Modular Enterprise Natural Language to SQL Using Small Language Models

Rodrigo Fernandes and Ricardo Malheiro
University of Leiria and Oeste, Portugal

2240537@ulo.pt
ricardo.malheiro@ulo.pt

Abstract: Enterprise organisations increasingly store critical data in relational databases, but business users often lack the SQL expertise to query them. This problem is acute in ERP/CRM systems with large, poorly documented schemas, multilingual requests, and constraints that may preclude proprietary cloud models. This work evaluates a modular Natural Language-to-SQL architecture built from small, locally deployable open-weight models. Its stages are query normalisation, intent classification, embedding-based schema retrieval with graph expansion, and LoRA-fine-tuned skeleton generation and query completion. A 9-billion-parameter model achieved over 80% recall across all intent classes, without consistent gains from larger models or ensembles. A 0.6-billion-parameter embedding model matched larger alternatives for broad candidate retrieval. In an isolated gold-grounded evaluation, the 9B skeleton model achieved 84.2% execution accuracy on the Spider test comparison split; a 0.8B model reached 80.3%. The 9B checkpoint achieved only 34.9% zero-shot on BIRD, with failures concentrated in structurally complex queries. Translation of noisy multilingual ERP metadata produced a median cosine similarity above 0.79. An end-to-end test achieved 26.0% execution accuracy with retrieved tables and 82.0% with correct tables, identifying schema retrieval as the dominant bottleneck. Although the components can run locally, this 50-query evaluation on one ERP demonstration database does not yet establish production readiness.

Keywords: Natural language-to-SQL, Small language models, Enterprise AI, Schema retrieval, On-premises AI

1. Introduction

Enterprise organisations are increasingly data-rich, yet business users cannot readily access relational data because extraction requires manually written SQL. They must therefore rely on data engineers, delaying access and separating decision-makers from the data.

This problem is harder in enterprises than in typical NL2SQL benchmarks. Enterprise Resource Planning (ERP) and Customer Relationship Management (CRM) systems can contain thousands of cryptically named, poorly documented tables. Requests may be multilingual, while privacy, cost, and data-sovereignty constraints can preclude sending schema metadata to proprietary cloud models.

Most NL2SQL approaches do not jointly address these constraints. They typically rely on very large general-purpose models, often through proprietary APIs, and assume that the full schema fits in context. Neither assumption suits enterprises: full-schema prompting does not scale, and external closed models can conflict with data-sovereignty requirements. Schema-linking and fine-tuning techniques (Section 2) offer partial solutions but rarely combine schema scale, multilinguality, and on-premises deployment.

This raises our central research question: can a modular NL2SQL architecture built exclusively from small, open-weight large language models (LLMs) that run locally perform competitively under these enterprise constraints without relying on the largest proprietary models?

The pipeline therefore comprises four independently evaluable and swappable stages: query normalisation, intent classification, embedding-based schema retrieval with graph expansion, and two-stage LoRA-fine-tuned SQL generation (Hu et al., 2021). This decomposition avoids relying on one large model's emergent capabilities.

This paper makes three contributions. First, it proposes a modular enterprise NL2SQL architecture supporting schema scale, multilingual input, and private on-premises inference. Its schema-retrieval method combines semantic search with foreign-key graph expansion. Second, results on Spider, BIRD, and noisy multilingual ERP metadata show that specialised small models can approach substantially larger models and ensembles at individual stages. Third, an end-to-end evaluation of 50 held-out ERP-derived queries propagates outputs through every stage and identifies schema retrieval as the dominant bottleneck (26.0% cascading execution accuracy versus 82.0% with oracle tables). This is evidence from one ERP demonstration database, not an estimate of production readiness.

2. Related Work

This section reviews the literature most relevant to the proposed architecture: LLM-based text-to-SQL, model specialisation and parameter-efficient fine-tuning, and evaluation in enterprise-like settings.

LLM-based Text-to-SQL. Large language models have become the dominant paradigm for Text-to-SQL, replacing earlier rule-based and sequence-to-sequence approaches; Zhan et al. (2025) survey this shift, noting a move towards prompt engineering, schema representation, and fine-tuning, evaluated mainly on Spider and BIRD. Li et al. (2025) propose CM-SQL, reconciling two schema representations across separate LLMs on Spider; Liu and Chu (2025) combine BERT-based retrieval, GPT-4 generation, and graph neural networks for schema modelling on enterprise-style queries. Both illustrate the trend of combining specialised retrieval and reasoning components with LLMs.

