Text-to-SQL Beyond Benchmarks: A Case Study on Large-Scale Deployment

Lucas Nobre[1] **, Vinicius Aguiar**[1] **, José Oliveira**[1] **,**
Lucas Alves[1] **, Anderson Soares**[1] **, Sávio Teles**[1]

[1] Instituto de Informática – Universidade Federal de Goiás (UFG)

lucasnobre212@discente.ufg.br
vinicius_aguiar2@discente.ufg.br
josefleury@discente.ufg.br
lucas.alves2@discente.ufg.br
anderson@inf.ufg.br
savioteles@ufg.br

Abstract. *Most Text-to-SQL evaluation evidence comes from academic benchmarks with small schemas and curated metadata, which differ from the databases these systems face in deployment. This study evaluates an agentic Text-to-SQL system deployed over open data from ANEEL, the Brazilian electricity regulator, a 140-table Brazilian-Portuguese database whose schema metadata is extracted automatically from dataset documentation. The agent is compared against a non-agentic one-shot baseline, with and without exploration tools, across schema scopes of 10, 50, and 140 tables, with BIRD-PT as an academic reference point. Each answer is scored under strict execution accuracy, relaxed execution accuracy, and a semantic-match judge. The agent reaches 63.56% strict execution accuracy on BIRD-PT and 15.01% on ANEEL under the same model, a gap of at least 36 percentage points on every metric. The correct table is retrieved for 60% of ANEEL questions against nearly 100% BIRD questions, and accuracy falls further during query construction. Across all schema scopes, strict execution accuracy ranks the baseline above the agent while relaxed execution accuracy and semantic match rank the agent above it, so the metric chosen decides which system a practitioner would select.*

1. Introduction

Large regulatory and operational databases contain high-value information for strategic and analytical decision making. However, accessing these datasets requires knowledge of SQL, familiarity with schema structures, and understanding of domain-specific terminology. In practice, many domain specialists need to query such systems but lack the expertise required to construct executable queries over complex schemas.

Recent work in Text-to-SQL has improved semantic parsing through relation aware schema encoding [Wang et al. 2020], modular decomposition [Li et al. 2023], in-context learning with self-correction [Pourreza and Rafiei 2023], and multi-path reasoning strategies [Pourreza et al. 2024]. Multi-agent frameworks extend these approaches with structured refinement and iterative correction [Wang et al. 2025,Cen et al. 2025]. Benchmarks such as BIRD [Li et al. 2024] evaluate larger databases, while multilingual studies show that performance varies across languages [de Carvalho et al. 2025,Nobre et al. 2026].

Most benchmarks assume limited schema size and full schema serialization into the model context. In operational settings, schemas often contain hundreds of tables with

heterogeneous metadata, and serializing all of them into the prompt increases noise and token usage. Execution accuracy, the metric adopted since Spider [Yu et al. 2018], measures correctness from query results but can penalize semantically valid queries that differ in structure. Under these conditions, a single execution-accuracy score can rank systems in the wrong order.

In this work, we present a deployment case study of an agentic Text-to-SQL system applied to open data from the Brazilian electricity sector. The target database holds 140 tables in Brazilian Portuguese, organized into 13 thematic groups, with schema metadata extracted automatically from dataset documentation. We compare the agent against a non-agentic one-shot baseline, with and without the agent's exploration tools across schema scopes of 10, 50, and 140 tables. We use BIRD-PT as a benchmark comparison point, and score each answer with strict execution accuracy, relaxed execution accuracy, and semantic match over the natural-language answer.

At the full 140-table scope the ranking depends on the metric, with the baseline leading on strict execution accuracy (19.99 versus 15.01) while the agent with exploration leads on relaxed execution accuracy (27.77 versus 22.68) and semantic match (33.26 versus 25.52). The same disagreement consistently appears across the smaller scopes. The truth table appears in the top five retrieved tables for 60% of questions, yet strict accuracy stays below 20 percent, which shows that surfacing the right table does not yield a correct query. Exploration raises relaxed accuracy and semantic match, and larger schema scope lowers both retrieval and generation accuracy.

This study contributes a deployment-grounded evaluation of an agentic Text-to-SQL system over ANEEL open data, with a benchmark of 2,571 Portuguese questions over a 140-table regulatory schema, each answer scored by strict execution accuracy, relaxed execution accuracy, and a semantic-match judge. The same agent reaches 63.56% strict execution accuracy on BIRD-PT but 15.01% on the ANEEL schema, so a translated academic benchmark does not reflect the difficulty of this deployment domain. At the 140-table scope, execution accuracy ranks the agent and the one-shot baseline in the opposite order from the other two metrics, showing that using only execution accuracy can change the optimal choice for a deployed system.

