

To Describe or Not to Describe? Benchmarking Database Representations for Schema Linking in Text-to-SQL

Daiane Ucceli Kreitlow and Hilário Tomaz Alves de Oliveira

Postgraduate Program in Applied Computing
Instituto Federal do Espírito Santo (IFES), Serra, Brazil
daianeucceli@gmail.com, hilario.oliveira@ifes.edu.br

Abstract

Text-to-SQL systems aim to translate natural language questions into Structured Query Language (SQL) queries, enabling database access without requiring SQL expertise. In real-world scenarios, these systems often need to manage multiple databases with heterogeneous schemas, making Schema Linking a crucial preliminary step for identifying relevant databases, tables, and columns. This study investigates Schema Linking for questions written in Brazilian Portuguese and compares two schema representation strategies: natural-language descriptions generated by Large Language Models (LLMs) and representations based on Data Definition Language (DDL) and Data Manipulation Language (DML) commands. Experiments conducted on a Brazilian Portuguese version of the Spider dataset, with over 200 databases, evaluated several LLMs and embedding models. The experimental results based on Hit@k show that natural language descriptions consistently outperform DDL/DML-based representations, demonstrating the effectiveness of LLM-generated schema descriptions for Schema Linking tasks.

1 Introduction

Natural Language Interfaces (NLI) to databases enable users to interact with relational data in a natural, intuitive way (Saha et al., 2016a). As digital data is generated and accessed on a massive scale, it has become an important resource for human activities ranging from business operations to scientific research. These data are commonly stored in relational databases managed through the Structured Query Language (SQL) (José and Cozman, 2021; Katsogiannis-Meimarakis and Koutrika, 2023).

The Text-to-SQL task, illustrated in Figure 1, aims to convert questions formulated in natural language into executable SQL queries, allowing both non-experts and advanced users to extract insights from complex datasets (Yu et al., 2021). In real

enterprise environments, this process requires preliminary steps, for instance, identifying the target database elements within an extensive repository of schemas, a task known as Schema Linking (Wang et al., 2025c). Thus, given a natural language question, the first step is to identify which database is needed to answer it, after which its schema can be provided as input to a Text-to-SQL system that generates the corresponding SQL query.

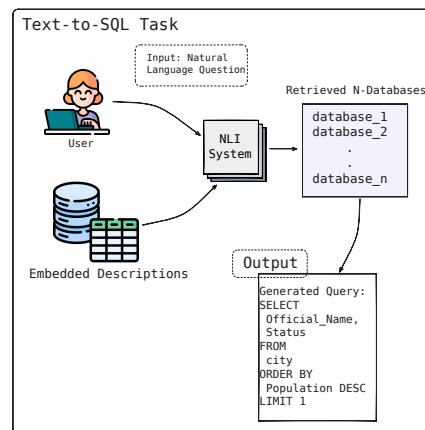


Figure 1: Typical workflow of a Text-to-SQL system.

Schema linking is essential for deploying Text-to-SQL solutions at scale across multiple databases (Wang et al., 2025b). Wang et al. (2025c) describe two key challenges associated with this task; the first, database selection, concerns finding the relevant database from a potentially large set of candidates. The second, schema item grounding, involves identifying the specific tables and columns relevant to the question within complex and often redundant schemas. This work focuses on the first challenge. Given a natural language question, we examine how to determine which database contains the information required to answer it and thereby support the generation of a correct SQL query.

The development of the Text-to-SQL task began with rule-based approaches designed to capture relationships between natural language queries

and database schema elements (Saha et al., 2016b). Performance improved with the introduction of neural architectures such as Recurrent Neural Networks (RNNs) and Transformers (Katsogiannis-Meimarakis and Koutrika, 2023). Large Language Models (LLMs) now represent the state of the art (Shi et al., 2025). Supported by task decomposition mechanisms and reasoning techniques such as Chain of Thought (CoT) and multi-agent debate, these models identify relevant schema components and generate correct SQL queries even in complex and redundant environments. Despite these advances, few studies address Brazilian Portuguese and even fewer examine Schema Linking (José and Cozman, 2021; Pedroso et al., 2025).

In this work, we investigate Schema Linking at the database level and formulate it as an Information Retrieval (IR) problem. As a first representation strategy, we use Data Definition Language (DDL) and Data Manipulation Language (DML) commands that define and populate tables. As a second strategy, we evaluate the feasibility of using LLMs to generate natural language descriptions of database schema elements (tables and columns). The following research questions guide our study:

Research Question 1 (RQ1): *To what extent can LLMs generate semantically informative natural language descriptions of databases, tables, and columns using only DDL and DML commands as input?*

Research Question 2 (RQ2): *Which representation strategy, DDL and DML or natural language descriptions, when embedded and indexed, yields higher accuracy in schema retrieval for questions written in Brazilian Portuguese?*

To address these research questions, we conducted experiments with several open-source and proprietary LLMs of different sizes and pre-training specializations. These models were evaluated for generating general database descriptions and individual descriptions of tables and columns. We also evaluated multiple embedding models for indexing both schema representations and natural language questions. The experiments were carried out using a translated version of the Spider dataset for Brazilian Portuguese, which includes more than 200 databases and 2,000 questions (José and Cozman, 2021; Pedroso et al., 2025). Evaluation was

based on the Hit@k metric, which measures the accuracy of retrieving the correct database for each question among the top-k returned results.