Schema linking and schema retrieval. Because enterprise schemas can far exceed a model's usable context, schema linking and retrieval have become central to recent work. Lan and Li (2024) propose PS-SQL, linking query phrases to table/column names; Liu and Zhang (2025) extend RESDSQL with graph-attention encoding and a fusion layer sharing context between schema linking and skeleton parsing. Both, however, are evaluated on Spider-scale schemas; neither addresses retrieval over hundreds of overlapping-vocabulary tables typical of ERP systems, nor exploits foreign-key graph structure (Newman, 2003) to expand a candidate set.

SQL skeleton generation and multi-stage pipelines. Decomposing SQL generation into multiple stages is an established strategy for constraining the model's search space. Wen et al. (2025) separate a schema-linking model from a downstream generation model, cutting GPU memory usage by 40% with negligible accuracy loss; Jiang et al. (2025) propose FGCSQL, a three-stage schema-filter/generate/correct pipeline; Long and Cao (2021) decompose Text2SQL into template-population sub-tasks. These works support the structural/schema-grounding separation adopted here, though none combine it with graph-based expansion or enterprise-scale schema noise.

PEFT/LoRA and locally deployed models. Parameter-efficient fine-tuning, and LoRA in particular, is well established for adapting language models to domain-specific tasks at a fraction of full fine-tuning's cost. Mao et al. (2025) survey LoRA variants and their efficiency gains; Razuvayevskaya et al. (2024) show LoRA can match or exceed full fine-tuning on multilingual classification at much lower cost — motivating LoRA for adapting small (0.8–9B) open models to SQL generation here.

Positioning of this work. The reviewed literature shows that schema linking, multi-stage generation, and PEFT are independently effective and can benefit from larger or ensembled models. However, most studies use Spider (Yu et al., 2018), whose schemas are not representative of enterprise conditions. BIRD (Li et al., 2023) provides an initial check against this gap in Section 5. Largely absent is an architecture that (a) combines embedding-based retrieval with graph expansion at schema scales an order of magnitude larger than Spider, (b) tests whether small (sub-10B) open models, individually fine-tuned per stage, can substitute for large proprietary pipelines, and (c) is evaluated beyond a single academic benchmark. This paper addresses (a) and (b) directly; for (c), Section 4 evaluates schema retrieval and translation on real ERP metadata, and Section 5 reports a zero-shot BIRD check and an end-to-end evaluation on 50 held-out enterprise queries, with broader validation left as future work (Section 6).

3. Proposed Modular Architecture

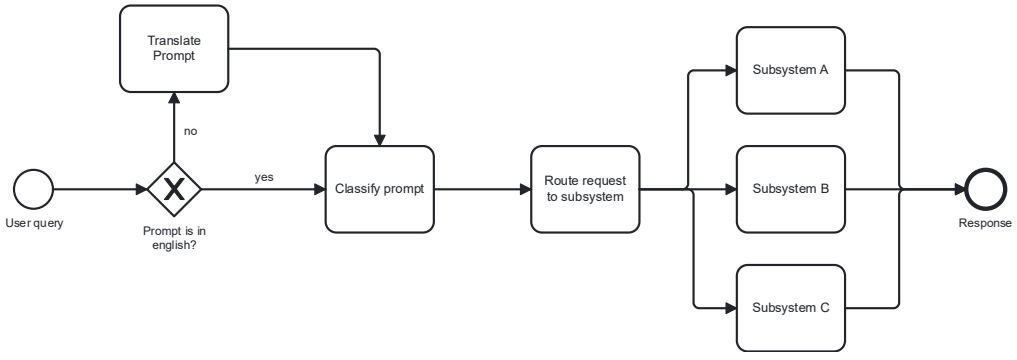

Figure 1: User-request lifecycle from the initial prompt through translation, intent classification, and routing

Rather than relying on a single large model to translate a natural language request directly into SQL, the proposed architecture decomposes the task into four specialised stages, each independently evaluable and replaceable (Figure 1).