2. Related work

Text-to-SQL has been studied as a semantic parsing task that maps natural language questions to executable SQL queries. Benchmarks such as Spider established a standard setting for cross-domain evaluation over unseen databases [Yu et al. 2018]. Subsequent models improved schema grounding by encoding relations between tables, columns, and question tokens. RAT-SQL introduced relation-aware schema encoding [Wang et al. 2020], while ResdSQL decoupled schema linking from skeleton parsing [Li et al. 2023], establishing schema linking as a core component.

The emergence of large language models shifted research toward prompting, in-context learning, and self-correction. DIN-SQL decomposes the task into intermediate reasoning steps with self-correction [Pourreza and Rafiei 2023], while CHASE-SQL applies multi-path reasoning and candidate selection [Pourreza et al. 2024].

Agentic and collaborative approaches extend this line of work. MAC-SQL introduces multi-agent collaboration across task stages [Wang et al. 2025], and SQLFixAgent

applies consistency-based refinement for query repair [Cen et al. 2025]. These systems show that tool use and iterative refinement improve robustness. In contrast, our work examines agentic Text-to-SQL under deployment constraints, where schema retrieval, context size, and execution cost become central.

Large-scale benchmarks such as BIRD expand evaluation to larger databases and more complex queries [Li et al. 2024]. However, benchmark settings differ from operational environments, where schemas may contain hundreds of tables, metadata quality varies, and latency constraints apply. Prior work also shows that NL2SQL remains unresolved in practical settings [Floratou et al. 2024].

Evaluation remains an open issue. Execution accuracy measures correctness based on query results but may penalize semantically equivalent outputs. Test suite accuracy [Zhong et al. 2020] and structural similarity metrics [Song et al. 2024] address these limitations. These considerations motivate the use of relaxed execution accuracy and semantic matching alongside strict execution accuracy.

In Text-to-SQL, methods such as CHESS emphasize schema selection and context construction [Talaei et al. 2024], while metadata extraction improves schema representations [Shkapenyuk et al. 2025].

Multilingual Text-to-SQL introduces additional challenges. Prior work evaluates performance across Brazilian Portuguese and English [Nobre et al. 2026] and explores multilingual settings with collaborative agents [Pham et al. 2025].

3. Industrial Setting and Design Constraints

We ground this study in a production deployment at a major Brazilian electric utility, where analysts and regulatory compliance teams routinely query public datasets released by ANEEL. These users have domain expertise but limited proficiency in SQL, which creates a gap between the questions they need to answer and the data required to answer them.

The underlying database contains 140 tables organized into 13 thematic groups, including transmission infrastructure, distributed generation, tariff components, power interruption records, and regulatory oversight data. The tables vary in structure, vocabulary, granularity, and temporal coverage. Some contain fewer than 100 rows of static reference data, while others aggregate millions of records collected over multiple years. This heterogeneity introduces ambiguity during schema linking, as different tables may use overlapping terminology while encoding distinct concepts. For example, multiple tables refer to generation capacity or tariff values, but differ in aggregation level, temporal resolution, and column semantics.

We use this setting to construct a benchmark for evaluating Text-to-SQL systems under realistic conditions. We curate a dataset of 2,571 natural language questions paired with SQL queries, all grounded in the ANEEL domain. The questions are expressed in Brazilian Portuguese and were generated from dataset descriptions and data dictionaries using Gemini-3-Flash, followed by manual review to ensure correctness and alignment with the underlying data. The benchmark covers all tables and thematic groups, with a distribution that reflects differences in domain size and complexity.

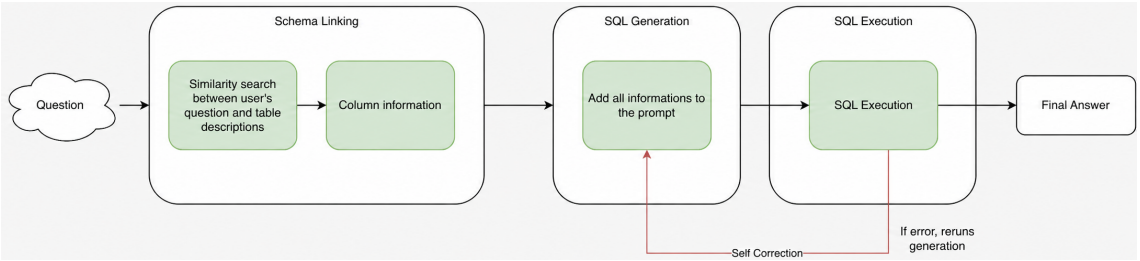


Fig. 1. High-level pipeline for the agentic Text-to-SQL system.

