Multi-Agent Architecture with RAG and Dynamic Context Windows for Text-to-SQL Optimization

Willgnner Ferreira Santos[1], **Paulo Victor dos Santos**[1]
Marcella Scoczynski Ribeiro Martins[2], **Larissa Freire Lekakis**[1], **Frederico Lemes Rosa**[1]
Bruno Matheus Costa[1], **Miguel Alves Pereira Filho**[1] and **Isabella Alves Montalvão**[1]
[1]SENAI Fatesg College, Applied Artificial Intelligence Core (NIAA), Brazil
[2]Federal University of Technology – Paraná (UTFPR), Brazil
{willgnnerferreira, paulosantos}@fieg.com.br
larissafreire.senai@fieg.com.br, marcella@utfpr.edu.br

Abstract

Natural language interfaces supported by LLMs have been used to translate user questions into SQL queries, but sending the complete database schema in each prompt entails high token consumption and computational cost, especially in corporate databases with hundreds of tables. This work presents a multi-agent Text-to-SQL architecture with dynamic context windows, which combines RAG and metadata dictionaries to select, at query time, only the relevant tables and columns. In a case study with Firebird enterprise databases, the approach reduces by an average of 84.4% the number of processed tokens, resulting in more efficient queries without loss of quality, thereby contributing to the democratization of access to corporate databases.

1 Introduction

The strategic use of information has become consolidated as an important component of organizational competitiveness, insofar as managerial decisions depend on fast and reliable access to data from different sectors (Paletta and Lago, 2021; Nudurupati et al., 2024). In this context, natural language interfaces based on Large Language Models (LLMs) (Manchanda et al., 2024) emerge as a promising alternative to democratize access to relational databases (Fan et al., 2024), allowing users to formulate queries without direct knowledge of Structured Query Language (SQL) (Liu et al., 2025).

LLM-based Text-to-SQL systems have been the subject of several recent studies (Hong et al., 2025; Huang et al., 2025). These studies show that such systems are capable of translating intentions expressed in natural language into complex SQL queries, but they also highlight important challenges: robustness in the face of extensive schemas, data security and privacy, as well as high operational costs resulting from intensive token usage. In real corporate databases, the schema may include hundreds of tables and thousands of columns (Bodensohn et al., 2025); including the complete schema in every prompt leads to context windows close to the model limit, information overload phenomena (Nascimento et al., 2025), and increased financial cost.

This scenario is important in Brazilian organizations that operate legacy Enterprise Resource Planning (ERP) systems in Firebird (Milosavljeva and Jankulovska, 2025; Mahfuz, 2024), with scarce documentation, heterogeneous naming conventions, and queries issued in Portuguese. In such cases, it becomes vital to reduce the amount of schema information sent to the LLM (Parciak et al., 2024), while preserving the semantic context necessary to generate correct SQL.

This work investigates a multi-agent Text-to-SQL architecture with dynamic context windows, aimed at large-scale enterprise databases. The main contributions are: (i) a multi-stage architecture that combines a contextual selection agent, decision graphs, and LLMs to select, at query time, salient tables and columns; (ii) a complete pipeline for the automatic generation of metadata dictionaries in `YAML/JSON` directly from Firebird databases; and (iii) a case study on three real ERP databases in Portuguese, in which the impact of context selection on SQL query accuracy, token consumption, and per-query cost is quantified.

2 Related work

LLM-based Text-to-SQL systems have increasingly incorporated Retrieval-Augmented Generation (RAG) to access external documents, examples, and metadata rather than relying only on parametric knowledge. Recent surveys indicate that this combination is especially useful for handling complex schemas, ambiguous questions, and multi-turn natural language interactions (Hong et al., 2025;

Proceedings of the 17th International Conference on Computational Processing of Portuguese (PROPOR 2026)
Vol. 1, pages 988–993

Mohammadjafari et al., 2024).

Recent approaches have explored different strategies to improve Text-to-SQL performance. AID-SQL introduces difficulty-sensitive instructions and contextual retrieval to adapt query generation to different complexity levels (Li et al., 2025). Another line of work focuses on schema selection or pruning to reduce prompt size while preserving relevant information; for example, MAC-SQL decomposes the process into table selection followed by column selection, showing that hierarchical selection can mitigate context overload (Gao et al., 2024; Wang et al., 2025). In parallel, agent-based approaches investigate architectures in which specialized agents plan, retrieve information, and iteratively refine queries using feedback and heuristics (Santos et al., 2025). Although effective, these studies generally assume benchmark-style schemas and do not address the systematic generation and reuse of metadata dictionaries in large-scale corporate databases.

