КІБЕРБЕЗПЕКА: освіта, наука, техніка
CYBERSECURITY: EDUCATION, SCIENCE, TECHNIQUE
№ 1 (29), 2025
ISSN 2663 - 4023

DOI 10.28925/2663-4023.2025.29.847
UDC 004.65

**Anatolii Kurotych**
Student of the Faculty of Informations Technologies and Mathematics
Lesya Ukrainka Volyn National University, Lutsk, Ukraine
ORCID ID: 0009-0006-8186-4063
*akurotych@gmail.com*

**Lesia Bulatetska**
PhD, Associate Professor
Associate Professor at the Department of Computer Science and Cybersecurity
Lesya Ukrainka Volyn National University, Lutsk, Ukraine
ORCID ID: 0000-0002-7202-826X
*Bulatetska.Lesya@vnu.edu.ua*

**Oksana Onyshchuk**
PhD, Associate Professor
Associate Professor at the Department of Computer Science and Cybersecurity
Lesya Ukrainka Volyn National University, Lutsk, Ukraine
ORCID ID: 0000-0002-8342-3011
*Onyshchuk.Oksana@vnu.edu.ua*

# RECONSTRUCTING ENTITY RELATIONSHIPS IN DATABASE SCHEMAS WITH PLANTUML AND LLMS

**Abstract.** The article explores the potential of using Large Language Models (LLMs) for automatically restoring relationships between tables in SQL databases with incompletely defined foreign keys. To evaluate the ability of LLMs to infer foreign keys from textual descriptions of table structures, an experimental database was created. The database schema, excluding relationships, was provided as input to two large language models: ChatGPT-4o and Claude 3.7 Sonnet. For analysis purposes, only basic information was provided to the LLMs: table names, field names, and primary keys, without any data examples. The ChatGPT-4o model successfully detected all relationships between tables but demonstrated limitations in determining the types of these relationships: all were classified as "one-to-one", regardless of their actual structure. This indicates the model's inability to accurately interpret the type of relationships based on textual descriptions. In contrast, the Claude 3.7 Sonnet model not only correctly identified all existing relationships, but also correctly determined their types (e.g., one-to-many), demonstrating higher accuracy and a deeper understanding of the database structure within the task at hand. The description of the table structure was provided to the language models in PlantUML format, ensuring a standardized, clear and unambiguous representation of the input data. Based on the modeling results, ER diagrams were also constructed in PlantUML format. The experiment confirms the effectiveness of LLMs in reconstructing missing foreign keys and shows potential for automated analysis, documentation, and improvement of existing databases. Following consistent naming conventions during schema design significantly simplifies both the work of developers and the automated processing of database structures by intelligent systems, playing a crucial role in these processes.

**Keywords:** Entity Relationship Diagram (ERD); PlantUML; Automatization; Relational Databases; Large Language Models (LLMs); ChatGPT-4o; Claude 3.7.

## INTRODUCTION

In modern software engineering, SQL databases are frequently developed or migrated without an explicit specification of the structural relationships between entities—namely, foreign keys. These constraints are critical for preserving data integrity and for constructing Entity-Relationship (ER) models, yet they are often missing or only partially defined. This is commonly due to legacy system limitations, oversight during development, or a lack of expertise. The absence of foreign key constraints can result in several problems:

Increased difficulty in understanding the schema, particularly for new developers or analysts.

Higher risk of introducing data inconsistency during manipulation.

Inability to generate complete ER diagrams automatically.

Decreased effectiveness of tools that rely on well-defined relationships,as automated code generation.

Conventional analysis methods rely solely on explicitly defined constraints and cannot infer hidden or implicit relationships between tables. However, such relationships often exist and may be reconstructed by analyzing column names, data types, naming patterns, and the overall semantics of tables.

With the rise of large language models (LLMs), such as GPT [1], Claude [2], LLaMA [3], and others, it has become possible to leverage their contextual awareness and semantic understanding to reconstruct implicit relationships within SQL schemas. In contrast to traditional algorithms, LLMs can take into account linguistic context and flexibly adapt to various schema writing styles.

Furthermore, the adoption of textual modeling tools like PlantUML [4] and Mermaid [5]—as noted in a study [6] of over 13,000 open-source repositories—reflects a growing shift toward programmatically generated diagrams.

Text-based representations offer better integration with version control systems and facilitate automation in software documentation, eliminating the need for visual editing [7]. The integration of LLMs with tools like PlantUML opens the door to advanced modeling workflows.