The main contributions of this work are summarized as follows:

- Investigate the feasibility of applying LLMs to generate database schema descriptions automatically.
- Compare schema representations derived from DDL and DML commands with those obtained from automatically generated natural language descriptions, and assess their feasibility for database schema retrieval.

All source code and full experimental results are available in a GitHub repository¹.

2 Related Work

Text-to-SQL and Schema Linking. The evolution of Text-to-SQL systems has been driven by advancements in Schema Linking, the process of identifying references to databases, tables, and columns in natural language questions. Early approaches such as Seq2SQL (Zhong et al., 2017) and SQL-Net (Xu et al., 2017) relied primarily on direct lexical matching between user input questions and database schemas. However, the introduction of relation-aware mechanisms, such as the one proposed in RAT-SQL (Wang et al., 2020), marked a paradigm shift by encoding the database schema as a graph to align question tokens with schema elements explicitly. Subsequent models, such as RESDSQL (Li et al., 2023) and S^2SQL (Hui et al., 2022), further refined this by decoupling schema linking, treating it as a distinct ranking or classification task. Despite these advances, schema linking struggles when there is a lexical gap between the vocabulary of the input question and the database nomenclature, a challenge that is increased in multilingual settings or when schemas use occult identifiers without metadata.

Metadata Augmentation and Retrieval. To bridge the semantic gap between database schemas and natural language, recent research emphasizes the role of external knowledge and schema documentation in natural language. Gao and Luo (2025) demonstrated that augmenting schemas with detailed descriptions improves the model’s ability

¹<https://github.com/daianekr/sql-desc-generation>

to bind ambiguous terms to the correct columns. However, real-world databases frequently suffer from the “cold-start” problem, lacking the manual documentation required for these methods. Simultaneously, industrial applications have shifted their focus to large-scale systems, where the correct database must first be identified from a repository. Systems like DBCopilot (Wang et al., 2025b) and CRUSH4SQL (Kothiyari et al., 2023) cast this as a retrieval problem. Our work sits at the intersection of these trends: we leverage LLMs to synthesize missing metadata (mitigating the “cold-start” problem) and use the generated descriptions to enhance dense retrieval in a cross-lingual (Brazilian Portuguese to SQL) environment.

Beyond individual-schema Text-to-SQL, several works rely on routing over collections of databases. Recent systems adopt large-scale schema selection and evaluate with ranking metrics: DBCopilot performs schema routing over massive databases and reports Recall@k and mAP (Wang et al., 2025b), CRUSH4SQL improves large-scale schema retrieval via hallucination-augmented candidates (Kothiyari et al., 2023), and RASL analyzes table retrieval with an emphasis on Recall@k in enterprise contexts (Eben et al., 2025). Along these lines, we frame database selection as an information retrieval task, aiming to retrieve the top- k most relevant database schemas for a given natural language question.

Motivated by these directions, our study explores automatic schema description generation using LLM and evaluates its effect on database retrieval, considering questions written in Brazilian Portuguese.

Schema Representation Strategies. The choice of schema representation is a critical factor in Text-to-SQL performance. MAC-SQL (Wang et al., 2025a) introduced a schema representation that combines table and column names with their natural language descriptions, along with high-frequency cell values, going beyond plain DDL to provide richer semantic context for LLMs. Building on this idea, Liu et al. (2026) proposed M-Schema, a semi-structured representation that presents hierarchical relationships between databases, tables, and columns, explicitly including data types, primary key markings, and example data values. These works share the motivation behind our strategy: enriching the raw schema with semantic information helps LLMs and re-

trieval models bridge the gap between natural language questions and database structures. While M-Schema and MAC-SQL Schema are designed primarily to feed schema context into SQL generators, our work focuses on the upstream database selection problem. We evaluated whether natural language descriptions generated by LLMs can provide more effective representations for identifying databases than a semi-structured approach based on DDL and DML commands, such as M-Schema.