In summary, the related works do not yet fully address the scenario investigated in this study. Approaches such as AID-SQL and MAC-SQL focus on controlled benchmarks, keeping the schema complete or nearly complete in the prompt and without analyzing the impact of context in enterprise databases with hundreds of tables. Proposals with intelligent agents address the orchestration of stages, but do not explore the systematic generation and reuse of metadata dictionaries for context selection. In contrast, this work proposes a multi-agent Text-to-SQL architecture that combines pre-generated `YAML/JSON` dictionaries with hierarchical selection of tables and columns, specifically designed for large-scale corporate databases.

3 Methodology

Although the number of databases and queries used in this study is limited, this constraint is inherent to the industrial research context and does not undermine the validity of the findings. Access to real ERP data in production environments is restricted by confidentiality agreements and operational risk, making it infeasible to scale experiments arbitrarily. Wohlin et al. (2000) recognize that controlled experiments in industrial software engineering frequently operate with small samples, emphasizing that internal validity-that is, the ability to establish causal relationships between the intervention and the observed outcomes under the same experimental conditions is more critical than sample size when the objective is comparative evaluation rather than population-level generalization.

The choice of ten queries per database was guided by three criteria. First, coverage of complexity levels: the query set spans simple filtering, aggregation with `GROUP BY`, multi-table `JOIN`s, and temporal filters, ensuring that different demands on schema comprehension are represented. Second, saturation of the comparison: since both approaches (full schema and proposed selection) are evaluated on identical queries, the relevant unit of analysis is the within-condition difference, not the absolute number of queries; ten paired observations per database are sufficient to reveal systematic behavioral differences between the methods. Third, alignment with prior Text-to-SQL evaluations in industrial settings: studies such as Nascimento et al. (2025) and Bodensohn et al. (2025) similarly conduct evaluations on restricted real-world databases, noting that benchmark-scale datasets (e.g., Spider, BIRD) do not capture the complexity of legacy enterprise schemas. The objective of this experiment is therefore not to generalize results to all possible scenarios, but to compare the performance of the baseline and the proposed method under the same controlled conditions, following established practice in applied software engineering research (Wohlin et al., 2000).

To enable the efficient operation of the system, we implement a multi-agent architecture responsible for the automatic extraction and selective use of metadata from relational databases. The main agent uses frameworks such as LangChain and LangGraph to perform structural inspection, capturing names, descriptions, and types of tables and columns. This procedure involves connecting to the Firebird database, synthesizing descriptions in Portuguese by means of Claude 3 Sonnet, accessed via AWS Bedrock, and organizing this information into consistent dictionaries.

The resulting dictionaries are stored in structured formats (`YAML/JSON`) for selective reuse across queries. In particular, the files `INFO_TBLS.yaml` and `INFO_COLUMNS.yaml` aggregate, respectively, table and column metadata, serving as the basis for the RAG mechanism. At query time, the system retrieves only the metadata that are important for the user's question, thereby reducing the amount of context sent to the LLM and, consequently, token consumption.

The processing flow, illustrated in Figure 1, is organized into two parts. Offline Phase runs when

the context and description files need to be created or updated. It creates two files containing information from the tables and columns and saves them. The Online Phase acts as the main text-to-SQL flow and starts with the user input as a Natural Language Question, which is used by the Table Selector Agent to disregard unwanted tables for the question and finally injects the question and the pruned context into the SQL Generator Agent.

The architecture is deployed and tested in the production environment of a private Brazilian organization. In compliance with the Brazilian General Data Protection Law (LGPD) (Garcia et al., 2020) and confidentiality agreements, the name of the institution, the solution code, and the raw data are not made publicly available without restriction. Researchers interested in replicating or further developing the study may contact the authors; controlled access to subsets of data, metadata, and code artifacts is available, subject to evaluation by the partner company and, when necessary, to the signing of confidentiality agreements or equivalent instruments.

