the years in various data storage solutions, and which current data storage options are available to store and manipulate such data.

DOI: 10.4018/978-1-7998-3479-3.ch053

Therefore, the objective of this article is to give a timeline overview of developed graph data storage solutions in order to gain insight into past, present and future trends of GDBMSs. Additionally, throughout this article, the most influential factors and reasons for changes in trends in GDBMSs' usage will be explored and analyzed.

BACKGROUND

In general, there are various definitions of a graph database; De Virgilio et al. defined graph database as a "multigraph g=(N,E), where every node $n \in N$ is associated with a set of pairs <key, value>, and every edge $e \in E$ is associated with a label" (De Virgilio, Maccioni, & Torlone, 2013), whereas He and Singh defined graph database as a set of graphs D={G_1, G_2, …, G_m}, where graph G is denoted by (V, E), V being a set of all vertices, and E being a set of all edges (He & Singh, 2006).

Graph database model can be defined as a data model, in which "data structures for the schema and instances are modeled as graphs or their generalizations, and data manipulation is expressed by graph-oriented operations and type constructors" (Angles & Gutierrez, 2008). Graph database model consists of three components (Angles & Gutierrez, 2008):

- structural component (graph data structures),
- operational component (graph-oriented operators), and
- integrity component (integrity constraints).

Nowadays, the most commonly used graph database model is the property graph data model (Figure 1), which can be defined as a "multigraph data structure, in which graph elements (vertices and edges) can have properties/attributes"(Ciglan, Averbuch, & Hluchy, 2012).

Data stored in a graph database can be queried by using a graph query language, which can provide support for different graph-related operations, such as graph union/intersection/difference, graph filtering, adjacency, path traversal, or pattern matching queries (Kaplan, Abdulla, Brugger, & Kohn, 2007).

Figure 1. Sample property graph data model (Neo4j Inc., 2018)