Uma Abordagem Baseada em Grafos de Conhecimento para Consulta a Bancos de Dados Relacionais em Linguagem Natural

Robson A. Campêlo[1], **Alberto H. F. Laender**[1], **Altigran S. da Silva**[2]

Departamento de Ciência da Computação
Universidade Federal de Minas Gerais (UFMG)
Belo Horizonte – MG – Brazil

[2]Instituto de Computação
Universidade Federal do Amazonas (UFAM)
Manaus – AM – Brazil

{robson.campelo,laender}@dcc.ufmg.br, alti@icomp.ufam.edu.br

Abstract. *Although Large Language Models (LLMs) have driven advances in natural language interfaces for databases, the NL2SQL task still suffers from semantic ambiguities and complex queries. In this work, we propose the use of Knowledge Graphs (KGs) as a semantic representation of a relational schema, integrated with LLMs to support the generation of SQL queries. The approach includes the automatic construction of KGs, Prompt Engineering strategies and an automatic refinement mechanism based on execution failures. A preliminary evaluation on a subset of 36 domains from the Spider benchmark showed that the proposed approach achieved an execution accuracy of 95.34%. The results indicate the potential of KG–LLM integration to improve robustness in NL2SQL.*

Resumo. *Embora os Grandes Modelos de Linguagem (LLMs) tenham impulsionado avanços em interfaces em linguagem natural para bancos de dados, a tarefa NL2SQL ainda sofre com ambiguidades semânticas e consultas complexas. Neste trabalho, propomos o uso de Grafos de Conhecimento (GCs) como representação semântica de um esquema relacional, integrados a LLMs para apoiar a geração de consultas SQL. A abordagem inclui a construção automática de GCs, estratégias de Prompt Engineering e um mecanismo de refinamento automático baseado em falhas de execução. Em uma avaliação preliminar sobre um subconjunto de 36 domínios do benchmark Spider, a abordagem alcançou 95,34% de acurácia de execução. Os resultados indicam o potencial da integração GCs-LLMs para aumentar a robustez em NL2SQL.*

1. Introdução

Interfaces em Linguagem Natural (*Natural Language Interfaces*) para Bancos de Dados (BDs) vêm recebendo crescente atenção, particularmente devido à sua capacidade em prover aos usuários uma experiência de interação mais simples e intuitiva com um BD por meio de consultas em Linguagem Natural (LN). Essa forma de interação dispensa que os usuários tenham necessariamente conhecimento acerca da complexidade de uma linguagem estruturada de consulta, bem como sobre a compreensão de detalhes técnicos

relativos ao esquema do BD considerado [Kim et al. 2020]. Nesse contexto, a tarefa de geração de consultas SQL a partir de sentenças em LN (*NL2SQL*) pode ser considerada como um problema de tradução, semelhante à *Machine Translation*, porém com complexidade adicional, pois exige não apenas a conversão linguística, mas também a compreensão semântica da consulta e seu mapeamento ao esquema do BD.

Assim, a tarefa *NL2SQL* consiste em gerar consultas SQL corretas a partir de sentenças em LN. Nos últimos anos, abordagens baseadas em Modelos de Linguagem Pré-treinados e, mais recentemente, em Grandes Modelos de Linguagem (*Large Language Models* – LLMs), incluindo modelos da série GPT [Dong et al. 2023, Pourreza and Rafiei 2023, Gao et al. 2024], têm alcançado avanços expressivos em tarefas *NL2SQL*, especialmente no *benchmark* Spider [Yu et al. 2018], com destaque para métodos como o *DAIL-SQL* [Gao et al. 2024], que alcança 86,6% de acurácia de execução no *benchmark* Spider. Ainda assim, persistem limitações relacionadas à geração de consultas complexas, ao tratamento de subconsultas e à generalização para diferentes esquemas de banco de dados. Nesse contexto, o desempenho dos LLMs depende fortemente das informações fornecidas, conforme o paradigma de *Prompt Engineering* [Liu et al. 2024].