The evaluation of the methodology combines quantitative and qualitative analyses. From a quantitative perspective, we measure the token reduction and the per-query cost achieved through context selection, in comparison with a baseline that includes the full schema. From a qualitative perspective, we compare the correctness and usefulness of the SQL queries generated in both scenarios, examining the extent to which the agents and the RAG contribute to preventing errors in schema understanding and semantically flawed queries.

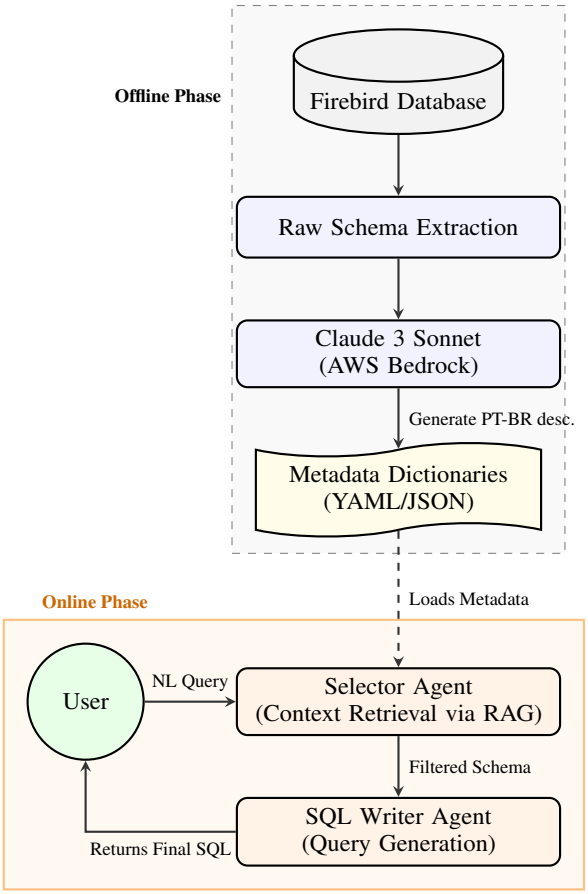

Figure 1: Complete pipeline architecture. The offline phase extracts schema metadata into dictionaries, while the online phase utilizes a multi-agent system to filter context and generate the SQL query.

4 Results

This section presents the experimental results of the proposed architecture in comparison with the baseline using the full schema, evaluating SQL generation accuracy, token consumption, and per-query cost on three real ERP databases in Firebird.

4.1 Experimental setup

To assess the effectiveness of the proposed approach, experiments were conducted with 10 questions in Portuguese applied to 3 distinct databases. The databases correspond to enterprise ERP systems based on Firebird (Njue, 2025), containing between 397 and 448 tables, totaling 13,143 columns (an average of 4,381 columns per database). The model used for SQL generation is Mistral Large, accessed via AWS Bedrock (Bhattacharjee, 2025). The evaluated metrics are: (i) SQL generation accuracy, (ii) number of tokens consumed, and (iii) average cost per query. The baseline corresponds to the traditional approach that sends the complete schema in all prompts, using a fixed context of 32,768 tokens. The 10 queries per database cover a range of complexity levels, including simple filtering, aggregation with GROUP BY, multi-table JOINs, and temporal filters. A query was considered correct if it executed without errors and returned results semantically equivalent to a manually validated reference answer, as verified by two domain experts from the partner organization.

The selection of 10 queries, while appearing quantitatively modest, follows a *purposive sampling* strategy aimed at high-complexity scenarios. In industrial ERP environments, databases undergo decades of natural evolution, leading to a chaotic state characterized by a data redundancy, deprecated tables that remain queryable for legacy

reasons, and semantic shifts in domain columns (Bodensohn et al., 2025). The chosen query set specifically targets these "hard cases" where preliminary agent architectures and baseline models consistently failed due to the inclusion of legacy or redundant entities in the SQL generation.

To ensure the robustness of the findings, each of the 10 queries underwent an extensive iterative refinement process during the architecture's development. This involved more than 20 execution-and-comparison cycles per query against gold-standard SQL statements authored by domain experts.