3 Database Schema Retrieval Pipeline

Figure 2 outlines the workflow of the proposed method, which formulates the Schema Linking task as a retrieval-based pipeline leveraging natural language descriptions generated by LLMs. The problem addressed here can be formally defined as a database schema retrieval task based on semantic similarity. Let $\mathcal{S} = \{s_1, s_2, \dots, s_n\}$ be the set of database schemas whose textual representations have been previously indexed in a vector database, and let q be a question formulated in natural language. The goal is to identify a subset $\mathcal{S}_k(q) \subseteq \mathcal{S}$ containing the k database schemas most relevant to answering the question q . To do this, both the question and the textual representations associated with each schema s_i are projected onto a continuous vector space using an embedding model $f(\cdot)$. The retrieval step is then performed based on the cosine similarity function $\text{sim}(\cdot, \cdot)$, so that the schemas selected in $\mathcal{S}_k(q)$ correspond to those that maximize the similarity between $f(q)$ and $f(s_i)$.

The proposed pipeline consists of two main phases. In the first phase, natural language descriptions of database schema elements are generated from DDL and DML commands, then encoded using an embedding model and indexed in a vector database. In the second phase, given a natural language question, it is encoded using an embedding model and used to search the vector database, aiming to retrieve the top- k most likely database schemas to serve as input to a Text-to-SQL system that generates SQL to answer the input question.

Schema Description Generation. The first step of the proposed pipeline aims to generate natural language textual descriptions from an input database schema. The descriptions are generated at multiple levels of granularity, encompassing the database, its tables, and columns, following the prompt model defined in Table 1. This prompt template guides the format and content of the output,

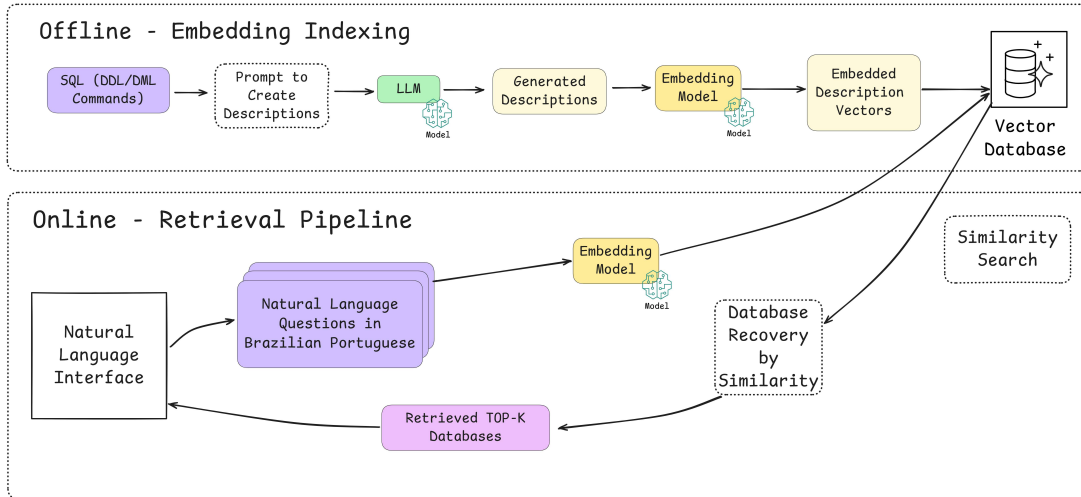


Figure 2: Overview of the workflow of the proposed database schema retrieval approach.

ensuring consistent descriptions of the database’s purpose, the tables’ functions, and the meaning of the columns.

The structural evidence used by the LLMs is derived exclusively from DDL commands, the CREATE TABLE instructions. Complementarily, when available, examples of DML commands, in our case, INSERT instructions, are used to provide the LLM with semantic clues, such as value formats of the columns, as illustrated in Table 2. This strategy aims to ensure that the generated descriptions remain grounded in the database schema while incorporating sufficient semantic information to support subsequent retrieval steps.

Table 3 presents an example of a Markdown-formatted text description generated in Brazilian Portuguese using the Qwen 2.5 Coder model for the database WineDB extracted from the Spider dataset and used as an example in Table 2. This example illustrates the result of the proposed multi-level description generation process, encompassing the database, its tables, and their respective columns. Initially, a general description is provided that summarizes the database’s purpose and the type of information it stores. Each table is accompanied by a concise explanation of its purpose, along with a list of columns, their data types, and their associated semantic meanings. This example demonstrates how the generated descriptions preserve the structure of the relational schema while translating the technical elements into understandable text, suitable for supporting semantic retrieval.

Database Schema Indexing and Retrieval. After generating natural language descriptions of the

database elements, the first step is to index the schema representation into a vector database. In this step, the textual representations generated in the previous step, which describe the database, its tables, and columns, are used as input for an embedding model. This model transforms textual descriptions into dense vector representations, while preserving their semantic content. The resulting vector representations are then stored in a vector database (in this work, ChromaDB² was used), associated with the identifiers of the respective schema and their elements, forming the indexing base used by the system.

Based on the index constructed from various textual descriptions of database schemas, it is possible to perform semantic searches. Thus, given a question written in natural language, the same embedding model is applied to generate its vector representation in the same semantic space as the previously indexed descriptions. This representation is used to query the vector database and retrieve the top- k database schemas that are most semantically similar to the input question. The idea is that the retrieved set can be used to define the scope for the subsequent steps in a Text-to-SQL application, thereby reducing the search space and improving alignment between the input question and the selected schemas.