Diante desse cenário, este artigo propõe uma abordagem *NL2SQL* centrada em LLMs, na qual um Grafo de Conhecimento (GC) [Fensel et al. 2020], derivado automaticamente do esquema do BD, é incorporado ao *prompt* por meio de evidências semânticas estruturadas, combinado a um mecanismo iterativo de refinamento baseado em erros de execução. Embora GCs já tenham sido empregados em diferentes sistemas de consulta e integração de dados, seu uso como mecanismo de enriquecimento semântico do *prompt* de LLMs para tarefas *NL2SQL* ainda é pouco explorado na literatura. Diferentemente do nosso trabalho anterior [Campêlo et al. 2023], baseado em regras e técnicas de NLP, a geração de consultas SQL passa a ser conduzida por LLMs, com o GC fornecendo contexto semântico ao *prompt* e um mecanismo de refinamento iterativo orientando a autocorreção das consultas, inspirado em estratégias de *self-refinement* [Xu et al. 2025].

Em abordagens recentes, como a proposta de [Nascimento et al. 2026], o GC atua como uma camada intermediária de recuperação de dados. Em contraste com essa abordagem, utilizamos o GC diretamente no *prompt* no intuito de prover evidências estruturadas e, assim, guiar a geração da consulta SQL pelo LLM. Nesse sentido, a hipótese central deste trabalho é que evidências semânticas estruturadas derivadas de um GC podem fornecer contexto adicional para LLMs em tarefas *NL2SQL*. Quando incorporadas ao *prompt*, essas evidências podem melhorar a geração de consultas SQL. Com base nessa hipótese, este trabalho contribui em três aspectos principais: (i) a construção automática de um GC a partir do esquema relacional e de exemplos de valores dos atributos; (ii) a utilização de evidências semânticas derivadas desse grafo para enriquecer os *prompts* fornecidos ao LLM; e (iii) uma arquitetura modular que integra geração do GC, tradução NL2SQL e refinamento iterativo baseado em erros de execução.

2. Abordagem Proposta

Esta seção apresenta a abordagem proposta, que contempla duas tarefas principais: (1) a geração automática de um GC em *RDF* (*Resource Description Framework*) a partir do esquema relacional considerado e (2) a geração de consultas SQL a partir de sentenças em LN, com base na semântica do GC. A Figura 1 apresenta a arquitetura da nossa abordagem, composta por quatro módulos: NLP, Interpretador, Tradutor e Executor. Esses

módulos formam um *pipeline* que processa a consulta em LN, constrói o mapeamento semântico, gera a consulta SQL por meio de um LLM e a executa no BD para validação.