4.2 SQL generation accuracy

Table 1 presents the accuracy comparison between the baseline approach (full schema) and the proposed approach (with table selection). Contrary to the intuition that more context always improves performance, context reduction not only preserved but actually increased the quality of the generated queries. The proposed approach achieves an average accuracy of 66.7%, whereas the baseline attains only 13.3%, a gain of 53.4% percentage points. This result is consistent with the information overload phenomenon: the excess of irrelevant information in the full schema makes it difficult for the LLM to focus on the elements that are truly necessary, as observed in studies with large-scale databases (Nascimento et al., 2025).

	N	Errors		Accuracy	
DB	**q.**	**Base**	**Prop.**	**Base**	**Prop.**
BD 1	10	9	3	10.0%	70.0%
BD 2	10	7	3	30.0%	70.0%
BD 3	10	10	4	0.0%	60.0%
Total	**30**	**26**	**10**	**13.3%**	**66.7%**

Table 1: SQL generation accuracy.

Note: Accuracy = correct queries / total (N=10 per database). Base = full schema; Prop. = proposed (schema with selection).

In database 3, the most complex one, with 448 tables, the baseline did not produce any correct query, whereas the proposed approach achieved 60% accuracy. This demonstrates that hierarchical context selection is not merely a cost optimization, but an essential component for preserving answer quality in highly complex databases.

Table 2 presents a representative example of a query generated by the proposed approach, illustrating the system's ability to produce correct SQL from a natural language question in Portuguese.

Natural language question (PT-BR):

Quais os 5 produtos mais vendidos no último mês? (What are the five best-selling products in the past month?)

Generated SQL:

```
SELECT p.DESCRICAO, SUM(i.QUANTIDADE) AS TOTAL
FROM ITENS_VENDA i
JOIN PRODUTOS p ON p.ID = i.ID_PRODUTO
WHERE i.DATA_VENDA >=
  DATEADD(-1 MONTH TO CURRENT_DATE)
GROUP BY p.DESCRICAO
ORDER BY TOTAL DESC
ROWS 5;
```

Table 2: Representative SQL query generated by the proposed approach from a natural language question in Portuguese (Database 1).

4.3 Context and cost reduction

Table 3 summarizes the efficiency of the proposed approach in comparison with the baseline. The prior selection of tables via LangGraph (Pourreza and Rafiei, 2023) reduced the average token consumption by 84.4%, which translates into an equivalent saving in the average cost per query. On average, about 27,656 tokens per query are saved relative to the fixed context of 32,768 tokens. In addition, the average number of tables in context dropped from 415 to the range of 4–8, while the average number of columns is reduced from 4,381 to 40–80, corresponding to reductions of 97.6% and 98.2%, respectively. These results are consistent with studies that demonstrate the benefits of schema selection techniques for mitigating context overhead (Gao et al., 2023; Wang et al., 2025).

Metric	**Schema Full**	**With Selection**	**Reduction (%)**
Average tokens/query	32,768	5,112	84.4%
Average cost/query ($)	0.0493	0.0077	84.4%
Tables in context	415	4–8	97.6%
Columns in context (average)	4,381	40–80	98.2%

Table 3: Efficiency comparison between approaches.

Note: Average values computed over 10 test queries on three Firebird databases (397, 400, and 448 tables).

4.4 Analysis by database

Figure 2 presents the detailed analysis of token consumption in each database. It is observed that, with prior selection, consumption ranged between 1,841 and 9,068 tokens, remaining consistently below the fixed limit of 32,768 tokens of the baseline. Database 1 (397 tables) shows an average of 6,151 tokens per query; database 2 (400 tables), 4,669; and database 3 (448 tables), 4,364 tokens,

indicating that the effectiveness of the selection is maintained regardless of the schema size.

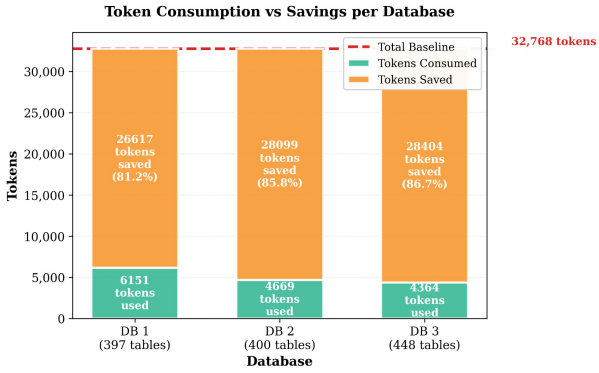

Figure 2: Token savings relative to the baseline of 32, 768 tokens per query.