4 Experimental Methodology

This section describes the experimental methodology adopted in this work. First, the Spider database is presented, highlighting its relevance to the task

²<https://www.trychroma.com/>

Table 1: Database description generation prompt template written in Brazilian Portuguese.

System Instructions

Você é um engenheiro de dados sênior especialista em modelagem relacional e documentação de sistemas legados.
Sua tarefa é gerar uma descrição objetiva e compreensível da estrutura de um banco de dados com base em comandos SQL de criação (“CREATE TABLE”) e, se fornecido, alguns exemplos de dados (“INSERT INTO”).

A descrição deve:

- Ser breve e clara.
- Apresentar o propósito geral do banco.
- Listar tabelas com seus nomes, finalidade e colunas.
- Explicar o que cada coluna representa, incluindo tipos e possíveis valores quando possível.

Formato da resposta:
Banco: <nome do banco>
Descrição geral: <...>
Tabela: <nome>
Descrição: <...>
Colunas:
 - “<coluna>” (“<tipo>”): <descrição>

ENTRADA:
 CREATE_SQL: {create_sql}
 INSERT_SQL: {insert_sql}

Table 2: Example of CREATE TABLE and INSERT commands used to replace create_sql and insert_sql in the prompt template.

CREATE TABLE

CREATE TABLE “grapes” (“ID” INTEGER PRIMARY KEY, “Grape” TEXT UNIQUE, “Color” TEXT);

CREATE TABLE “appellations” (“No” INTEGER PRIMARY KEY, “Appellation” TEXT UNIQUE, “County” TEXT, “State” TEXT, “Area” TEXT, “isAVA” TEXT);

CREATE TABLE “wine” (“No” INTEGER, “Grape” TEXT, “Winery” TEXT, “Appellation” TEXT, “State” TEXT, “Name” TEXT, “Year” INTEGER, “Price” INTEGER, “Score” INTEGER, “Cases” INTEGER, “Drink” TEXT, FOREIGN KEY (“Grape”) REFERENCES grapes(“Grape”), FOREIGN KEY (“Appellation”) REFERENCES appellations(“Appellation”));

INSERT INTO

INSERT INTO grapes (ID, Grape, Color) VALUES (1, ‘Barbera’, ‘Red’);

INSERT INTO appellations (No, Appellation, County, State, Area, isAVA) VALUES (1, ‘Alexander Valley’, ‘Sonoma’, ‘California’, ‘North Coast’, ‘Yes’);

INSERT INTO wine (No, Grape, Winery, Appellation, State, Name, Year, Price, Score, Cases, Drink) VALUES (1, ‘Zinfandel’, ‘Robert Biale’, ‘St. Helena’, ‘California’, ‘Old Kraft Vineyard’, 2008, 44, 93, 275, ‘now’);

under investigation. Next, we detail the embedding models and LLMs investigated in our experiments. Finally, the experimental setup and the evaluation metric employed are presented.

4.1 Spider Dataset

The Spider dataset is a widely used benchmark for Text-to-SQL applications (Yu et al., 2018) comprising 10,181 natural language questions and 5,693 SQL queries, distributed across 206 databases spanning 138 domains. Each entry includes several elements: a natural language question (originally in

Table 3: Example of description generated by the Qwen 2.5 Coder model for the database presented in Table 2.

```

### Banco: WineDB
Descrição geral: O banco de dados WineDB armazena informações sobre vinhos, suas uvas, regiões de produção e detalhes específicos sobre cada vinho, incluindo preço, classificação e recomendações de consumo.
#### Tabela: grapes
Descrição: Esta tabela contém informações sobre diferentes tipos de uvas utilizadas na produção de vinho, incluindo o nome da uva e sua cor.
Colunas:
- 'ID' ('INTEGER'): Identificador único da uva.
- 'Grape' ('TEXT'): Nome da uva.
- 'Color' ('TEXT'): Cor da uva (exemplos: 'Red', 'White').
#### Tabela: appellations
Descrição: Esta tabela fornece detalhes sobre as regiões de produção de vinho (appellations), incluindo informações geográficas e se a região é uma AVA (American Viticultural Area).
Colunas:
- 'No' ('INTEGER'): Identificador único da appellation.
- 'Appellation' ('TEXT'): Nome da appellation.
- 'County' ('TEXT'): Nome do condado onde a appellation está localizada.
- 'State' ('TEXT'): Nome do estado onde a appellation está localizada.
- 'Area' ('TEXT'): Área geográfica da appellation.
- 'isAVA' ('TEXT'): Indica se a appellation é uma AVA (American Viticultural Area) ('Yes' ou 'No').
#### Tabela: wine
Descrição: Esta tabela contém informações detalhadas sobre os vinhos, incluindo a uva utilizada, a vinícola, a região de produção, o ano de produção, o preço, a classificação e recomendações de consumo.
Colunas:
- 'No' ('INTEGER'): Identificador único do vinho.
- 'Grape' ('TEXT'): Nome da uva utilizada no vinho.
- 'Winery' ('TEXT'): Nome da vinícola que produziu o vinho.
- 'Appellation' ('TEXT'): Nome da appellation onde o vinho foi produzido.
- 'State' ('TEXT'): Nome do estado onde o vinho foi produzido.
- 'Name' ('TEXT'): Nome do vinho.
- 'Year' ('INTEGER'): Ano de produção do vinho.
- 'Price' ('INTEGER'): Preço do vinho.
- 'Score' ('INTEGER'): Classificação do vinho.
- 'Cases' ('INTEGER'): Número de garrafas produzidas.
- 'Drink' ('TEXT'): Recomendação de consumo do vinho (exemplos: 'now', '2012').

