A METHODOLOGICAL APPROACH FOR CONVERTING RELATIONAL TO GRAPH DATABASES

Stefan Krstović*[1], Ognjen Pantelić[1], Ana Pajić Simović[1]
[1]University of Belgrade – Faculty of Organizational Sciences
*Corresponding author, e-mail: stefan.krstovic@fon.bg.ac.rs

OBJECTIVE

Graph databases are better for data containing many relationships, and are most common with social networks and similar information systems. Many systems that would benefit from using graph databases do not utilize them. This paper aims to increase their adoption rate by streamlining the migration process.

The main question answered in this paper is which steps should be taken to reliably and consistently transfer data from a relational DBMS (DataBase Management System) to a graph DBMS. A demonstration of the transfer is provided, along with a comparison between queries in the two environments.

Microsoft Access is used to manage the relational database, while Neo4j and the Cypher query language are used for the graph database. The approach can be used by institutions and companies to make their information systems better suited to their work areas.

METHODOLOGY

The starting point was developing a theoretical approach to transferring data between the systems. Previous research, notably by Virgilio, Roberto & Maccioni (2013) influenced this development phase.

The approach was refined by applying the proposed steps to the Northwind demo database from Microsoft (Microsoft, 2019). Afterwards, all of the data was migrated. The database overview is provided in Figure 1.

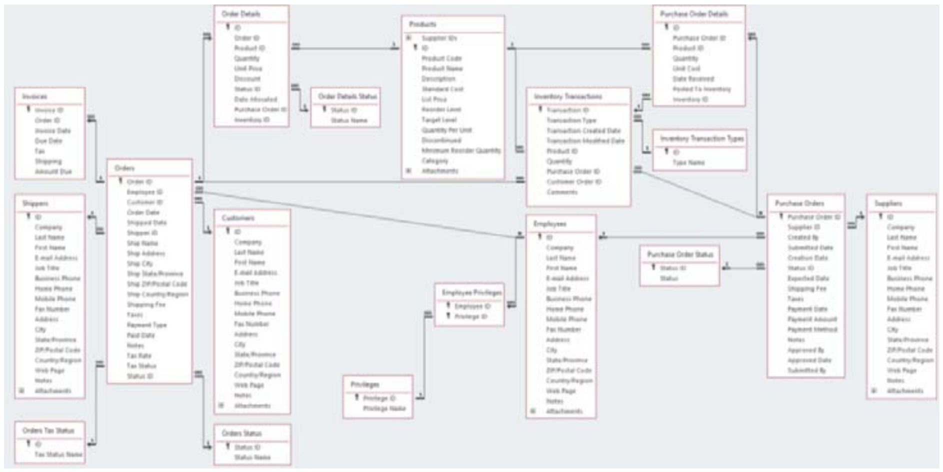

Figure 1: The overview of the database used

The last step was to compare the queries between the two systems to determine the superior simplicity, efficiency and clarity of graph queries.

RESULTS

The approach discovered can be divided into 3 major sections:

1. Preparing the initial database. This step brings the database into a desired state and helps discover its characteristics, primarily how the entities relate to one another and which dependencies exist.
2. Loading and properly connecting the data in the graph system. The dependencies and relationships discovered during the previous step should be used to determine how the connections should be implemented in the graph database. The join table (which stores the many-to-many relationship data between two tables) can be swapped for a set of relationship entities in the graph DBMS. Entities of the same type can have varying attributes without the need for separation, unlike in the RDBMS.
3. Optimizing the resulting graph database. Some artefacts that remain in the system are redundant after the conversion and should be removed to reduce bloat and minimize the chance for future mistakes. These include join tables and foreign key attributes (as their function is performed by the relationship entities).

The SQL queries are more complex, difficult to write and error-prone. Their Cypher equivalents are simpler, as the relationship patterns required can be described through ASCII art, rather than JOIN statements. The more tables required for the query, the higher the discrepancy.
In addition, the user can easily request a graphical representation of the results, so insights can be inferred at a glance. Figure 2 is an example of how buyers and the products they bought, connected by the orders, could be represented graphically.