There is also a growing interest in the application of artificial intelligence to software development [8], particularly in the area of UML diagram modeling [9]. In their study [10], Rouabhia and Hadjadj proposed a methodology for using ChatGPT to automatically enhance UML class diagrams based on natural language use case descriptions structured in tabular form. Their approach employs PlantUML for diagram visualization and demonstrates the effectiveness of integrating LLMs into modeling processes. Härer [11] introduced the concept of a conceptual model interpreter that leverages LLMs to generate and visualize models in PlantUML and Graphviz formats based on natural language descriptions. This approach enables interactive creation and refinement of models through dialogue. Conrardy and Cabot [12] explore the use of LLMs for transforming photos of hand-drawn UML diagrams—such as those created on whiteboards—into formal models. The authors emphasize that PlantUML syntax is particularly well-suited for this task, as it is most effectively interpreted by LLMs. The paper by Kanuka et al. [13] addresses the challenge of bidirectional traceability between source code and UML design using the GPT-4 LLM model. The authors demonstrate the model's ability to generate UML diagrams from code and vice versa, maintaining consistency between artifacts. PlantUML is used as an intermediate format, which confirms its effectiveness for integration with LLMs. The study [14] presents a multi-agent platform that spans all stages of the Software Development Life Cycle (SDLC), including architectural design. At this stage, an LLM-based agent generates UML diagrams in PlantUML format by transforming user requirements into an architectural representation. The authors also note that the generated UML diagrams serve as a basis for subsequent code generation. This integration of PlantUML and LLMs demonstrates the potential for an automated transition from requirements to technical implementation.

## PROBLEM STATEMENT

Database schemas frequently have structural deficiencies such as the absence of foreign keys, while available tools offer insufficient assistance to engineers in constructing consistent and high-quality database models. The goal of this study is to develop an approach to restore structural relationships between entities in SQL schemas in the absence of foreign keys or when they are incomplete by using large language models (LLMs) and generating ER diagrams in the PlantUML format.

## RESULTS AND DISCUSSION

The study [15] analyzes the capabilities of PlantUML for creating ER diagrams and proposes improvements for their quality and automation potential. The tool Sqlant [16] is introduced for automatically generating PUML code from a PostgreSQL database, which significantly simplifies schema construction. A PlantUML library for describing SQL entities [17] is proposed to improve code readability and diagram maintainability. Furthermore, [15] outlines a sequence of steps utilizing Sqlant and PlantUML to automatically generate ER diagrams.

In this work, we extend the previously described workflow by adding an additional step to interact with an LLM (Fig. 1), which is responsible for foreign key reconstruction. The implementation relies on Sqlant version 0.5.0.

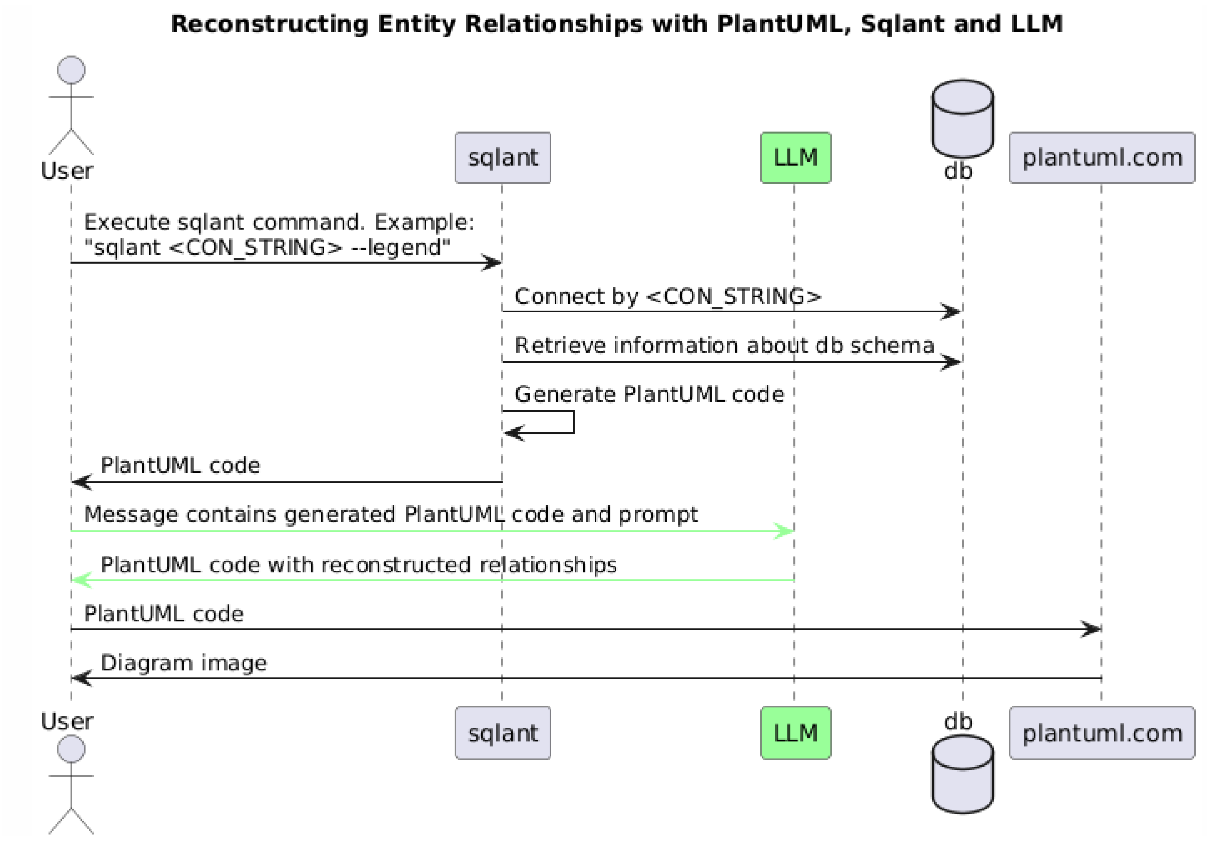
Reconstructing Entity Relationships with PlantUML, Sqlant and LLM
User
sqlant
LLM
db
plantuml.com
Execute sqlant command. Example: "sqlant <CON_STRING> --legend"
Connect by <CON_STRING>
Retrieve information about db schema
Generate PlantUML code
PlantUML code
Message contains generated PlantUML code and prompt
PlantUML code with reconstructed relationships
PlantUML code
Diagram image
User
sqlant
LLM
db
plantuml.com