```

English), the corresponding SQL query, an indication of which database should be used to answer the question, and other relevant metadata. The dataset is publicly available and is organized into three subsets: train, validation, and test.

For the experiments reported in this work, we used a Brazilian Portuguese translation of the Spider database, which has been used in several other studies (José and Cozman, 2021; Pedroso et al., 2025; Carvalho et al., 2025). The development subset, translated by José and Cozman (2021), and the test subset, translated by Pedroso et al. (2025), were adopted. We used all 206 Spider databases, which have an average of 5.12 tables per set and a standard deviation of 3.69. The smallest database has 2 tables, while the largest has 26 tables. The development subset contains 166 databases and 1,034 questions, while the test subset contains 206 databases and 2,147 questions. This distribution

reflects the diversity of schemas and the volume of queries in Spider, characteristics relevant to analysis and evaluation in environments with multiple databases.

4.2 LLMs and Embedding Models

Different LLMs and embedding models were evaluated to compose the proposed workflow. Several LLMs were considered for producing textual descriptions of the database schemas, while different embedding models were examined for the indexing and searching stages in the vector database. Due to space constraints, we report only the models that achieved the best results in our experiments. The complete list of models analyzed, along with all experimental results, is available in the GitHub repository associated with the project.

LLMs. Both open-source and proprietary models, with distinct pre-training specializations, were considered for generating textual descriptions of the databases. The proprietary models used were OpenAI’s GPT-4o (OpenAI et al., 2024), and Sabiá-3 (Abonizio et al., 2025) developed by Maritaca.ai and specialized in Brazilian Portuguese. The proprietary models were used via the companies’ own Application Programming Interfaces (APIs). Among the open-source models, we employed Gemma 3 (Gemma Team et al., 2024) in its 12 billion parameter version, developed by Google, and Qwen 2.5 Coder (Qwen Team, 2024) with 32 billion parameters, developed by Alibaba, which is specialized in source code. These open-source models were executed with the support of the Transformers³ (Wolf et al., 2020) and the Unsloth⁴ (Han and Han, 2023) library, which provides quantization-optimized versions of the models in 4-bit, allowing for more efficient use of computational resources during inference. All LLMs were configured with a fixed temperature of 0.1, and each generation setup was executed once. Although this low temperature promotes greater determinism in the model outputs, it does not guarantee complete reproducibility, and the lack of repetitions represents a limitation of this study.

Embedding Models. The Qwen3 Emb 0.6B and BGE M3 embedding models were used to index the schema representations of the databases, including both DDL and DML information and the textual descriptions generated by the LLMs. The embedding models generate dense vector representations that allow the retrieval of the schemas considered relevant during the search process. The same architectures were applied to the encoding of natural language questions to semantically align their representations with the schema representation stored in the vector database.

4.3 Experimental Setup

To answer the research questions RQ1 and RQ2, experiments were conducted to compare two strategies for representing database schemas. The first representation uses only DDL (CREATE TABLE) and DML (INSERT command), as illustrated in Table 2, while the second uses natural language descriptions generated by LLMs, as presented in Table 3. Eight configurations were evaluated to

³<https://huggingface.co/>

⁴<https://unsloth.ai/>

compose the generation and retrieval steps of the proposed approach, considering the four LLMs and the two embedding models investigated.

We adopted the k -hit rate (Hit@ k) metric to evaluate the performance of the proposed retrieval approach, verifying whether the correct database associated with a natural language question is among the top- k ranked database schemas by the retrieval pipeline (Wu et al., 2024). This metric is particularly suitable for multi-database retrieval scenarios, where the main goal is to restrict the search space for subsequent steps, such as generating the SQL query, without necessarily requiring the correct database to rank first, but rather to rank in the top- k .