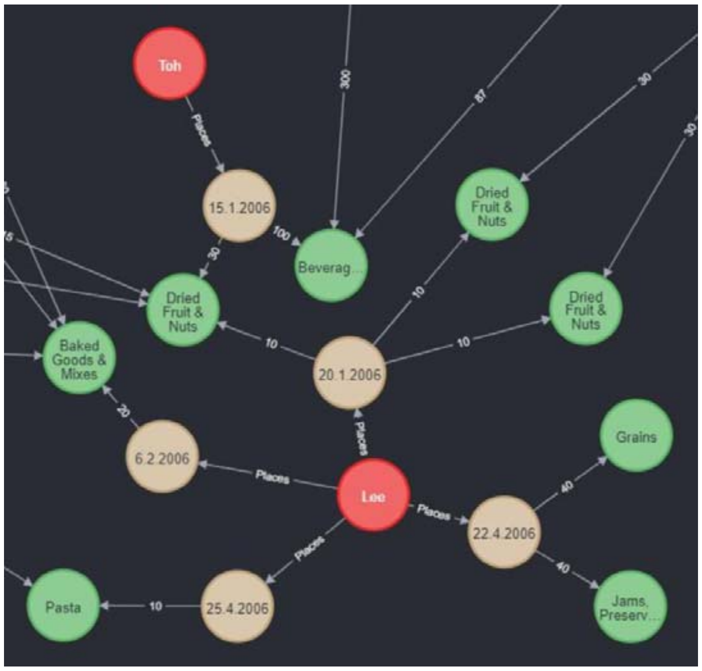

Figure 2: A graphical representation of buyers and the products they bought

CONCLUSION

The steps laid out previously provide a clear path to migrating data between the two systems. Unlike the approaches described in other literature (Virgilio et al., 2013; Ramachandran, 2015) which outline a more general procedure, this paper defines specific recommendations for dealing with different cardinalities and removing redundancies.

The limitation of this study, which will be addressed in future work is the automation of the migration process, so that the user could provide only the minimum necessary information, and a software tool could take care of the rest.

Keywords: *relation, graph, database, conversion*

REFERENCES

[1] Virgilio, R., Maccioni, A., Torlone, R. (2013). Converting relational to graph databases. *First International Workshop on Graph Data Management Experiences and Systems.* DOI: 10.1145/2484425.2484426.

[2] Lazarević, B., Marjanović Z., Aničić N., Babarogić S. (2016). Baze podataka (Databases). Faculty of organizational sciences, University of Belgrade.

[3] Microsoft. (2019). Get the sample databases for ADO.NET code samples. Retrieved from docs.microsoft.com/en-us/dotnet/framework/data/adonet/sql/linq/downloading-sample-databases (Accessed 1.2.2022)

[4] Lutkevich B., Biscobing J. (2021). Relational database. Retrieved from searchdatamanagement.techtarget.com/definition/relational-database (Accessed 4.2.2022)

[5] IBM Cloud Education. (2021). Structured vs. Unstructured Data: What's the Difference? Retrieved from: www.ibm.com/cloud/blog/structured-vs-unstructured-data (Accessed 3.2.2022)

[6] Neo4j. (n.d.). Model: Relational to Graph. Neo4j Developer. Retrieved from: neo4j.com/developer/relational-to-graph-modeling/ (Accessed 25.1.2022)

[7] Ramachandran, S. (2015). GRAPH DATABASE THEORY, Comparing Graph and Relational Data Models. LambdaZen.

[8] Ugander, J., Karrer, B., Backstrom, L., & Marlow, C. (2011). The anatomy of the Facebook social graph. arXiv preprint arXiv:1111.4503.

[9] Codd E. F. (1970). A relational model of data for large shared data banks. Commun. ACM 13, 6 (June, 1970), pp. 377-387. DOI: 10.1145/362384.362685

[10] Foote K. (2017). A brief history of database management. Dataversity. Retrieved from www.dataversity.net/brief-history-database-management/