The queries include common SQL operations such as `WHERE` filters, `GROUP BY`, `COUNT`, `SUM`, and `AVG`. All queries operate over single tables, reflecting the denormalized structure of the ANEEL datasets. This design isolates schema selection and column grounding as the primary challenges, without introducing join reasoning. The system operates entirely in Brazilian Portuguese, including user questions, schema descriptions, and final responses.

This setting allows us to study Text-to-SQL under conditions where schema scale, metadata heterogeneity, and context constraints affect both retrieval and query generation.

4. Agentic Pipeline Architecture

We implement the Text-to-SQL system as a tool-augmented agent for large relational schemas under deployment constraints. Instead of serializing the full schema into one prompt, the agent invokes tools for schema retrieval, metadata enrichment, SQL generation, execution, validation, and controlled exploration. This design bounds context size and database access while enabling execution-guided refinement.

Figure 1 summarizes the pipeline. The process has four stages: schema linking, SQL generation, SQL execution, and refinement. Each stage is exposed as a callable tool, so the agent can use intermediate outputs to decide whether to generate, execute, refine, or explore.

Schema linking identifies the tables and columns relevant to the user question. We retrieve candidate tables through embedding-based similarity search between the question and table descriptions. The selected tables are enriched with column names and descriptions extracted from data dictionaries, which provide grounding signals for distinguishing similar tables and columns.

Given this schema context, the agent generates and executes a candidate SQL query. If execution fails or returns inconsistent results, the system allows a bounded number of refinement iterations. Execution feedback is used to correct syntax errors, adjust query structure, and check whether the result aligns with the user question.

The agent can also inspect data before producing the final query. The explore_tables tool runs constrained SQL queries with an explicit `LIMIT` clause and returns the execution status, executed query, sampled rows, and number of returned records. This mechanism supports disambiguation when schema descriptions are insufficient.

Figure 2 shows the implementation over a PostgreSQL database with vector embeddings. Schema-linking tools retrieve and enrich metadata, while execution tools support exploration and final SQL execution.