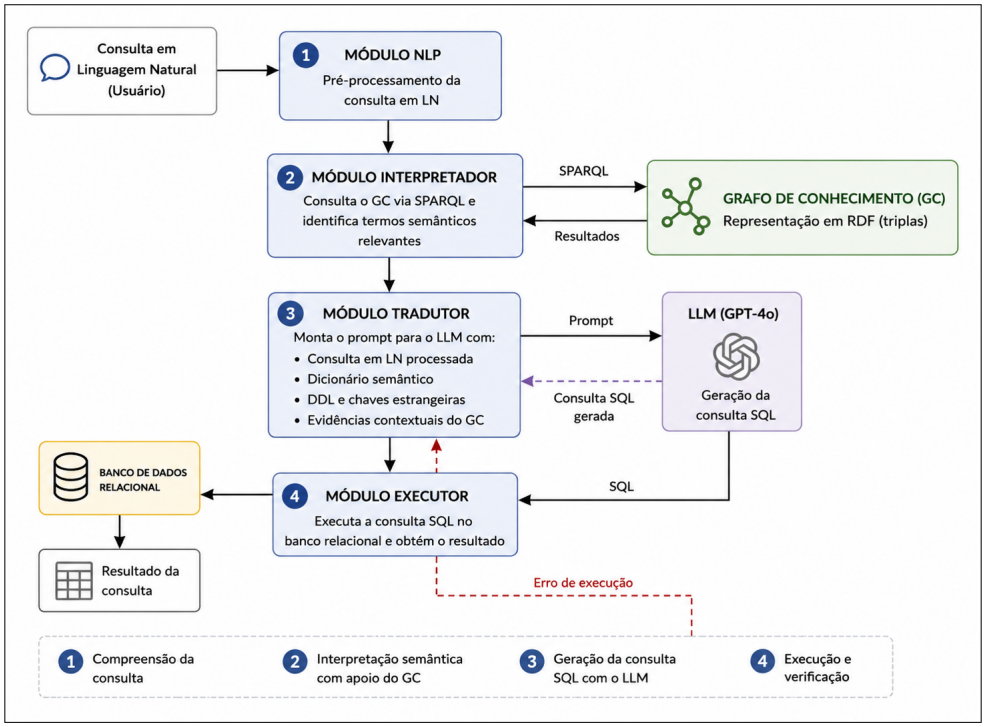

Figura 1. Arquitetura da abordagem de geração de consultas *NL2SQL*.

O GC é gerado automaticamente a partir do esquema relacional do BD por meio da extração de metadados estruturais e exemplos de tuplas. Inicialmente, são recuperadas informações sobre tabelas e colunas a partir dos metadados disponíveis no esquema, formando o vocabulário estrutural do grafo. Em seguida, são incorporados exemplos de valores das tabelas como evidências semânticas. Esses elementos são organizados em um grafo cujos nós representam entidades do esquema e valores, enquanto as arestas representam atributos e relações, incluindo aquelas definidas por chaves estrangeiras. Por fim, a estrutura é convertida em triplas RDF, nas quais tabelas atuam como *sujeitos*, atributos como *predicados* e valores como *objetos*. Por exemplo, para uma tabela *author*, o GC pode conter triplas RDF associando atributos e valores observados, tais como *(author, name, "John Smith")* e *(author, aid, "5")*, bem como relações estruturais derivadas de chaves estrangeiras, como *(domain_author.aid, hasForeignKey, author.aid)*.

Dessa forma, o GC combina informações estruturais do esquema relacional com evidências semânticas derivadas dos registros armazenados, permitindo capturar o vocabulário essencial utilizado no mapeamento entre termos das consultas em LN e elementos do BD. Uma vez construído, o GC desempenha papel central na abordagem proposta, fornecendo contexto semântico estruturado para a geração de consultas SQL por meio de LLMs. Conforme ilustrado na Figura 1, a abordagem *NL2SQL* proposta inicia-se com o processamento linguístico e a interpretação semântica da consulta em LN, os quais são conduzidos pelos módulos NLP e Interpretador. O Módulo NLP realiza a análise linguística da consulta, cujos resultados subsidiam o Módulo Interpretador, responsável por mapear semanticamente os termos da consulta aos elementos do GC, por meio de con-

sultas *SPARQL* e do uso de recursos léxicos para identificação de sinônimos. Como saída, o módulo produz um dicionário semântico que explicita o mapeamento entre expressões da sentença em LN e componentes do esquema relacional (tabelas, colunas e valores).

Na próxima etapa, o Módulo Tradutor constrói um *prompt* estruturado para o LLM, combinando os seguintes componentes: (1) a consulta em LN, (2) o dicionário semântico derivado do GC, (3) a definição do esquema relacional (DDL), (4) as chaves estrangeiras e (5) um conjunto de evidências contextuais inferidas a partir do GC, que explicitam relações semânticas e junções necessárias. Por exemplo, uma evidência contextual pode explicitar que as tabelas *author* e *publication* devem ser associadas por meio da tabela *writes*, a partir dos atributos *aid* e *pid*, auxiliando o LLM na geração de junções semanticamente adequadas. Essa estratégia fundamenta-se em princípios de *In-Context Learning* e *Instruction Prompt Design*, explorando o uso de estruturas de *prompt* que orientam a geração de consultas SQL sintaticamente válidas e semanticamente coerentes, conforme abordagens recentes baseadas em LLMs [Zhang et al. 2023].