Formally, given a set of questions \mathcal{Q} from the Spider database, for each question $q \in \mathcal{Q}$, let D_q be the correct database, and $\mathcal{R}_k(q)$ be the set of top- k database schemas retrieved. The Hit@ k metric is defined as:

$$\text{Hit}@k = \frac{1}{|\mathcal{Q}|} \sum_{q \in \mathcal{Q}} \mathbb{I}(D_q \in \mathcal{R}_k(q)),$$

where $\mathbb{I}(\cdot)$ is an indicator function that takes the value 1 when the correct database is among the top- k retrieved and 0 otherwise. Thus, higher Hit@ k values indicate a greater ability of the method to retrieve, among the first k positions, the correct database schema relevant to answering the input question.

5 Results

Table 4 presents the results obtained for the different combinations of LLMs, embedding models, and representation strategies evaluated, showing how these factors influence schema retrieval from natural language questions. For simplicity, the representation using only DDL and DML commands is abbreviated DDL/DML, and the representation using natural language descriptions generated by the LLM is abbreviated as DESC. The values highlighted in bold indicate the best performance for each block of results, based on the Hit@1 (H@1), Hit@3 (H@3), and Hit@5 (H@5) metrics in the Spider development (Dev) and test sets. The analysis of these results allows us to answer research questions RQ1 and RQ2 and identify patterns in the behavior of the evaluated architectures.

Regarding RQ1, the results obtained from the natural language representations (DESC) indicate

Table 4: Experimental results based on Hit@1, Hit@3, and Hit@5 metrics in the Spider development (Dev) and test sets.

LLM	Embedding	Representation	Dev			Test		
			H@1	H@3	H@5	H@1	H@3	H@5
Gemma 3 12B	Qwen3 Emb. 0.6B	DDL / DML	0.3762	0.6083	0.6789	0.4122	0.5878	0.6619
		DESC	0.5716	0.7727	0.8327	0.4933	0.7336	0.8165
	BGE M3	DDL / DML	0.3878	0.5880	0.6528	0.4117	0.5580	0.6171
		DESC	0.5696	0.7631	0.8269	0.4867	0.6549	0.7424
Qwen 2.5 coder 32B	Qwen3 Emb. 0.6B	DDL / DML	0.3578	0.5948	0.6838	0.2361	0.3368	0.4769
		DESC	0.5677	0.7621	0.8327	0.5627	0.6474	0.7294
	BGE M3	DDL / DML	0.3772	0.5735	0.6364	0.3265	0.4360	0.5636
		DESC	0.6248	0.8017	0.8598	0.5487	0.6069	0.6935
GPT-4o	Qwen3 Emb. 0.6B	DDL / DML	0.3656	0.6093	0.6828	0.3787	0.5613	0.6446
		DESC	0.5532	0.7147	0.7979	0.4779	0.7052	0.7909
	BGE M3	DDL / DML	0.3830	0.5841	0.6518	0.4099	0.5515	0.6102
		DESC	0.5774	0.7631	0.8104	0.5179	0.7089	0.7955
Sabiá-3	Qwen3 Emb. 0.6B	DDL / DML	0.3898	0.5899	0.6741	0.4048	0.5790	0.6544
		DESC	0.5861	0.7940	0.8559	0.4891	0.7066	0.8053
	BGE M3	DDL / DML	0.3723	0.5696	0.6402	0.4066	0.5463	0.6111
		DESC	0.5812	0.7824	0.8462	0.5040	0.6623	0.7420

that the LLMs generated semantically informative descriptions of the elements of the database schemas. In all scenarios evaluated, the description-based representation consistently outperformed the direct use of structured DDL and DML commands. This result indicates that the LLMs generated database schema descriptions that were better aligned with natural language questions, thereby favoring schema retrieval.

When comparing the analyzed LLMs, Qwen 2.5 32B achieved the best results on the development set when paired with the BGE M3 embedding, especially with the DESC strategy, achieving Hit@1 of 0.6248, Hit@3 of 0.8017, and Hit@5 of 0.8598. In the test set, the highest Hit@1 value was obtained by the Qwen 2.5 32B combined with the Qwen3 Emb 0.6B embedding, though with smaller differences across the other metrics. The Gemma 3 12B model, jointly with the embedding representations generated by Qwen3 Emb 0.6B, achieved the best performance in Hit@3 (0.7336) and Hit@5 (0.8165). The GPT-4o and Sabiá-3 models also achieved high results with the description-based strategy, although lower than those achieved by Qwen 2.5 Coder 32B. This behavior suggests that specializing the model in the code domain led to more detailed schema descriptions, thereby improving retrieval performance. It is also noted that the results on the test set were slightly lower than those on the development set, possibly because the test

set contained more databases and questions, making the task more challenging.

Regarding the embedding models, the results show that BGE M3 presented the highest values in most scenarios involving the development set, especially when combined with Qwen 2.5 32B and the textual representation. In the test set, however, the Qwen3 Emb 0.6B embedding achieved competitive results and surpassed BGE M3 in some combinations. This variation indicates that the performance of the embedding models depends on the interaction between the type of representation and the LLM that generates the descriptions, rather than on the isolated capacity of each embedding.