*Fig. 1. Sequence diagram for reconstructing entity relationships with PlantUML, Sqlant, and LLMs*

The authors created a unique database design in order to test the capability of an LLM to restore foreign keys based on textual descriptions. The database contains 11 tables and 11 foreign keys, as shown in Fig. 2. During the creation of the test database no open resources or LLMs were used. This is important to ensure that the LLM has no prior knowledge about the database structure. Otherwise, there is a risk that the LLM may simply reproduce already known templates.

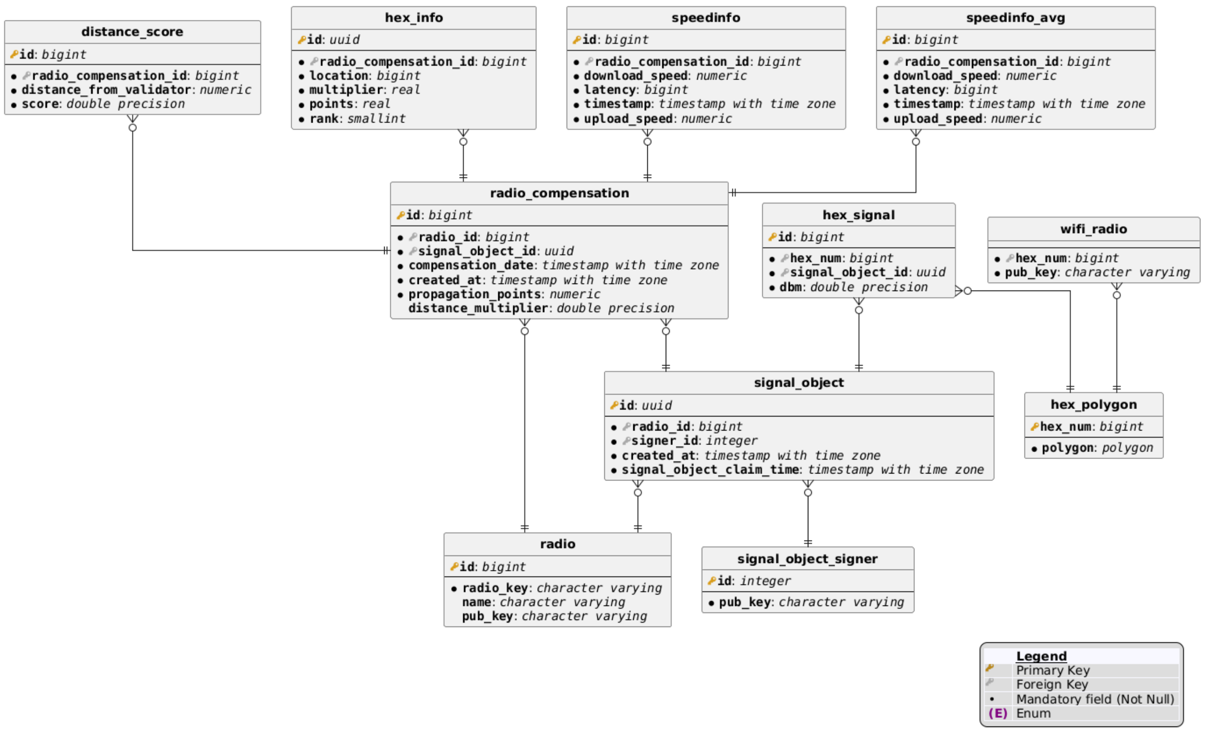

*Fig. 2. The initial database schema*

Here "(PlantUML code)" refers to the PlantUML code of incomplete schema generated by the Sqlant.

All foreign keys have been removed to prepare input data for the LLM, which is shown in Fig. 3. After data preparation was completed, the steps described in Fig. 1 were followed. The text below shows how to locally install them.