Por fim, o Módulo Executor processa no BD a consulta SQL gerada pelo LLM. Em caso de falha de execução, a abordagem emprega um mecanismo de refinamento iterativo baseado no retorno de erros de execução, permitindo que a consulta SQL gerada seja corrigida em iterações sucessivas [Xu et al. 2025]. Nesse processo, a mensagem de erro retornada pelo BD é incorporada a um novo *prompt*, que é reenviado ao LLM para a correção da consulta. Esse mecanismo de autocorreção visa reduzir erros sintáticos e estruturais, aumentando a robustez do processo de execução da tarefa *NL2SQL*.

Para ilustrar esse mecanismo, consideremos um banco de dados telefônico e a seguinte consulta em LN: “Mostre nomes e telefones de clientes que não possuem endereço”. Em uma primeira tentativa, o LLM pode gerar uma consulta SQL incorreta, resultando em erro de execução devido a uma associação inadequada entre tabelas. Nesse caso, a mensagem de erro retornada pelo BD é incorporada a um novo *prompt* e reenviada ao modelo LLM. Com base nesse retorno e no contexto semântico fornecido pelo GC, o LLM pode corrigir automaticamente a consulta em uma nova iteração, produzindo uma consulta SQL válida e semanticamente adequada. Por exemplo, uma associação inicialmente realizada entre tabelas incompatíveis pode ser substituída pelo relacionamento correto identificado a partir das evidências semânticas derivadas do GC. Adicionalmente, a abordagem adota princípios de modularidade e reutilização de código por meio de padrões de projeto, favorecendo a extensibilidade da solução e a substituição de componentes, como LLMs, BDs e mecanismos de consulta ao GC, sem impacto na lógica principal.

3. Resultados Preliminares

Nesta seção, apresentamos uma síntese dos resultados preliminares obtidos com a abordagem proposta. Os experimentos foram conduzidos utilizando o *benchmark* Spider [Yu et al. 2018], amplamente adotado na literatura de *NL2SQL*, considerando, nesta versão preliminar, um subconjunto aleatório de 36 domínios, totalizando 1202 consultas distribuídas em diferentes áreas de aplicação. Os resultados são apresentados em três etapas: (1) descrição do ambiente experimental e do subconjunto do *benchmark* utilizado, (2) avaliação quantitativa por meio das métricas *Execution Accuracy* (EX) e *Exact Match* (EM), e (3) análise qualitativa dos principais tipos de erro observados na geração das consultas SQL.

Adotamos o GPT-4o como modelo experimental, pertencente à série GPT da OpenAI, configurado com os seguintes parâmetros: *max_tokens* = 2048 (limite máximo de tamanho da resposta), *n* = 1 (uma única saída por requisição) e *temperature* = 0.3 (baixo grau de aleatoriedade), priorizando estabilidade, consistência e controle de custo computacional na geração das consultas SQL [Renze 2024]. O GPT-4o foi selecionado por ser um modelo amplamente adotado na literatura recente de *NL2SQL* e por oferecer acesso estável via API para experimentação. Ressalta-se, contudo, que a possível exposição desse modelo ao *benchmark* Spider durante seu processo de treinamento constitui uma potencial ameaça à validade externa dos resultados, aspecto discutido na Seção 4.

Para execução dos experimentos, foram elaborados *prompts* estruturados, com *template* fixo contendo esquema, mapeamento semântico e instruções explícitas. A avaliação utilizou as métricas *Execution Accuracy* (EX) e *Exact Match* (EM), amplamente adotadas na literatura de *NL2SQL*. A abordagem proposta alcançou 95,34% de EX, correspondendo a 1146 consultas corretas de 1202 avaliadas. Em termos da métrica EM, obteve-se 38,1%, valor consistente com a literatura. A diferença em relação à EX é esperada, pois consultas semanticamente equivalentes podem assumir formas sintáticas distintas, penalizando EM sem afetar EX.

Embora os resultados apresentados sejam preliminares e tenham sido obtidos em um subconjunto de 36 domínios do *benchmark* Spider, a abordagem proposta alcançou 95,34% de acurácia de execução. Como referência de ordem de grandeza, sem pretensão de comparação direta, dado que utilizam o Spider integralmente e configurações experimentais distintas das nossas, métodos recentes reportam: 86,6% de EX para o *DAIL-SQL* [Gao et al. 2024], 85,3% para o *DIN-SQL + GPT-4* [Pourreza and Rafiei 2023] e 82,3% para o *C3 + ChatGPT + Zero-Shot* [Dong et al. 2023]. Os resultados obtidos neste trabalho sugerem o potencial da abordagem proposta e motivam avaliações mais abrangentes em trabalhos futuros.