Comparing the two representation strategies allows us to answer RQ2. The textual natural language representation consistently outperformed command-based DDL and DML representations across all models and evaluation scenarios, both in the development and test sets. This difference was especially pronounced in scenarios using the Qwen 2.5 32B and Sabiá-3 models, which showed substantial gains with the adoption of generated descriptions. These results suggest that the linguistic mediation introduced by LLMs better approximates the content of the questions and favors semantic alignment between them and the schemas stored in the vector database.

Finally, the results obtained under the Hit@5 scenario reinforce the effectiveness of several eval-

uated configurations, especially the combination of Qwen 2.5 32B with BGE M3 in the development set and the combination of Gemma 3 12B with Qwen3 Emb 0.6B in the test set. The best configurations reached accuracy rates exceeding 80.00% in both the development and test sets. These results indicate that, across the 166 databases in the development set and the 206 in the test set, it was possible to correctly retrieve, within the top five positions returned by the vector database, the schema necessary to answer the input question. These results show that, even in scenarios with a large number of possible databases, automatically generating textual descriptions using LLMs for the Schema Linking task, combined with information retrieval-based modeling, constitutes a viable approach.

6 Conclusions

This work investigated the use of textual descriptions generated by large language models as a schema representation strategy for the Schema Linking task in the context of Text-to-SQL. Experimental results obtained using a Brazilian Portuguese translation of the Spider benchmark consistently demonstrate that natural language representations outperform the direct use of DDL and DML commands across all evaluated settings. These findings indicate that LLM-generated descriptions provide improved semantic alignment between natural language questions and schema elements, resulting in higher Hit@k performance. A comparative analysis of the models shows that architectures specialized in the code domain, such as the open-source models Qwen 2.5 Coder 32B, generate more informative descriptions and achieve superior results across multiple scenarios, even compared to the proprietary models GPT-4o and Sabiá-3.

Despite the encouraging results, this study has several limitations. First, all LLMs were executed with a fixed temperature of 0.1, and each configuration was run only once. Although this low temperature setting encourages more deterministic outputs, it does not ensure full reproducibility, and the lack of repeated runs prevents a formal assessment of variability. Second, the evaluation is limited to retrieval-level metrics (Hit@k) and does not assess the downstream impact of schema linking on SQL generation accuracy. Third, the study does not include a human qualitative evaluation of the LLM-generated descriptions.

As future work, we plan to incorporate a hu-

man evaluation of the generated descriptions to assess their faithfulness and informativeness. We also intend to extend our comparison to alternative schema representation strategies, including semi-structured formats that combine hierarchical schema organization with data types, primary key annotations, and example data values, as explored in recent Text-to-SQL systems. Additionally, we aim to introduce a retrieval stage that selects only the tables most likely needed to generate the target SQL query, further reducing the search space. Another key direction is to integrate the proposed retrieval pipeline into a complete Text-to-SQL system, enabling a direct assessment of how the quality of schema linking affects SQL generation accuracy.

References

- Hugo Abonizio, Thales Sales Almeida, Thiago Laitz, Roseval Malaquias Junior, Giovana Kerche Bonás, Rodrigo Nogueira, and Ramon Pires. 2025. *Sabiá-3 technical report*. Preprint, arXiv:2410.12049.
- Luís Carvalho, Paulo S. Júnior, and Hilário Oliveira. 2025. *Benchmarking large language models for text-to-sql in brazilian portuguese and english*. In *Anais do XVI Simpósio Brasileiro de Tecnologia da Informação e da Linguagem Humana*, pages 101–112, Porto Alegre, RS, Brasil. SBC.
- Jeffrey Eben, Aitzaz Ahmad, and Stephen Lau. 2025. *Rasl: Retrieval augmented schema linking for massive database text-to-sql*. In *Proceedings of the KDD Workshop on Structured Knowledge for Large Language Models (SK-LLM)*. See also arXiv:2507.23104.
- Yingqi Gao and Zhiling Luo. 2025. *Automatic database description generation for text-to-sql*. arXiv preprint arXiv:2502.20657. Also cross-listed in cs.CL and cs.DB.
- Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, and 1 others. 2024. *Gemma: Open models based on gemini research and technology*. arXiv preprint arXiv:2403.08295.
- Daniel Han and Michael Han. 2023. *Unsloth: Faster and memory efficient llm fine-tuning*.
- Binyuan Hui, Ruiying Geng, Qiyu Ren, Binhua Li, Yongbin Li, Jian Sun, Luo Si, and Pengfei Zhu. 2022. *s²sql: Injecting syntax to question-schema interaction graph for text-to-sql*. In *ACL*.
- Marcelo Archanjo José and Fabio Gagliardi Cozman. 2021. *mRAT-SQL+GAP: A Portuguese Text-to-SQL Transformer*, page 511–525. Springer International Publishing.