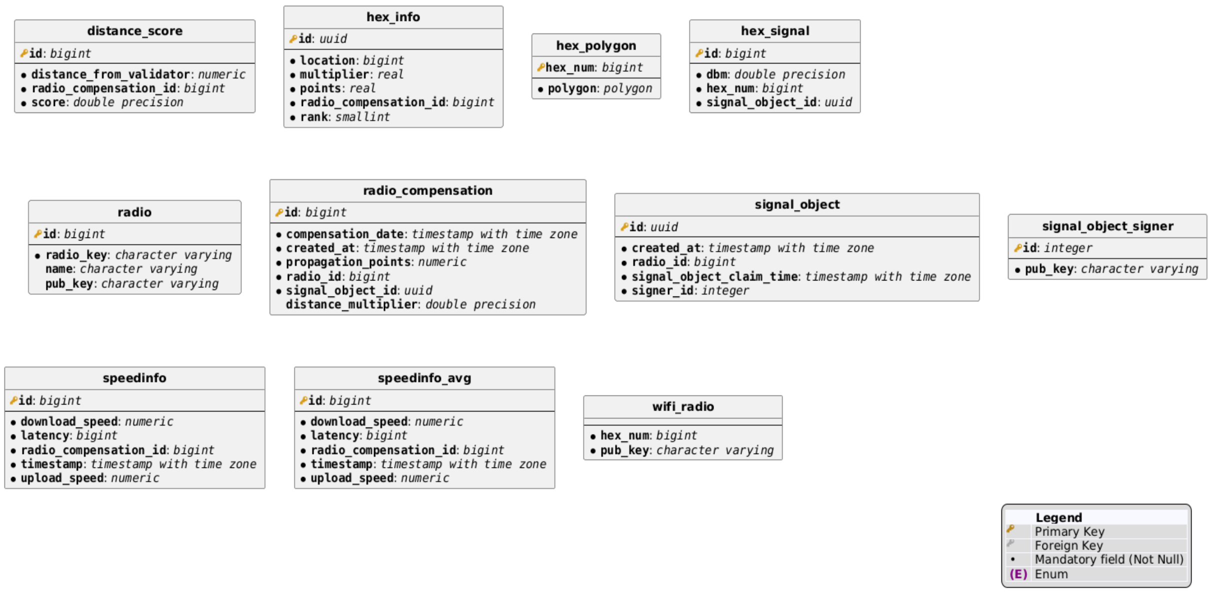

*Fig. 3. Incomplete schema*

The PlantUML code of the incomplete schema was used as input for two LLMs: ChatGPT-4o and Claude 3.7, for analysis. The input data contains the following information about the database schema: table names, field names, field types, and primary keys. No data examples or metadata were provided. The prompt is as follows:

*This is PlantUML code that describes a DB schema (generated by the sqlant tool).*
*Add any potentially missing foreign keys to this code.*
*Use Information Engineering-style relationships in the format: <table_name> <type_of_relationship> <table_name>.*

*Insert them into the provided PlantUML code.*

*(PlantUML code)*

The LLMs proposed a list of foreign keys between tables following the format provided in the prompt. The ChatGPT-4o model successfully reconstructed all 11 foreign keys (Fig. 4), but it failed to correctly identify the relationship types and labeled all of them as one-to-one.