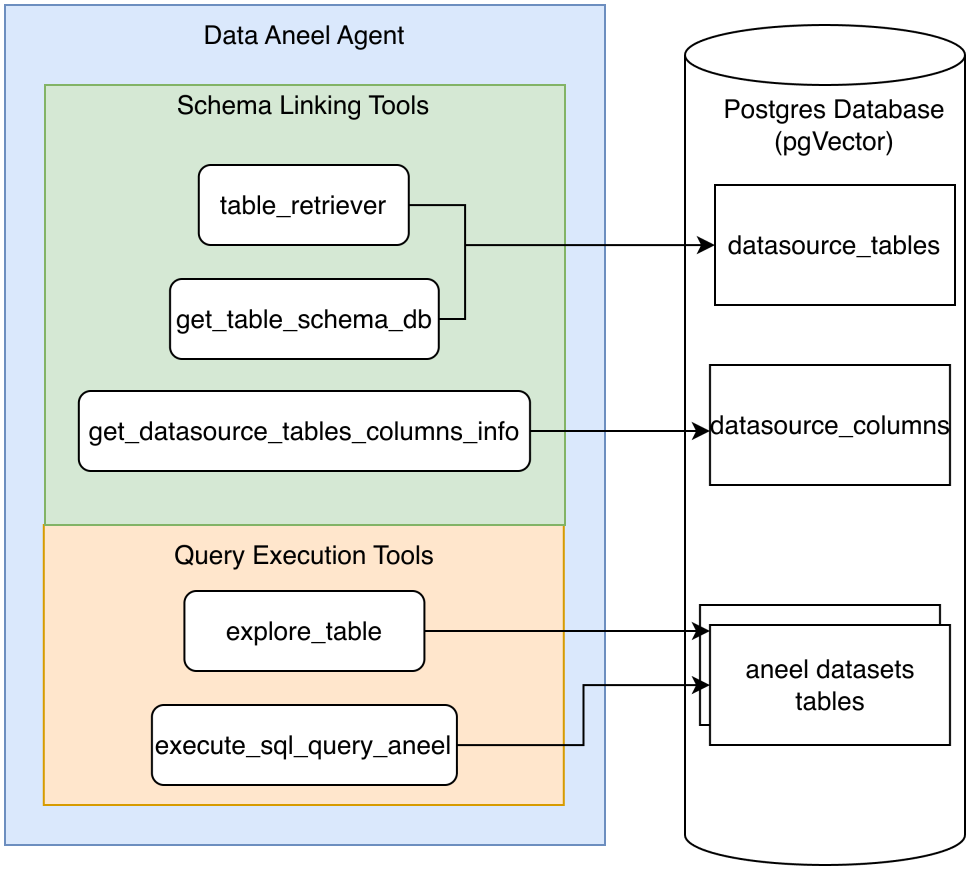


Fig. 2. Tool-based implementation of the agent.

The tools are implemented at a coarse-grained level to keep orchestration predictable. This pipeline allows us to study how schema grounding, controlled exploration, and execution feedback interact with retrieval and schema scale in deployment.

5. Methodology

We evaluate a deployed agentic Text-to-SQL system over a Brazilian regulatory database. We run the system on the ANEEL benchmark to measure its behavior on the deployment domain, and on BIRD-PT to compare that behavior against an academic benchmark in the same language. We score every answer under three metrics, because execution accuracy alone can penalize a correct answer shaped differently from the gold query. We vary schema scale, retrieval, and the exploration interface to locate where errors arise.

5.1. Benchmarks

The first benchmark is the ANEEL set of 2,571 Portuguese question and query pairs over the 140-table regulatory schema. A subset of 704 questions received independent human review, of which 661 (94%) were approved. The 592 approved questions that carry run results form a validated subset spanning 138 tables, nearly the entire schema, which we use for the qualitative analysis. The second is BIRD-PT, a Portuguese translation of the BIRD-SQL development set, included as an academic-benchmark reference point against which to read the ANEEL results. It contains 1,534 questions over 11 SQLite databases with 75 tables. The translation covers the questions and the evidence hints, while the

schema and stored values remain in English. We use the provided database identifiers as retriever filters and optionally include the evidence hints in the input.

5.2. Metrics

Execution Accuracy compares result sets, but a query can fail it while still returning the user the right answer, for example when the agent projects extra columns alongside the requested value. To separate row-level exactness from answer correctness, we score each output under three metrics. Execution Accuracy requires an exact match of result sets after normalization. Relaxed Execution Accuracy allows additional columns and checks whether the ground truth rows are contained in the predicted result. Semantic Match evaluates whether the returned answer resolves the user query.

Semantic Match is computed by an LLM-as-a-judge, evaluated with two models, `gemini-2.5-flash` at temperature 0 and `claude-opus-4-7`. Each judge receives the question, the ground truth rows, and the agent's final answer, and returns a binary verdict on whether the prose answer is consistent with the ground truth. The judge tolerates aliasing, formatting, and minor numerical rounding, but flags wrong values, missing rows, or incorrect ordering. Because Gemini models are used to construct the ANEEL benchmark and to produce the system outputs, the Gemini judge shares model-family biases with the evaluated system. The second judge, from a different model family, tests sensitivity to that overlap.

5.3. Experimental Setup

All experiments use Gemini-3-Flash (Vertex AI) with temperature 0.2. The agent operates in Brazilian Portuguese and interacts with the database through tool calls until it produces a final SQL query. Retrieval embeds the question with text-multilingual-embedding-002 (768 dimensions) and ranks tables by cosine similarity over precomputed description embeddings. We fix the retriever to its top five tables in all configurations, which bounds the context and separates retrieval failures from generation errors.

We vary four experimental factors: benchmark (ANEEL or BIRD-PT), system configuration (agent or non-agentic baseline), availability of the exploration tool, and schema scope. For ANEEL, we evaluate three schema sizes: 10, 50, and 140 tables. Subsets of 10 and 50 tables are sampled deterministically using three independent seeds. Each agent run is paired with a baseline run over the same questions and schema subset. For BIRD-PT, we additionally vary whether the retriever operates over the target database or the full set of tables, and whether evidence hints are included in the input.