4.5 Variability and scalability

The analysis reveals substantial variability in token consumption as a function of query complexity, with ranges of up to 326% between simple and complex cases. This dynamic adaptation indicates that the system exploits the context budget intelligently, using more tokens only when necessary. The efficiency gain remains consistent even in extensive schemas (between 397 and 448 tables), with consumption reductions between 81.2% and 86.7%, which confirms the scalability of the approach.

These results reinforce evidence that the performance of Text-to-SQL systems tends to decline in the presence of very large databases (Hong et al., 2025). The strategy of using pre-generated contexts in YAML files, such as `INFO_TBLS.yaml` and `INFO_COLUMNS.yaml`, proves effective for accelerating information selection, avoiding repeated inspections of the schema and adding only minimal overhead to the process.

5 Conclusion

This work presented a multi-agent Text-to-SQL architecture with dynamic context windows, combining RAG, decision graphs, and metadata dictionaries in YAML/JSON to operate on large-scale corporate databases. By selecting, at query time, only the necessary tables and columns, the system significantly reduced the number of processed tokens and the cost per query, while simultaneously increasing SQL generation accuracy compared with the baseline using the full schema. The results of the case study on three real ERP databases indicate that hierarchical context selection is not merely an efficiency optimization, but an important component for maintaining answer quality in scenarios with hundreds of tables and queries in Portuguese. Thus, the proposed methodology provides a concrete foundation for the development of scalable and economically viable Text-to-SQL solutions in production environments.

Limitations

Although the results are promising, this study has important limitations. The evaluation was conducted in a single partner organization using three Firebird databases and only ten test queries, which limits generalization to other domains, database management systems, and usage profiles. However, the within-subjects design, in which both the baseline and the proposed method were tested on the same queries and databases, supports internal validity by indicating that the observed differences in accuracy and token consumption are attributable to the method rather than to query selection bias. The approach also assumes that table and column names are at least minimally informative, which may not hold in highly heterogeneous or inconsistent schemas. In addition, the study does not include LLM fine-tuning or usability evaluations with end users, so aspects such as user experience, response time, and integration into daily workflows remain for future work.

Ethical considerations

The research is conducted in compliance with the Brazilian LGPD, using only data from a controlled corporate environment and under confidentiality agreements with the partner organization. The system incorporates guardrail mechanisms to reduce the risk of leakage of personal and sensitive data in the prompts sent to the models. Nevertheless, it is acknowledged that LLMs are subject to biases and hallucinations; therefore, the SQL queries generated should not be executed automatically in production without specialized human validation. Data and code artifacts derived from this study may be made available in a restricted manner for academic purposes, subject to evaluation by the partner company and the signing of specific confidentiality agreements.

Acknowledgements

The authors thank SENAI Fatesg College for encouraging research and supporting the development

of projects, and the NIAA for carrying out the work and providing the necessary environment and infrastructure.

References

A. Bhattacharjee. 2025. *A Practical Guide to Generative AI Using Amazon Bedrock: Building, Deploying, and Securing Generative AI Applications*. Professional and Applied Computing. Apress.

Jan-Micha Bodensohn, Ulf Brackmann, Liane Vogel, Anupam Sanghi, and Carsten Binnig. 2025. Unveiling challenges for llms in enterprise data engineering. *arXiv preprint arXiv:2504.10950*.

Chongjiong Fan, Zhicheng Pan, Wenwen Sun, Chengcheng Yang, and Wei-Neng Chen. 2024. Latuner: An llm-enhanced database tuning system based on adaptive surrogate model. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 372–388. Springer.

Dawei Gao, Haibin Wang, Yaliang Li, Xiuyu Sun, Yichen Qian, Bolin Ding, and Jingren Zhou. 2023. Text-to-sql empowered by large language models: A benchmark evaluation. *arXiv preprint arXiv:2308.15363*.

Yingqi Gao, Yifu Liu, Xiaoxia Li, Xiaorong Shi, Yin Zhu, Yiming Wang, Shiqi Li, Wei Li, Yuntao Hong, Zhiling Luo, and 1 others. 2024. A preview of xiyan-sql: A multi-generator ensemble framework for text-to-sql. *arXiv preprint arXiv:2411.08599*.