Motivated by prior Turkish text-to-SQL work with schema context (Demirkıran et al., 2024), language normalisation detects the input language and translates non-English prompts into English, reducing downstream linguistic variability.

Once normalised, the request passes to an intent-classification stage, which determines whether it is a database query, another supported capability (e.g. image generation), or general conversation, and routes it accordingly; only database-query requests continue through the pipeline. This routing layer, inspired by the Query Classifier in the Monkuu framework (Yu et al., 2025), avoids invoking the full pipeline unnecessarily and provides a natural extension point for future capabilities.

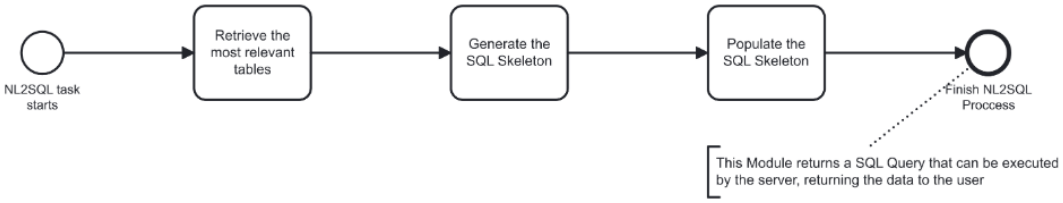

Figure 2: NL2SQL subsystem comprising schema retrieval, skeleton generation, and skeleton population

Requests classified as database queries (Figure 2) proceed to schema retrieval. Because exposing the full schema is impractical for ERP systems with hundreds or thousands of tables, the database is pre-processed offline. Qwen3.5-9B describes each table from its name, columns, and available sample rows. These descriptions are embedded and indexed for semantic top-k retrieval by cosine similarity.

Because semantic retrieval alone can miss tables needed for joins, the candidate set is expanded over a bidirectional graph of tables and foreign-key relationships via breadth-first search from the top-k tables, for up to h hops. The final list preserves the semantic ranking for initially retrieved tables and appends graph-expanded ones by increasing distance, capped at N candidates.

Finally, the resolved schema context and normalised request pass to a two-stage SQL-generation module. The first stage produces an abstract SQL skeleton — the logical structure (joins, filters, aggregations, grouping) — with identifiers replaced by placeholders; the second, a query-completion pass, grounds this skeleton in the concrete tables, columns, and values retrieved by schema retrieval, producing an executable statement. This follows the rationale of Graph-Fusion RESDSQL (Liu and Zhang, 2025): each stage is smaller and better-defined than end-to-end generation, and both can be independently fine-tuned and inspected for debugging.

Query execution, result validation, and access control (e.g. role-based access control) remain in the application layer. The NL2SQL subsystem generates SQL but does not enforce data-governance policy.

4. Experimental Methodology

This section describes the experimental setup: datasets, evaluation splits, metrics, and the ablation procedure used to isolate each design decision in Section 3's architecture.

Datasets. The pipeline is evaluated using two complementary sources of schema and query data. Spider (Yu et al., 2018) is a cross-domain semantic parsing benchmark; the release used here comprises four partitions — train_spider, train_others, dev, and test — totalling 206 SQLite databases and 11,840 question–SQL pairs, exceeding the commonly cited train+dev release (200 databases, 10,181 questions) because it includes the later test partition. Spider's domain diversity supports both training and, after merging per-database schemas into a single graph, evaluation of schema retrieval under controlled cross-domain ambiguity.

The second source, Mais Ritmo, contains two artefacts from an ERP demonstration database provided by vendor PHC and accessed through the GoGest client application; neither was used for training. The first comprises 472,404 Portuguese table- and field-level metadata entries across 8,960 tables, machine-translated into English for translation evaluation and schema-retrieval embeddings. The second contains 279 English natural-language/SQL pairs for one SQLite mirror (8,404 tables, 487,267 columns, and 438 declared relationships) and was evaluated once as the only fully independent ERP-derived test set (Section 5). Its gold SQL consists of real GoGest production queries that an LLM simplified into minimal equivalent forms; an LLM generated the

questions from those forms. Neither artefact has train/dev/test splits; both are used in full and available on request.