The non-agentic baseline shares the agent's model, temperature, retriever, top-k, and schema metadata. It issues one retrieval call, one SQL-generation call, and one execution, with no exploration, no feedback loop, and no retry. The two systems therefore differ only in whether they can act on intermediate results, which isolates the effect of agentic orchestration.

6. Results

We analyze system behavior across benchmarks, schema scales, and configurations to understand how Text-to-SQL systems operate under deployment conditions. We focus on how retrieval, generation, and evaluation interact.

Table 1. Cross-benchmark performance. BIRD-PT varies `evidence` and `explore`, ANEEL varies `explore`. Bold marks the best score per benchmark.

Benchmark	System	Strict EA	Relaxed EA	Sem. Match
BIRD-pt (1,534 q)	Baseline (one-shot LLM)	50.39	55.67	51.83
	Agent (no ev., no ex.)	51.96	63.04	60.76
	Agent (no ev., +ex.)	52.22	63.30	59.91
	Agent (+ev., no ex.)	63.23	**70.86**[*]	69.04
	Agent (+ev., +ex.)	**63.56**	**70.73**[*]	**69.69**
ANEEL (2,571 q, 140 t)	Baseline (one-shot LLM)	**19.99**	22.68	25.52
	Agent (no ex.)	12.76	24.04	26.99
	Agent (+ex.)	15.01	**27.77**	**33.26**

[*]On BIRD-pt the no-explore variant (70.86) edges the +explore variant (70.73) on relaxed EA by 0.13 pp; both are bolded as a tie. All values are percentages.

6.1. Cross-benchmark behavior

Table 1 reports performance on BIRD-PT and ANEEL. On BIRD-PT, the agent reaches 63.56% strict Execution Accuracy, while on ANEEL the same configuration reaches 15.01%. This gap exceeds 36 percentage points across all metrics, despite using the same model and language.

This result indicates that the same model behaves differently depending on the database setting. Compared to BIRD-PT, the ANEEL setting introduces larger schema scale, heterogeneous metadata, and increased ambiguity during schema selection.

On BIRD-PT, the agent outperforms the baseline across all metrics. On ANEEL, the ranking depends on the metric. Under strict Execution Accuracy, the baseline is higher, while under relaxed Execution Accuracy and semantic match the agent is higher. This difference reflects how each system structures its outputs rather than differences in row-level correctness.

6.2. Retrieval and downstream errors

Table 2 compares retrieval recall@5 and strict Execution Accuracy. Recall exceeds 55% on ANEEL and approaches 100% on BIRD-PT for both systems. The agent consistently retrieves the correct table more often than the baseline.

Despite this, strict Execution Accuracy remains lower than recall@5 on ANEEL. This indicates that errors arise after retrieval, including incorrect column selection, filter conditions, and aggregation choices. Retrieval provides access to the correct table, but does not ensure correct query construction.

6.3. Effect of schema scale

Table 3 reports performance across schema scopes. For the baseline, strict Execution Accuracy remains stable across 10, 50, and 140 tables. In contrast, the agent varies across scopes, particularly without exploration.

This pattern suggests that the baseline does not fully use schema context, while the agent is more sensitive to the quality of retrieved context. Reducing schema size improves retrieval conditions, but does not eliminate downstream errors.

Table 2. Retrieval recall@K (any-call) vs. strict execution accuracy. ANEEL multi-seed rows average recall over seeds A, B, C; strict EA is from Table 3.

Scope	System	R@1	R@3	R@5	Strict EA
ANEEL 140 t	Baseline	29.72	46.95	55.46	19.99
	Agent	32.67	51.23	**60.02**	15.01
ANEEL 50 t (A/B/C)	Baseline	31.68	48.37	57.19	20.01
	Agent	33.40	50.90	59.41	14.70
ANEEL 10 t (A/B/C)	Baseline	30.37	50.79	63.63	20.22
	Agent	34.95	57.31	**68.30**	16.41
BIRD-pt	Baseline	80.70	97.78	99.48	50.39
	Agent	82.59	98.57	**100.00**	52.22

Values in percent. ANEEL agent rows use `+explore`. BIRD agent row uses no-evidence, no-explore.

6.4. Structured exploration

The effect of exploration is most clearly observed at the 140-table scope. Enabling explore_tables increases strict Execution Accuracy by 2.25 percentage points and semantic match by 6.27 points.

The larger effect on semantic match indicates that exploration improves alignment between generated queries and observed data values. The agent uses exploration to verify assumptions about column contents, which affects how results are constructed.