Lara Rocha Garcia, Edson Aguilera-Fernandes, Rafael Augusto Moreno Gonçalves, and Marcos Ribeiro Pereira-Barretto. 2020. *Lei Geral de Proteção de Dados (LGPD): guia de implantação*. Editora Blucher.

Z. Hong, Z. Yuan, Q. Zhang, and 1 others. 2025. Next-generation database interfaces: A survey of llm-based text-to-sql. In *IEEE Journals & Magazine*.

Feiran Huang, Junnan Dong, Hao Chen, and Qinggang Zhang. 2025. Enhancing security in text-to-sql systems: A novel dataset & evaluation framework for prompt injection attacks. *Natural Language Engineering*.

Xiuwen Li, Qifeng Cai, Yang Shu, Chenjuan Guo, and Bin Yang. 2025. Aid-sql: Adaptive in-context learning of text-to-sql with difficulty-aware instruction and retrieval-augmented generation. In *2025 IEEE 41st International Conference on Data Engineering (ICDE)*, pages 3945–3957. IEEE.

Xinyu Liu, Shuyu Shen, Boyan Li, Peixian Ma, Runzhi Jiang, Yuxin Zhang, Ju Fan, Guoliang Li, Nan Tang, and Yuyu Luo. 2025. A survey of text-to-sql in the era of llms: Where are we, and where are we going? *IEEE Transactions on Knowledge and Data Engineering*.

Jorge Motta Mahfuz. 2024. Adoption of enterprise resource planning systems in brazilian small and medium enterprises: Overcoming barriers and maximizing opportunities. Master's thesis, Universidade NOVA de Lisboa (Portugal).

Jiya Manchanda, Laura Boettcher, Matheus Westphalen, and Jasser Jasser. 2024. The open source advantage in large language models (llms). *arXiv preprint arXiv:2412.12004*.

Jovana Milosavljeva and Viktorija Jankulovska. 2025. Report for support in business development. Technical report, CARBONICA Project (Horizon Europe). Deliverable D4.3.

A. Mohammadjafari, A. S. Maida, and R. Gottumukkala. 2024. From natural language to sql: Review of llm-based text-to-sql systems. *arXiv preprint arXiv:2410.01066*.

Eduardo Nascimento and 1 others. 2025. Llm-based text-to-sql for real-world databases. *SN Computer Science*, 6(1):1–18.

Deborah Karimi Njue. 2025. *ENTERPRISE RESOURCE PLANNING SYSTEM INTEGRATION AND PERFORMANCE OF COMMERCIAL BANKS IN EMBU COUNTY, KENYA*. Ph.D. thesis, Kenyatta University.

Sai S Nudurupati, Sofiane Tebboune, Patrizia Garengo, Richard Daley, and Julie Hardman. 2024. Performance measurement in data intensive organisations: resources and capabilities for decision-making process. *Production Planning & Control*, 35(4):373–393.

Francisco Carlos Paletta and Jader Jaime Costa do Lago. 2021. Gestão da informação corporativa. *Environmental smoke*, 4(21):54–64.

Marcel Parciak, Brecht Vandevoort, Frank Neven, Liesbet M Peeters, and Stijn Vansummeren. 2024. Schema matching with large language models: an experimental study. *arXiv preprint arXiv:2407.11852*.

Mohammadreza Pourreza and Davood Rafiei. 2023. Din-sql: Decomposed in-context learning of text-to-sql with self-correction. *Advances in Neural Information Processing Systems*, 36:36339–36348.

F. Santos, L. Almeida, R. Pereira, and 1 others. 2025. Optimizing text-to-sql conversion techniques through the integration of intelligent agents and large language models. In *Information Processing & Management*.

Bing Wang and 1 others. 2025. Mac-sql: A multi-agent collaborative framework for text-to-sql. In *Proceedings of the 2025 International Conference on Computational Linguistics (COLING)*, pages 540–557.

Claes Wohlin, Per Runeson, Martin Höst, Magnus C. Ohlsson, Björn Regnell, and Anders Wesslén. 2000. *Experimentation in Software Engineering: An Introduction*, 1 edition. Kluwer Academic Publishers.