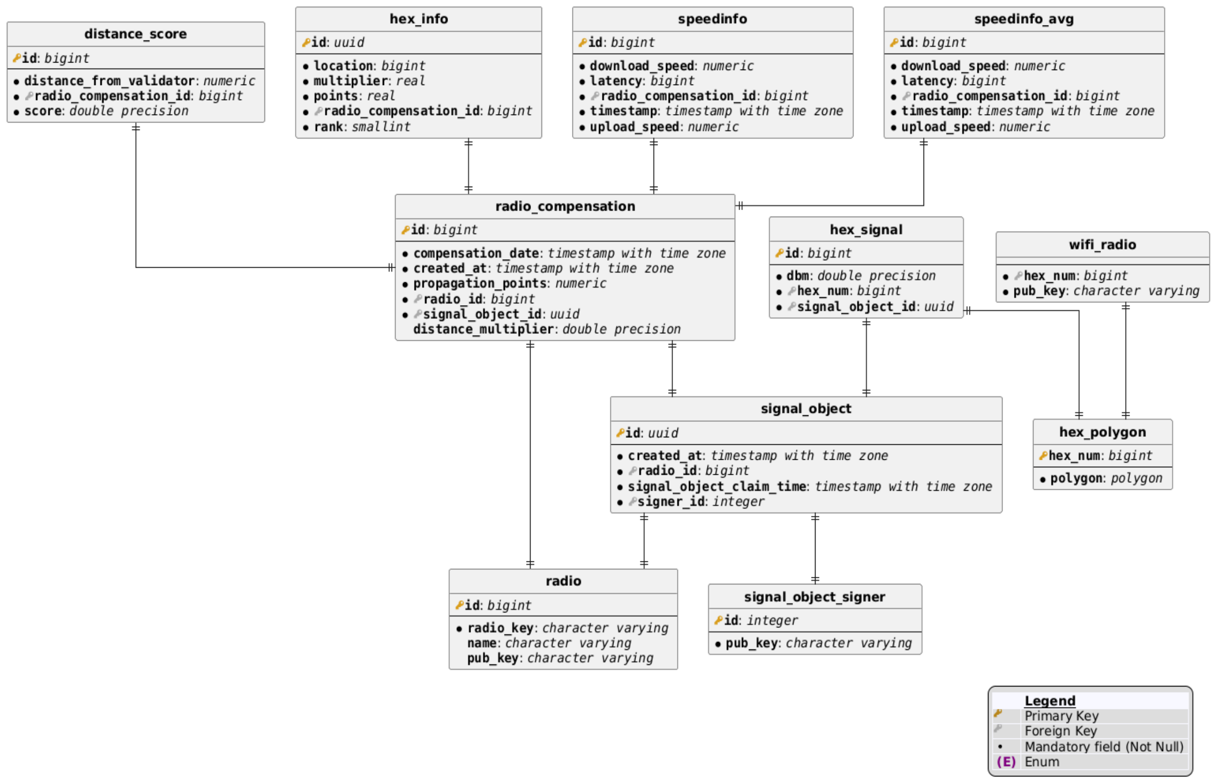

*Fig. 4. Foreign keys restored by ChatGPT-4o*

Claude 3.7 Sonnet produced results (Fig. 5) with foreign keys and relationship types that matched the original database schema, regardless of the field order, which did not affect correctness. Thus, Claude 3.7 Sonnet demonstrates the capability for logical analysis of a database schema and the identification of hidden relationships without any prior information about the specific schema.

Probably, the key role in the restoration of relationships was played by the naming of tables and fields. Due to the use of naming conventions (for example including suffixes "_id", shared roots in field and table names) the LLM was able to identify semantic correspondences between columns and tables.This indicates a high sensitivity of LLMs to linguistic patterns embedded by developers during database design, and at the same time highlights the importance of adhering to consistent naming conventions—particularly when designing database schemas that should be understandable to both humans and machine learning models.