6.5. Metric disagreement

The largest difference between the agent and the baseline appears when comparing strict and relaxed metrics. At 140 tables, the increase from strict to relaxed Execution Accuracy is 2.69 points for the baseline and 12.76 points for the agent.

This pattern reflects differences in output structure. The baseline produces queries that closely match the gold projection, while the agent often includes additional columns alongside the target values. Strict Execution Accuracy penalizes these outputs, while relaxed Execution Accuracy and semantic match do not.

As a result, strict Execution Accuracy alone reverses the ranking between the agent and the baseline observed under other metrics. This behavior is consistent with prior observations that strict Execution Accuracy may not capture semantic equivalence for systems that produce richer outputs [Floratou et al. 2024,Song et al. 2024].

6.6. Error analysis

We classify the failures of the Agent Base configuration on the 592-question validated subset. An LLM labels each failing case against a fixed taxonomy, re-executing the gold and generated SQL to confirm the root cause, with flagged labels re-checked. After removing 125 cases lost to a SQL-capture artifact in this run, 377 real failures remain (Table 4).

Schema linking is the largest failure mode, with wrong-table and wrong-column errors accounting for 151 of the 377 failures (40%), consistent with the retrieval gap on the large ANEEL schema. A further 66 cases (17.5%) are not agent errors. Of these, 56 are answers that strict Execution Accuracy scores zero on column or formatting differences but that relaxed and semantic metrics accept, and 10 are questions whose gold query is

Table 3. ANEEL means by scope, system, and metric. Multi-seed rows (50 t, 10 t) average over seeds A, B, C.

Scope	System	Strict EA	Relaxed EA	SM (Gem.)	SM (Cla.)
140 t	Baseline	**19.99**	22.68	25.52	29.65
	Agent	12.76	24.04	26.99	31.58
	Agent +ex.	15.01	**27.77**	**33.26**	**39.86**
50 t (A/B/C)	Baseline	**20.01**	22.39	25.76	29.46
	Agent	15.05	**28.74**	**31.36**	37.73
	Agent +ex.	14.70	26.32	30.41	**38.47**
10 t (A/B/C)	Baseline	**20.22**	22.65	27.04	31.45
	Agent	19.01	**30.09**	**32.05**	37.86
	Agent +ex.	16.41	27.23	27.34	**38.95**

Values in percent. Bold marks the best score in each column within a scope. Multi-seed scopes pool seeds A, B, C (50 t: 913/900/938 questions, 10 t: 164/179/179); 140 t is the full 2,571-question corpus. SM (Gem.) uses `gemini-2.5-flash` and SM (Cla.) uses `claude-opus-4-7`; errored items count as 0.

Table 4. Error taxonomy for the Agent Base configuration on the validated subset (377 real failures).

Category	Count	Share
wrong-table	140	37.1%
correct-but-format-differs	56	14.9%
filter-value	55	14.6%
abstention (genuine)	40	10.6%
aggregation	23	6.1%
projection-mismatch	18	4.8%
wrong-column	11	2.9%
join-error	10	2.7%
gold-query-wrong	10	2.7%
ordering-limit	8	2.1%
wrong-metric	4	1.1%
hallucination	2	0.5%
Total	**377**	**100%**

Failures of the Agent Base run (exploration off) on the 592 validated questions, classified by an LLM with re-execution of the gold and generated SQL and re-checking of flagged labels. 125 cases lost to a SQL-capture artifact in this run are excluded. `correct-but-format-differs` and `gold-query-wrong` are not agent errors.

itself wrong, which quantifies the metric limitation behind the strict-versus-relaxed reversal. The remaining failures are filter-value errors (55), genuine abstentions (40, some over-cautious refusals), and a long tail of aggregation, projection, join, ordering, and metric errors.

Table 5 illustrates these categories on cases drawn from the validated subset, with the correct table retrieved in each. Cases 1 to 4 are downstream errors that all metrics penalize, while Case 5 is a correct answer that strict Execution Accuracy scores zero on a projection difference but relaxed and semantic metrics accept.

7. Discussion

Strict Execution Accuracy places the baseline above the agent at every scope, while relaxed Execution Accuracy and both semantic-match judges place the agent above it, so which system is selected depends on which metric is reported. The qualitative cases explain the inversion, since the agent often returns the requested value alongside extra columns, which exact matching scores as wrong while the relaxed and semantic metrics

Table 5. Five illustrative ANEEL cases (truth table retrieved). Cases 1–4 are downstream errors; Case 5 shows the strict vs. relaxed difference.