Uma análise de 63 falhas observadas nas consultas SQL geradas pelo LLM indicou predominância de erros relacionados a mapeamento para o esquema (30,2%) e consultas aninhadas/operações de conjunto (30,2%), seguidos por erros em junções (17,5%). Os 22,1% restantes distribuíram-se entre categorias menos frequentes. Tal distribuição é compatível com resultados reportados na literatura, que indicam maior incidência de erros relacionados ao mapeamento para o esquema e à estruturação de consultas complexas, como subconsultas e operações de conjunto [Pourreza and Rafiei 2023, Dong et al. 2023]. Não foram observadas ocorrências de consultas SQL sintaticamente inválidas, indicando robustez estrutural das consultas geradas.

4. Discussão e Limitações

Os resultados obtidos indicam que a integração de GCs gerados automaticamente a partir de esquemas relacionais com LLMs apresenta potencial para a tarefa *NL2SQL*. A abordagem proposta alcançou 95,34% de acurácia de execução no subconjunto avaliado do *benchmark* Spider (1146/1202 consultas), reforçando a hipótese de que evidências semânticas derivadas do GC podem contribuir para a geração de consultas SQL mais precisas. Adicionalmente, os resultados sugerem que tais ganhos podem ser obtidos sem a necessidade de ajuste fino do modelo, indicando o potencial de estratégias baseadas em *instruction prompting* e *in-context learning*. Do ponto de vista metodológico, a arquitetura modular, baseada em padrões de projeto, favorece a reusabilidade e a substituição de componen-

tes, permitindo sua adaptação a diferentes LLMs, BDs e mecanismos de consulta ao GC. Contudo, a ausência de comparações com outros LLMs e com configurações sem enriquecimento semântico limita conclusões mais amplas sobre a contribuição individual de cada componente da nossa abordagem.

Não obstante os resultados promissores, algumas limitações foram identificadas. Observou-se uma variabilidade significativa na acurácia das consultas SQL geradas entre diferentes grupos de domínios, indicando que a abordagem ainda é sensível à complexidade estrutural dos esquemas e às particularidades semânticas de cada domínio. A análise de erros, mostrada na Seção 3, indica dificuldades persistentes do LLM em lidar com estruturas mais complexas, especialmente subconsultas e junções envolvendo múltiplas tabelas. Além disso, o escopo experimental restringe-se a um subconjunto do Spider, não contemplando integralmente o *benchmark*. Todos os experimentos foram realizados com um único modelo (GPT-4o), o que limita análises comparativas entre diferentes LLMs. Da mesma forma, a ausência de experimentos comparativos com outros LLMs e configurações sem enriquecimento semântico limita a avaliação do ganho incremental proporcionado pelo GC. Por fim, não é possível descartar completamente a eventual exposição prévia do GPT-4o ao *benchmark* Spider durante seu treinamento, o que constitui uma ameaça à validade externa comum em avaliações baseadas em LLMs.

5. Conclusão e Trabalhos Futuros

Este estudo constitui uma investigação inicial da abordagem proposta. Os resultados preliminares obtidos sustentam a hipótese de que o enriquecimento semântico com evidências derivadas de Grafos de Conhecimento pode melhorar a acurácia das consultas SQL geradas por LLMs. Contudo, permanecem desafios relacionados à geração de consultas estruturalmente complexas e à generalização da abordagem para cenários empresariais e bancos de dados de maior diversidade. Como trabalhos futuros, pretendemos (1) realizar estudos de ablação para quantificar a contribuição individual das evidências derivadas do GC, (2) aprimorar as heurísticas de geração de evidências contextuais derivadas do GC e (3) avaliar a abordagem em *benchmarks* mais desafiadores, como o Spider 2.0.