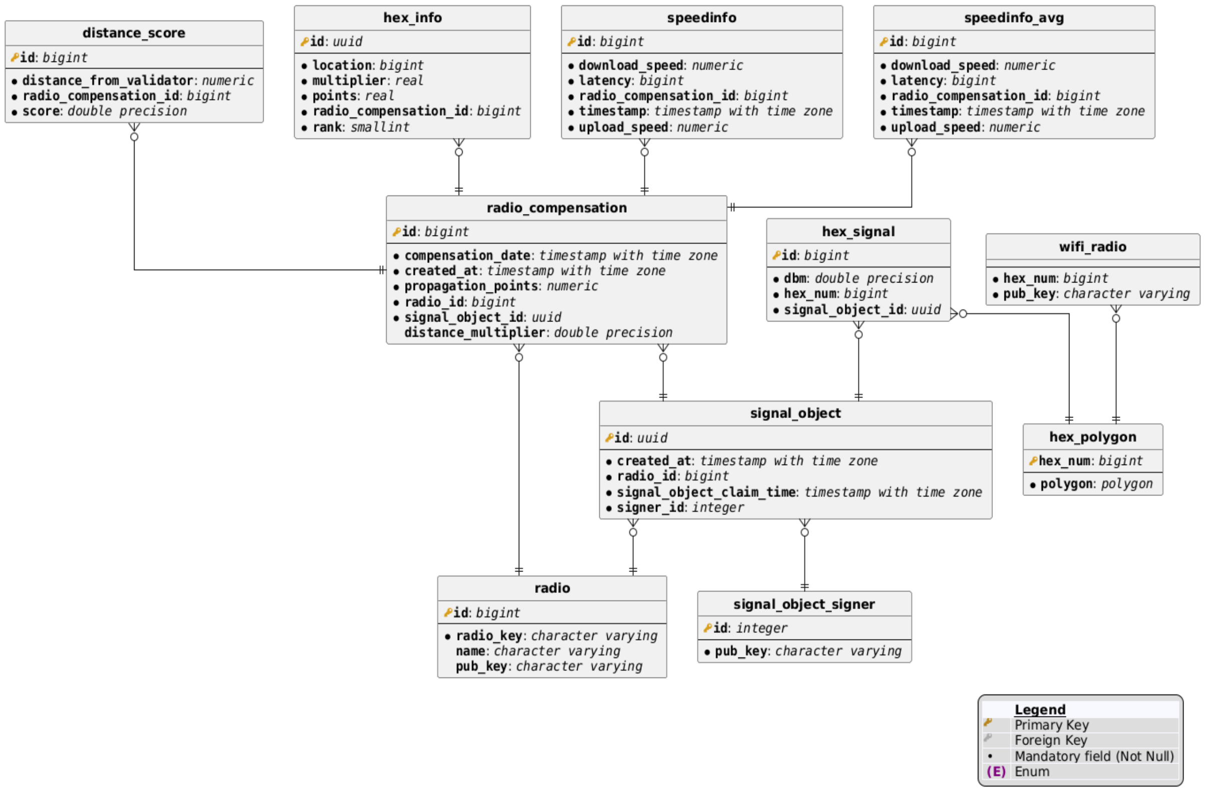

*Fig. 5. Foreign keys restored by Claude 3.7 Sonnet*

## CONCLUSIONS

This study demonstrates the potential of large language models (LLMs) for the automatic reconstruction of structural relationships between entities in SQL database schemas where foreign keys are missing or undefined. The proposed approach integrates with the ER diagram generation process supported by the sqlant tool and extends its functionality by involving an LLM for semantic schema analysis.

The conducted experiment showed that even in the complete absence of explicitly defined relationships, the LLMs were able to reconstruct them based solely on table structure, field names, and logical correspondences. The ChatGPT-4o model successfully identified all 11 relationships but failed to correctly classify their types, assigning all of them as one-to-one. In contrast, the Claude 3.7 Sonnet model not only identified all relationships but also correctly determined their types, indicating higher accuracy in the context of this task.

**Declaration on Generative AI:** During the preparation of this work, the author used ChatGPT in order to: Grammar and spelling check. After using this tool/service, the authors reviewed and edited the content as necessary and take full responsibility for the publication’s content