Q#	Truth element	Agent's choice	Error class
1	Table `projetos_eficiencia_energetica_equipamento`	Table `projetos_eficiencia_energetica_empresa`	Wrong table among similarly-named candidates
2	Filter `dsc_unidade = 'R$/MWh'`	Filter `dsc_unidade_terciaria = 'MWh'`	Wrong column for filter
3	`SUM(COALESCE(vlr_multa_apos_diretoria, vlr_multa_apos_juizo, vlr_penalidade))`	`SUM(vlr_penalidade)`	Wrong aggregation column
4	Filter year IN (1996, 1997, 1998)	Filter `ano = 2024`	Wrong filter value
5	Truth row: `{percentual: 55.39}`	Agent row: `{pot_2010: 112399, pot_2020: 174659, percentual: 55.39}`	Strict EA fails, relaxed EA + sem. match pass (presentation penalty)

score as correct. A deployment that ranks systems on strict accuracy alone would prefer the system that projects fewer columns over the one that answers more questions, so it should report relaxed Execution Accuracy and a semantic-match metric alongside strict accuracy.

The same agent reaches 63.56% strict Execution Accuracy on BIRD-PT and 15.01% on ANEEL, a gap of at least 36 percentage points on every metric. Part of the gap is measurable, since removing the evidence hints that BIRD provides drops the BIRD agent from 70.86% to 63.04% relaxed accuracy, still far above ANEEL's 27.77%. The benchmarks also differ in schema scale and in whether descriptions are curated or extracted automatically, and these are not varied independently, so the residual gap is consistent with schema scale and metadata quality rather than traceable to one cause. For this system, the BIRD-PT score did not predict ANEEL performance, which argues for evaluating on a domain benchmark.

The correct table reaches the top five for only about 60% of ANEEL questions, against nearly all BIRD questions, and the qualitative cases show wrong-table and wrong-column selection among the agent's errors. Retrieval is a genuine bottleneck on this large, Portuguese, automatically-described schema. The table and column descriptions extracted from PDF data dictionaries are the factor most directly tied to this failure mode.

Running over the full corpus, the agent issues several database calls per question for retrieval, optional exploration, and execution, which raised database load and produced timeouts under batch and concurrent use. We have not quantified the per-query token, latency, and call costs, and report that as future work. Structured exploration raised the user-facing metrics more than strict accuracy at the 140-table scope, but the agent without the typed channel can still inspect data through ordinary SQL and the smaller-scope exploration runs predate a dispatch fix, so we treat that benefit as preliminary.

8. Conclusion

This work is a deployment evaluation case study of an agentic Text-to-SQL system over ANEEL open data, a 140-table Brazilian-Portuguese regulatory database. It asks whether

an academic benchmark can substitute for this domain and whether execution accuracy is sufficient to compare deployed systems.

The same agent reaches 63.56% strict Execution Accuracy on BIRD-PT and 15.01% on ANEEL, a gap of at least 36 percentage points on every metric under the same model. Retrieval recall is near 100% on BIRD but about 60% on ANEEL, so the academic benchmark score did not predict performance on this deployment.

Strict Execution Accuracy ranks the baseline above the agent at every schema scope, while relaxed Execution Accuracy and the semantic-match judges rank the agent above it, because the agent's answers are often correct but penalized by exact matching. Reporting strict accuracy alone would invert the system choice, so deployed evaluations should report row-level and answer-level metrics together. For Text-to-SQL deployed over a large regulatory database, the choice of benchmark and of metrics shapes the conclusions as much as the system does.

This study has limitations. The benchmark contains only single-table queries, so the findings do not extend to join-heavy workloads. All systems use one model family, Gemini-3-Flash for generation with gemini-2.5-flash and claude-opus-4-7 as semantic-match judges, so the results are a case study rather than a model-independent claim. Semantic Match is judged by these models and not yet validated against human ratings. The exploration comparison does not fully isolate exploration, since the agent can inspect data through ordinary SQL when the typed channel is off. Per-query cost in tokens, latency, and database calls is not quantified.

9. Acknowledgments

This work has been funded by P&D CEMIG/ANEEL PD-04950- D0677/2023 supported by the Center of Excellence in Artificial Intelligence (CEIA), with financial resources from the MCTI grant number 057/2023, signed with EMBRAPII. This work was also supported by the National Institute of Science and Technology (INCT) in Responsible Artificial Intelligence for Computational Linguistics and Information Treatment and Dissemination (TILD-IAR) grant number 408490/2024-1.