A purpose-built set of 851 labelled prompts (300 image-generation, 300 database-query, and 251 general/conversational) evaluates intent classification. BIRD (Li et al., 2023), a 69-database cross-domain benchmark, evaluates zero-shot generalisation only. No model was trained on its 9,428 training-split questions; 483 were excluded because their normalised gold SQL failed to execute against the corresponding database within 15 seconds, leaving 8,945. Finally, a fixed-seed sample of 50 Mais Ritmo pairs supports end-to-end evaluation (Section 5): each original Portuguese question passes through all stages, with every stage consuming its predecessor's actual output.

Models. The final end-to-end pipeline used Infomaniak-AI/vllm-translategemma-4b-it (Finkelstein et al., 2026) for translation, Qwen/Qwen3.5-9B for intent classification and table selection, Qwen/Qwen3-Embedding-0.6B for candidate retrieval, Qwen/Qwen3.5-9B with the selected skeleton-generation LoRA, and Qwen/Qwen3.5-4B with the selected query-completion LoRA. Both adapters were evaluated at training step 795 and used rank 32, alpha 64, and dropout 0.1. The locally loaded SQL-generation models used bfloat16 without quantisation and their native context limits; outputs were capped at 1,024 tokens for skeleton generation and 768 for completion. The sequential pipeline's largest resident model is the 9B generator (~18 GB bfloat16 weights, excluding runtime and KV-cache overhead). Intent-classification sweeps additionally evaluated Qwen3.5 and Llama-3.1/3.2 variants (Qwen Team, 2026; Llama Team, AI @ Meta, 2024). Schema-selection sweeps included Qwen3.5, Granite-4.1, mistralai/Mistral-7B-Instruct-v0.2, and google/gemma-4-E4B-it (Gemma Team, 2026). SQL-generation sweeps included Qwen3.5-0.8B/4B/9B, Qwen/Qwen2.5-Coder-3B-Instruct, ibm-granite/granite-4.1-3b-base, and google/gemma-3-1b-it (Gemma Team, 2025).

LoRA configuration. Generation models were fine-tuned with Low-Rank Adaptation under two regimes: baseline (rank 16, alpha 32, dropout 0.05) and expanded (rank 32, alpha 64, dropout 0.1), the latter adopted after an ablation (Section 5) showed consistent gains from the larger rank. Training data also passed through schema-name mangling to reduce vocabulary memorisation; a query-normalisation pass was applied in most configurations, and two augmentation axes — query paraphrasing and table-description variation — were evaluated independently.

Metrics. Each stage is evaluated with metrics suited to its output: recall per class for intent classification; full-match and partial-match rates for schema retrieval; reference-free cosine similarity and MetricX-24 scores (Juraska et al., 2024; 0 = perfect, 25 = poor) for translation; and execution accuracy — whether the generated SQL returns the same result as the reference — for SQL generation, evaluated on Spider's dev/test, Mais Ritmo, and, as a zero-shot check, BIRD. Because a skeleton must first populate into an executable query before being judged correct, execution accuracy is further decomposed into an execution rate and a correctness rate among executed predictions.

Experimental protocol. Experiments ran in reactive Marimo notebooks (Marimo, 2024). Because Spider development and test guided ablations and configuration selection, their results are comparison-set results, not unbiased generalisation estimates. Mais Ritmo and BIRD were each evaluated once with frozen checkpoints and fixed sampling seeds. Skeleton generation used greedy decoding; completion and translation used temperature 0. Intent classification and table selection retained the server's default decoding parameters, so not every stage was fully deterministic. Component experiments mainly used one NVIDIA RTX A6000 (48 GB); end-to-end evaluation used a Thunder Compute instance with two NVIDIA A100 GPUs (80 GB each). Stage latency and peak VRAM were not logged.

5. Results

This section reports the experimental findings, proceeding from component-level evaluation to end-to-end accuracy across the three evaluation splits, closing with ablations isolating each design decision from the previous section.

Intent classification. Model scale was not the primary driver of accuracy. With a 1-shot prompt, Qwen3.5-9B outperformed Qwen3.5-35B, matched Llama-3.1-70B, and was not improved by ensembling. It was therefore selected as the best-performing model for the final pipeline. Separately, Qwen3.5-4B with 3-shot prompting and temperature 0 produced the most balanced per-class recall: 82.9% for general prompts, 100.0% for image generation, and 82.0% for database queries.