## REFERENCES (TRANSLATED AND TRANSLITERATED)

1. OpenAI. GPT-4 Technical Report. OpenAI, arXiv:2303.08774 [cs.CL], 2024. doi: 10.48550/arXiv.2303.08774
2. Anthropic. Introducing Claude, 2023. URL: https://www.anthropic.com/news/introducing-claude
3. Touvron, H., Lavril, T., Izacard, G., et al. LLaMA: Open and Efficient Foundation Language Models. Meta AI, arXiv:2302.13971 [cs.CL], 2023. doi: 10.48550/arXiv.2302.13971
4. PlantUML, 2025. URL: https://plantuml.com/.
5. Mermaid | Diagramming and charting tool. 2025. URL: https://mermaid.js.org/
6. J. Romeo, M. Raglianti, C. Nagy and M. Lanza, "UML is Back. Or is it? Investigating the Past, Present, and Future of UML in Open Source Software," in 2025 IEEE/ACM 47th International Conference on Software Engineering (ICSE), Ottawa, ON, Canada, 2025, pp. 692-692, doi: 10.1109/ICSE55347.2025.00155.
7. Terrastruct, Text to diagram, 2025. URL: https://text-to-diagram.com/.
8. Feras A. Batarseh, Rasika Mohod, Abhinav Kumar, Justin Bui, 10 - The application of artificial intelligence in software engineering: a review challenging conventional wisdom. Data Democracy (2020) 179-232. doi: 10.1016/B978-0-12-818366-3.00010-1
9. Javier Cámara, Javier Troya, Lola Burgueño, Antonio Vallecillo, On the assessment of generative AI in modeling tasks: an experience report with ChatGPT and UML. Software and Systems Modeling 22 (2023), 781–793. doi: 10.1007/s10270-023-01105-5
10. D. Rouabhia, I. Hadjadj, Enhancing Class Diagram Dynamics: A Natural Language Approach with ChatGPT, arXiv:2406.11002v1 [cs.SE], 2024. doi: 10.48550/arXiv.2406.11002.
11. Härer, Felix, Conceptual model interpreter for large language models. arXiv:2311.07605 [cs.SE], 2023. doi:10.48550/arXiv.2311.07605.
12. Conrardy, Aaron, and Jordi Cabot, From image to uml: first results of image based uml diagram generation using llms. arXiv:2404.11376 [cs.SE], 2024. doi:10.48550/arXiv.2404.11376
13. Hideyuki Kanuka, Genta Koreki, Ryo Soga, Kazu Nishikawa, Exploring the chatgpt approach for bidirectional traceability problem between design models and code. arXiv:2309.14992, 2023. doi: 10.48550/arXiv.2309.14992
14. Malik Abdul Sami, Muhammad Waseem, Zeeshan Rasheed, Mika Saari, Kari Systä, Pekka Abrahamsson. Experimenting with multi-agent software development: Towards a unified platform. arXiv:2406.05381, 2024. doi:10.48550/arXiv.2406.05381
15. O. Kurotych, L. V. Bulatetska, Optimizing the process of ER diagram creation with PlantUML, CEUR Workshop Proceedings (2025) 47–57. https://cssesw.easyscience.education/cssesw2024/CSSESW2024/paper12.pdf
16. Kurotych, GitHub - kurotych/sqlant: Generate PlantUML/Mermaid ER diagram textual description from SQL connection string, 2024. URL: https://github.com/kurotych/sqlant.
17. Kurotych, A. (2024). db_ent.puml – PlantUML library for database entities. GitHub. Retrieved April 23, 2025, from https://github.com/kurotych/sqlant/blob/6c4a5030dfade0731b65e33f1b5f16595d0229c0/puml-lib/db_ent.puml

**Куротич Анатолій Олександрович**
студент факультету інформаційних технологій і математики
Волинський національний університет імені Лесі Українки, Луцьк, Україна
ORCID ID: 0009-0006-8186-4063
*akurotych@gmail.com*

**Булатецька Леся Віталіївна**
к. ф.-м. н., доцент, доцент кафедри комп'ютерних наук та кібербезпеки
Волинський національний університет імені Лесі Українки, Луцьк, Україна
ORCID ID: 0000-0002-7202-826X
*Bulatetska.Lesya@vnu.edu.ua*

**Онищук Оксана Олександрівна**
к.т.н., доцент, доцент кафедри комп'ютерних наук та кібербезпеки
Волинський національний університет імені Лесі Українки, Луцьк, Україна
ORCID ID: 0000-0002-8342-3011
*Onyshchuk.Oksana@vnu.edu.ua*

# ВІДНОВЛЕННЯ ЗВ'ЯЗКІВ МІЖ СУТНОСТЯМИ В СХЕМАХ БАЗ ДАНИХ ЗА ДОПОМОГОЮ PLANTUML ТА LLM

**Анотація.** У роботі досліджуються перспективи використання великих мовних моделей (LLMs) для автоматичного відновлення зв'язків між таблицями в SQL-базах даних з неповністю визначеними зовнішніми ключами. Для оцінки спроможності LLM-моделі відтворювати зовнішні ключі на основі текстового опису структури таблиць була сформована експериментальна база даних. Схема бази даних без зв'язків була подана на вхід двом великим мовним моделям ChatGPT-4o та Claude 3.7 Sonnet. Для аналізу LLMs було надано лише базову інформацію: назви таблиць, назви полів і первинні ключі — без жодних прикладів даних. Модель ChatGPT-4o успішно виявила всі зв'язки між таблицями, проте продемонструвала обмеження у визначенні типів цих зв'язків: усі вони були класифіковані як «один до одного», незалежно від їх фактичної природи. Це свідчить про недостатню здатність моделі точно інтерпретувати семантику реляційних зв'язків на основі текстового опису. Натомість модель Claude 3.7 Sonnet не лише коректно ідентифікувала всі наявні зв'язки, а й правильно визначила їх типи (наприклад, «один до багатьох»), що демонструє вищу точність і глибше розуміння структури бази даних у рамках поставленого завдання. Опис структури таблиць подавався мовним моделям у форматі PlantUML, що забезпечило стандартизоване, чітке та однозначне представлення вхідних даних для обробки. На основі результатів моделювання були побудовані ER-діаграми також у форматі PlantUML. Експеримент підтверджує ефективність LLMs у реконструкції відсутніх зовнішніх ключів та демонструє їхній потенціал для автоматизованого аналізу, документації та вдосконалення наявних баз даних. Дотримання послідовних правил іменування під час проєктування схеми суттєво спрощує як роботу розробників, так і автоматизовану обробку структур баз даних інтелектуальними системами, відіграючи ключову роль у цих процесах.