References

Cen, J., Liu, J., Li, Z., and Wang, J. (2025). Sqlfixagent: Towards semantic-accurate text-to-sql parsing via consistency-enhanced multi-agent collaboration. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 49–57.

de Carvalho, L. F. C., Júnior, P. S. d. S., and de Oliveira, H. T. A. (2025). Benchmarking Large Language Models for Text-to-SQL in Brazilian Portuguese and English. In *Simpósio Brasileiro de Tecnologia Da Informação e Da Linguagem Humana (STIL)*, pages 101–112. SBC.

Floratou, A., Psallidas, F., Zhao, F., Deep, S., Hagleither, G., Tan, W., Cahoon, J., Alotaibi, R., Henkel, J., Singla, A., Grootel, A. V., Chow, B., Deng, K., Lin, K., Campos, M., Emani, K. V., Pandit, V., Shnayder, V., Wang, W., and Curino, C. (2024). Nl2sql is a solved problem... not! In *Conference on Innovative Data Systems Research*.

Li, H., Zhang, J., Li, C., and Chen, H. (2023). Resdsql: Decoupling schema linking and skeleton parsing for text-to-sql. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 13067–13075.

Li, J., Hui, B., Qu, G., Yang, J., Li, B., Li, B., Wang, B., Qin, B., Geng, R., Huo, N., et al. (2024). Can llm already serve as a database interface? a big bench for large-scale database grounded text-to-sqls. *Advances in Neural Information Processing Systems*, 36.

Nobre, L., Sousa, S., Teles, S., and Soares, A. (2026). Language effects in text-to-SQL across English and Portuguese. In Souza, M., de Dios-Flores, I., Santos, D., Freitas, L., Souza, J. W. d. C., and Ribeiro, E., editors, *Proceedings of the 17th International Conference on Computational Processing of Portuguese (PROPOR 2026) - Vol. 1*, pages 270–280, Salvador, Brazil. Association for Computational Linguistics.

Pham, K. T., Nguyen, T. H., Jo, J., Nguyen, Q. V. H., and Nguyen, T. T. (2025). Multilingual Text-to-SQL: Benchmarking the Limits of Language Models with Collaborative Language Agents.

Pourreza, M., Li, H., Sun, R., Chung, Y., Talaei, S., Kakkar, G. T., Gan, Y., Saberi, A., Ozcan, F., and Arik, S. O. (2024). CHASE-SQL: Multi-Path Reasoning and Preference Optimized Candidate Selection in Text-to-SQL.

Pourreza, M. and Rafiei, D. (2023). DIN-SQL: Decomposed In-context learning of text-to-SQL with self-correction. *Advances in Neural Information Processing Systems*, 36:36339–36348.

Shkapenyuk, V., Srivastava, D., Johnson, T., and Ghane, P. (2025). Automatic Metadata Extraction for Text-to-SQL.

Song, Y., Ezzini, S., Tang, X., Lothritz, C., Klein, J., Bissyandé, T., Boytsov, A., Ble, U., and Goujon, A. (2024). Enhancing text-to-sql translation for financial system design. In *Proceedings of the 46th International Conference on Software Engineering: Software Engineering in Practice*, pages 252–262.

Talaei, S., Pourreza, M., Chang, Y.-C., Mirhoseini, A., and Saberi, A. (2024). CHESS: Contextual Harnessing for Efficient SQL Synthesis.

Wang, B., Ren, C., Yang, J., Liang, X., Bai, J., Chai, L., Yan, Z., Zhang, Q.-W., Yin, D., Sun, X., et al. (2025). Mac-sql: A multi-agent collaborative framework for text-to-sql. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 540–557.

Wang, B., Shin, R., Liu, X., Polozov, O., and Richardson, M. (2020). Rat-sql: Relation-aware schema encoding and linking for text-to-sql parsers. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 7567–7578.

Yu, T., Zhang, R., Yang, K., Yasunaga, M., Wang, D., Li, Z., Ma, J., Li, I., Yao, Q., Roman, S., et al. (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.

Zhong, R., Yu, T., and Klein, D. (2020). Semantic evaluation for text-to-SQL with distilled test suites. In Webber, B., Cohn, T., He, Y., and Liu, Y., editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 396–411, Online. Association for Computational Linguistics.