Schema retrieval. At top-k = 50, all embedding models retrieved at least one correct table at similar rates and outperformed TF-IDF; the smallest, Qwen3-Embedding-0.6B, slightly outperformed larger variants. Under the stricter full-schema-match criterion, Gemma-4-E4B-it led without graph expansion, achieving 70.5% full match and 89.1% partial match. After LLM relevance filtering, one-hop expansion increased Qwen3.5-4B's full-match accuracy from 64.7% to 82.7%, compared with 87.2% for Gemma-4-E4B-it; gains diminished beyond one or two hops. These results are oracle-aided upper bounds. Because the evaluation merges multiple Spider schemas, identically named tables are mapped to graph nodes using the gold table set, which is unavailable at inference time. The results show expansion's potential under correct disambiguation, not expected deployment performance.

Translation module. Metadata translation produced median cosine similarities of 0.792 for tables and 0.797 for fields; median MetricX-24 scores were 5.58 and 4.80, respectively, on its 0 (perfect)–25 (poor) scale. Schema identifiers were protected by design: only table and field descriptions were translated, while table and field_name values were copied unchanged. An audit of 472,254 well-formed aligned records confirmed exact preservation of both identifiers in every record. Runtime prompt translation was assessed separately in the 50-query end-to-end test using the character n-gram F-score (chrF; higher is better), with a mean of 79.3. chrF did not correlate positively with table-retrieval recall (Pearson r = −0.152); empty retrievals had higher, not lower, mean chrF than non-empty retrievals (83.3 versus 77.8). This sample provides no evidence that lower translation quality drove retrieval failure, although its size and automatic metrics cannot exclude terminology-level effects.

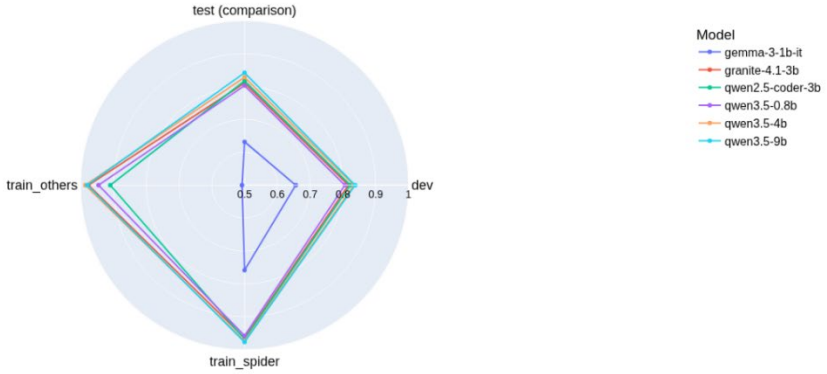

Figure 3: Execution accuracy of the best LoRA-fine-tuned skeleton-generation run for each base model across the Spider splits (train_spider, train_others, dev, and test comparison split). All models use schema mangling and LoRA rank 32. Performance was similar across splits

SQL generation. SQL generation is decomposed into two independently trained and evaluated stages. The first, skeleton generation, converts the normalised request into an abstract SQL structure with identifiers replaced by placeholders (Section 3). After LoRA fine-tuning on Spider, the 9B model achieved 84.20% execution accuracy on test and 83.72% on dev. The 0.8B model reached 80.30%, only 3.90 points lower despite having less than one-tenth as many parameters. Ablations showed that increasing the LoRA rank and alpha improved accuracy, while schema-name mangling outperformed the unmangled condition. Query paraphrasing caused a reproducible regression (82.39% versus 84.20%), whereas table-description variation was neutral to slightly beneficial (84.10%) (Figure 3).

