1. A Distributional Semantics Approach for Selective Reasoning on Commonsense Graph Knowledge Bases : Natural Language Processing and Information Systems, 2014. - 21-23
2. From Frequency to Meaning: Vector Space Models of Semantics : Journal of Artificial Intelligence Research, 2010. - 141-188
3. Hierarchical Knowledge Bases and Efficient Disjunctive Reasoning : Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning/Ronald J. Brachman, Hector J. Levesque, Ray Reiter M. Kaufmann, 1989. - 33-43
4. Latent Semantic Indexing: An overview INFOSYS 240 Spring 2000

INVESTIGATION OF THE EFFICIENCY DEPENDENCE OF RELATIONAL AND GRAPH DATABASES ON DATA

Nedaiev I. S., student,
Gubaryeva O. S., PhD, Associate Professor,
Kharkiv National University of Radio Electronics

There are various types of databases, each with its own purpose, strengths and weaknesses. Since each type specializes in solving its own specific task, it is impossible to determine which one is the best overall. However, if a specific situation is considered, it is possible to determine which type of database is better suited for that particular case.

In general, many DBMSs support different data types, and almost any information can be converted to the appropriate format one way or another. However, such solutions can greatly affect the speed and quality of their work, or even their security and fault tolerance. For example, relational databases are considered a classic and most information can be represented as tables and relationships between them [7]. However, complex relationships significantly affect performance and resource consumption, as well as complicate the architecture of the database itself, which can lead to its malfunction. On the other hand, other types of databases may be better suited for such situations but will not be able to reproduce a simple scenario of using

relational databases. Such comparisons can be made for each type of database.

In this study, we investigate the efficiency of relational and graph [6] databases and their dependence on data. The object of the study is the process of working with graph and relational databases, while the subject of the research is the methods of storing, processing, and accessing data in relational and graph database management systems, specifically lazy loading using the Entity Framework. The aim of the study is to analyze the methods of working with relational and graph databases, and to develop recommendations for their use depending on the input data, as well as determining the feasibility of using lazy loading with the Entity Framework.

Relational and graph databases are two commonly used types of database management systems that are crucial to the effective management of data. These systems differ in the way they store and organize data, which can have an impact on their efficiency. In order to determine the most effective way to manage data, it is important to investigate the efficiency of these systems and their dependence on the type of data being stored.

In the literature, there are some general recommendations for choosing a database, but there is no clear distribution – it is up to the developers to decide. The purpose of this paper is to determine the optimal scenarios for using different types of databases with different types of source data.

Many studies indicate that graph databases are much more efficient than relational databases if the database contains many relationships between tables [2, 5], while with a large amount of data and a small number of relationships, their performance is much lower [1, 3, 4].

The object of this study is the process of working with graph and relational databases. This involves examining the way data is stored, processed, and accessed in these systems, and how they differ in terms of their efficiency.

The subject of this study is the methods of storing, processing, and accessing data in relational and graph database management systems. Specifically, we will be looking at lazy loading using the Entity Framework. Lazy loading is a technique used to defer the loading of related data until it is actually needed.

Regression and correlation analyses will be conducted to examine the performance of the databases depending on the data samples. Regression analysis will be used to forecast the performance and determine the theoretically optimal point and extrema, while correlation analysis will investigate the relationship between the data sample and performance. The optimal sample will be determined for each data provider, followed by a comparison of the data providers' performance under different conditions, and conclusions will be drawn regarding the feasibility of using lazy loading and the relationship between performance and data sample.

As a result of the research, usage scenarios for graph databases and usage scenarios for relational databases, as well as the advantages and disadvantages of graph and relational databases depending on the input data, and the relevance of using lazy loading in Entity Framework should be obtained.

To do this, a project needs to be developed to test the performance of various data providers, libraries with implementations of graph databases (Neo4j) and relational databases (Entity Framework and Entity Framework with lazy loading), and databases themselves with different relationship structures and data volume.

References

1. Лазурик В.М., Тимошенко Є.С. Застосування графових баз даних для моделювання соціальних графів // Вісник Харківського національного університету імені В.Н. Каразіна, серія «Математичне моделювання. Інформаційні технології. Автоматизовані системи управління». 2019. (43). С. 46-53.

2. Codd E. F. A relational model of data for large shared data banks. Communications of the ACM. 1970. Т. 13, № 6. С. 377–387. URL: https://doi.org/10.1145/362384.362685 (дата звернення: 03.04.2023).

3. Comparison of relational databases and graph databases. https://stackoverflow.com/. URL: https://stackoverflow.com/questions/13046442/comparison-of-relational-databases-and-graph-databases (дата звернення: 03.04.2023).

4. Kotiranta P., Junkkari M., Nummenmaa J. Performance of graph and

relational databases in complex queries. Applied sciences. 2022. Т. 12, № 13. С. 6490. URL: https://doi.org/10.3390/app12136490 (дата звернення: 03.04.2023).

5. Lazarska M., Siedlecka-Lamch O. Comparative study of relational and graph databases. 2019 IEEE 15th international scientific conference on informatics, м. Poprad, Slovakia, 20–22 листоп. 2019 р. 2019. URL: https://doi.org/10.1109/informatics47936.2019.9119303 (дата звернення: 03.04.2023).

6. What is a graph database?. https://neo4j.com/. URL: https://neo4j.com/developer/graph-database/ (дата звернення: 03.04.2023).

7. What is a relational database?. https://azure.microsoft.com/. URL: https://azure.microsoft.com/en-us/resources/cloud-computing-dictionary/what-is-a-relational-database/#whatis (дата звернення: 03.04.2023).

COMPARATIVE ANALYSIS OF THE CONTENT RENDERING TECHNIQUES FOR WEB APPLICATION DEVELOPMENT

Vasyliev D.O., student,

M. Suknov, PhD, Associate Professor,

Kharkiv National University of Radioelectronics,

The development of web applications has evolved over the years, from simple websites to complex Web applications and streaming platforms such as YouTube and Netflix, with a range of approaches available. Nowadays, we can distinguish three main techniques: Single-Page Applications (SPAs), Server-Side Rendering (SSR), and Static Site Generation (SSG). These approaches differ in terms of their impact on user experience, with implications for speed and responsiveness [1]. In this study, we compared the user experience of SPAs, SSR, and SSG web applications, using key metrics such as time to first byte, time to first paint, and time to interactive.

The development of web applications has seen a proliferation of approaches, including SPAs, SSR, and SSG. These approaches differ in terms of how the application is rendered on the client or server side and have different implications for user experience [2]. SPAs load once and dynamically update content, while SSR pre-