Agradecimentos

Os autores agradecem ao CNPq e à CAPES pelo apoio financeiro recebido por meio de bolsas de pesquisa e de doutorado, bem como aos revisores pelas observações e sugestões apresentadas para este trabalho.

Declaração sobre o uso de IA

O uso de ferramentas de IA generativa foi restrito ao apoio na revisão linguística do manuscrito, utilizando modelos da série GPT disponibilizados pela OpenAI. As contribuições científicas, a concepção da abordagem, os experimentos, a implementação e as análises são de responsabilidade exclusiva dos autores.

Referências

Campêlo, R. A., Laender, A. H. F., and Da Silva, A. S. (2023). Using Knowledge Graphs to Generate SQL Queries from Textual Specifications. In Sales, T. P., Araújo, J., Borbinha, J., and Guizzardi, G., editors, *Advances in Conceptual Modeling*, volume 14319, pages 85–94. Springer Nature Switzerland, Cham. Series Title: Lecture Notes in Computer Science.

Dong, X., Zhang, C., Ge, Y., Mao, Y., Gao, Y., Chen, l., Lin, J., and Lou, D. (2023). C3: Zero-shot Text-to-SQL with ChatGPT. Version Number: 1.

Fensel, D., Şimşek, U., Angele, K., Huaman, E., Kärle, E., Panasiuk, O., Toma, I., Umbrich, J., and Wahler, A. (2020). *Knowledge Graphs: Methodology, Tools and Selected Use Cases*. Springer International Publishing, Cham.

Gao, D., Wang, H., Li, Y., Sun, X., Qian, Y., Ding, B., and Zhou, J. (2024). Text-to-SQL Empowered by Large Language Models: A Benchmark Evaluation. *Proceedings of the VLDB Endowment*, 17(5):1132–1145.

Kim, H., So, B.-H., Han, W.-S., and Lee, H. (2020). Natural language to SQL: where are we today? *Proceedings of the VLDB Endowment*, 13(10):1737–1750.

Liu, X., Shen, S., Li, B., Ma, P., Jiang, R., Zhang, Y., Fan, J., Li, G., Tang, N., and Luo, Y. (2024). A Survey of Text-to-SQL in the Era of LLMs: Where are we, and where are we going? Version Number: 6.

Nascimento, E. R., Avila, C. V. S., Izquierdo, Y. T., García, G. M., Andrade, L. F. L., Silva, M. O., Facina, M. S. P., Lemos, M., and Casanova, M. A. (2026). A Text-to-SQL strategy based on large language models and knowledge graphs for real-world databases. *Data & Knowledge Engineering*, 164:102580.

Pourreza, M. and Rafiei, D. (2023). DIN-SQL: Decomposed In-Context Learning of Text-to-SQL with Self-Correction. In Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., and Levine, S., editors, *Advances in Neural Information Processing Systems*, volume 36, pages 36339–36348. Curran Associates, Inc.

Renze, M. (2024). The Effect of Sampling Temperature on Problem Solving in Large Language Models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 7346–7356, Miami, Florida, USA. Association for Computational Linguistics.

Xu, W., Zhu, H., Yan, L., Liu, C., Han, P., Duan, S., and Pan, J. Z. (2025). TS-SQL: Test-driven Self-refinement for Text-to-SQL. In Christodoulopoulos, C., Chakraborty, T., Rose, C., and Peng, V., editors, *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 2864–2889, Suzhou, China. Association for Computational Linguistics.

Yu, T., Zhang, R., Yang, K., Yasunaga, M., Wang, D., Li, Z., Ma, J., Li, I., Yao, Q., Roman, S., Zhang, Z., and Radev, D. (2018). Spider: A Large-Scale Human-Labeled Dataset for Complex and Cross-Domain Semantic Parsing and Text-to-SQL Task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3911–3921, Brussels, Belgium. Association for Computational Linguistics.

Zhang, H., Cao, R., Chen, L., Xu, H., and Yu, K. (2023). ACT-SQL: In-Context Learning for Text-to-SQL with Automatically-Generated Chain-of-Thought. In Bouamor, H., Pino, J., and Bali, K., editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 3501–3532, Singapore. Association for Computational Linguistics.