**Ключові слова:** ERD-діаграма; PlantUML; автоматизація; реляційні бази даних; великі мовні моделі (LLMs); ChatGPT-4o; Claude 3.7.

## СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. OpenAI. GPT-4 Technical Report. OpenAI, arXiv:2303.08774 [cs.CL], 2024. doi: 10.48550/arXiv.2303.08774
2. Anthropic. Introducing Claude, 2023. URL: https://www.anthropic.com/news/introducing-claude
3. Touvron, H., Lavril, T., Izacard, G., et al. LLaMA: Open and Efficient Foundation Language Models. Meta AI, arXiv:2302.13971 [cs.CL], 2023. doi: 10.48550/arXiv.2302.13971
4. PlantUML, 2025. URL: https://plantuml.com/.
5. Mermaid | Diagramming and charting tool. 2025. URL: https://mermaid.js.org/

6. J. Romeo, M. Raglianti, C. Nagy and M. Lanza, "UML is Back. Or is it? Investigating the Past, Present, and Future of UML in Open Source Software," in 2025 IEEE/ACM 47th International Conference on Software Engineering (ICSE), Ottawa, ON, Canada, 2025, pp. 692-692, doi: 10.1109/ICSE55347.2025.00155.
7. Terrastruct, Text to diagram, 2025. URL: https://text-to-diagram.com/.
8. Feras A. Batarseh, Rasika Mohod, Abhinav Kumar, Justin Bui, 10 - The application of artificial intelligence in software engineering: a review challenging conventional wisdom. Data Democracy (2020) 179-232. doi: 10.1016/B978-0-12-818366-3.00010-1
9. Javier Cámara, Javier Troya, Lola Burgueño, Antonio Vallecillo, On the assessment of generative AI in modeling tasks: an experience report with ChatGPT and UML. Software and Systems Modeling 22 (2023), 781–793. doi: 10.1007/s10270-023-01105-5
10. D. Rouabhia, I. Hadjadj, Enhancing Class Diagram Dynamics: A Natural Language Approach with ChatGPT, arXiv:2406.11002v1 [cs.SE], 2024. doi: 10.48550/arXiv.2406.11002.
11. Härer, Felix, Conceptual model interpreter for large language models. arXiv:2311.07605 [cs.SE], 2023. doi:10.48550/arXiv.2311.07605.
12. Conrardy, Aaron, and Jordi Cabot, From image to uml: first results of image based uml diagram generation using llms. arXiv:2404.11376 [cs.SE], 2024. doi:10.48550/arXiv.2404.11376
13. Hideyuki Kanuka, Genta Koreki, Ryo Soga, Kazu Nishikawa, Exploring the chatgpt approach for bidirectional traceability problem between design models and code. arXiv:2309.14992, 2023. doi: 10.48550/arXiv.2309.14992
14. Malik Abdul Sami, Muhammad Waseem, Zeeshan Rasheed, Mika Saari, Kari Systä, Pekka Abrahamsson. Experimenting with multi-agent software development: Towards a unified platform. arXiv:2406.05381, 2024. doi:10.48550/arXiv.2406.05381
15. O. Kurotych, L. V. Bulatetska, Optimizing the process of ER diagram creation with PlantUML, CEUR Workshop Proceedings (2025) 47–57. https://cssesw.easyscience.education/cssesw2024/CSSESW2024/paper12.pdf
16. Kurotych, GitHub - kurotych/sqlant: Generate PlantUML/Mermaid ER diagram textual description from SQL connection string, 2024. URL: https://github.com/kurotych/sqlant.
17. Kurotych, A. (2024). db_ent.puml – PlantUML library for database entities. GitHub. Retrieved April 23, 2025, from https://github.com/kurotych/sqlant/blob/6c4a5030dfade0731b65e33f1b5f16595d0229c0/puml-lib/db_ent.puml