- George Katsogiannis-Meimarakis and Georgia Koutrika. 2023. [A survey on deep learning approaches for text-to-SQL](#). *The VLDB Journal*, 32(4):905–936.
- Mayank Kothiyari, Dhruva Dhingra, Sunita Sarawagi, and Soumen Chakrabarti. 2023. [Crush4sql: Collective retrieval using schema hallucination for text2sql](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- Haoyang Li, Jing Zhang, Cuiping Li, and Hong Chen. 2023. [Resdsq: Decoupling schema linking and skeleton parsing for text-to-sql](#). In *AAAI*.
- Yifu Liu, Yin Zhu, Yingqi Gao, Zhiling Luo, Xiaoxia Li, Xiaorong Shi, Yuntao Hong, Jinyang Gao, Yu Li, Bolin Ding, and Jingren Zhou. 2026. [Xiyansql: A novel multi-generator framework for text-to-sql](#). *IEEE Transactions on Knowledge and Data Engineering*, 38(4):2474–2487.
- OpenAI, Aaron Hurst, and 1 others. 2024. [Gpt-4o system card](#). *arXiv preprint arXiv:2410.21276*.
- Breno Carvalho Pedrosa, Marluce Rodrigues Pereira, and Denilson Alves Pereira. 2025. [Performance evaluation of LLMs in the text-to-SQL task in portuguese](#). In *Anais do Simpósio Brasileiro de Sistemas de Informação (SBSI 2025)*, pages 260–269. Sociedade Brasileira de Computação (SBC).
- Qwen Team. 2024. [Qwen2.5 technical report](#). *arXiv preprint arXiv:2412.15115*.
- Diptikalyan Saha, Avriella Floratou, Karthik Sankaranarayanan, Umar Farooq Minhas, Ashish R. Mittal, and Fatma Özcan. 2016a. [Athena: an ontology-driven system for natural language querying over relational data stores](#). *Proc. VLDB Endow.*, 9(12):1209–1220.
- Diptikalyan Saha, Avriella Floratou, Karthik Sankaranarayanan, Umar Farooq Minhas, Ashish R. Mittal, and Fatma Özcan. 2016b. [Athena: an ontology-driven system for natural language querying over relational data stores](#). 9(12):1209–1220.
- Liang Shi, Zhengju Tang, Nan Zhang, Xiaotong Zhang, and Zhi Yang. 2025. [A survey on employing large language models for text-to-sql tasks](#). *ACM Comput. Surv.*, 58(2).
- Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. 2020. [RAT-SQL: Relation-aware schema encoding and linking for text-to-sql parsers](#). In *Proceedings of ACL*.
- Bing Wang, Changyu Ren, Jian Yang, Xinnian Liang, Jiaqi Bai, Linzheng Chai, Zhao Yan, Qian-Wen Zhang, Di Yin, Xing Sun, and 1 others. 2025a. [Mac-sql: A multi-agent collaborative framework for text-to-sql](#). In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 540–557.
- Tianshu Wang, Hongyu Lin, Xianpei Han, Le Sun, Xiaoyang Chen, Hao Wang, and Zhenyu Zeng. 2025b. [Dbcopilot: Natural language querying over massive databases via schema routing](#). In *Proceedings of the 28th International Conference on Extending Database Technology (EDBT)*.
- Yihan Wang, Peiyu Liu, and Xin Yang. 2025c. [LinkAlign: Scalable schema linking for real-world large-scale multi-database text-to-SQL](#). In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 977–991, Suzhou, China. Association for Computational Linguistics.
- Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, and 1 others. 2020. [Transformers: State-of-the-art natural language processing](#). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45.
- Shirley Wu, Shiyu Zhao, Michihiro Yasunaga, Kexin Huang, Kaidi Cao, Qian Huang, Vassilis N Ioannidis, Karthik Subbian, James Zou, and Jure Leskovec. 2024. [Stark: Benchmarking llm retrieval on textual and relational knowledge bases](#). *Advances in Neural Information Processing Systems*, 37:127129–127153.
- Xiaojun Xu, Chang Liu, and Dawn Song. 2017. [Sqlnet: Generating structured queries from natural language without reinforcement learning](#). *arXiv preprint arXiv:1711.04436*.
- Tao Yu, Chien-Sheng Wu, Xi Victoria Lin, Bailin Wang, Yi Chern Tan, Xinyi Yang, Dragomir Radev, Richard Socher, and Caiming Xiong. 2021. [Grappa: Grammar-augmented pre-training for table semantic parsing](#). *Preprint*, arXiv:2009.13845.
- Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. 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.
- Victor Zhong, Caiming Xiong, and Richard Socher. 2017. [Seq2sql: Generating structured queries from natural language using reinforcement learning](#). *arXiv preprint arXiv:1709.00103*.