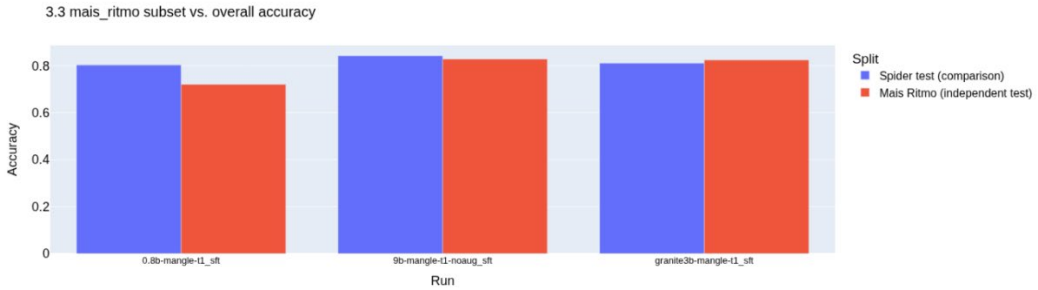

Figure 4: Execution accuracy of the three skeleton-generation checkpoints under gold-grounded query completion on the independent Mais Ritmo ERP-derived query set (n = 279) and the Spider test comparison split (n = 2,101)

For the independent ERP-derived evaluation, the Qwen3.5-9B, Qwen3.5-0.8B, and Granite-4.1-3B skeleton checkpoints were run on Mais Ritmo (n = 279), an ERP-derived query set from a real demonstration database (Figure 4). The Qwen3.5-9B and Granite-4.1-3B checkpoints remained within 1.4 points of their Spider-test accuracy, while Qwen3.5-0.8B dropped 8.26 points. This suggests that while sub-1B models are competitive on Spider-scale queries, robustness to the noisier ERP-derived distribution tracks model size more closely than raw Spider accuracy does.

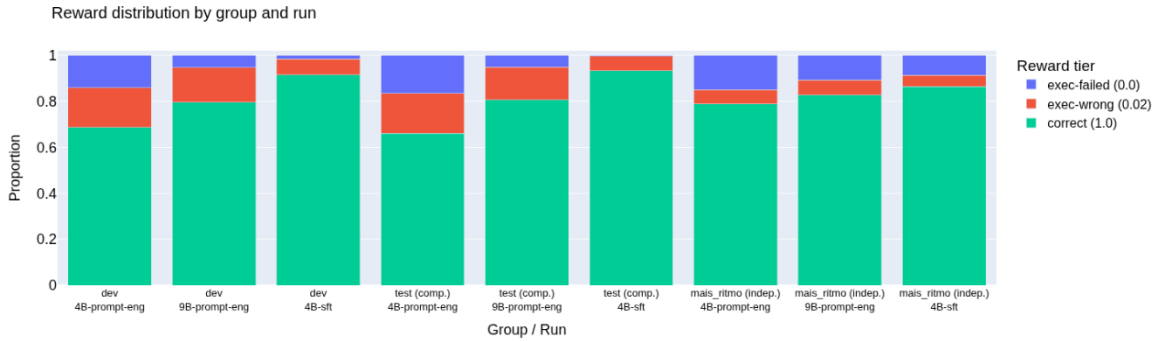

Figure 5: Query-completion outcomes with gold skeletons, tables, and columns on Spider dev/test and Mais Ritmo

Query completion was evaluated in isolation using gold skeletons, tables, and columns, thereby excluding upstream errors. The LoRA-fine-tuned 4B model achieved 93.38% execution accuracy on the Spider test split, compared with 80.63% for the prompt-engineered 9B baseline and 66.06% for the 4B baseline (Figure 5). The ranking was unchanged on Mais Ritmo, although the fine-tuned model's accuracy fell to 86.38%. Fine-tuning therefore transferred to the ERP-derived set, but less strongly than to Spider.

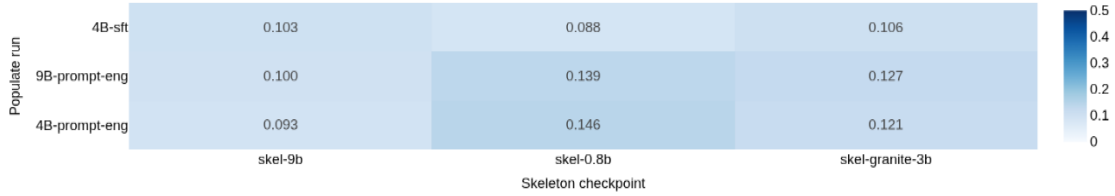

Figure 6: Jaccard similarity between failures of query-completion runs (rows) and skeleton checkpoints (columns). Overlap remains below 0.5, indicating mostly distinct failure sets

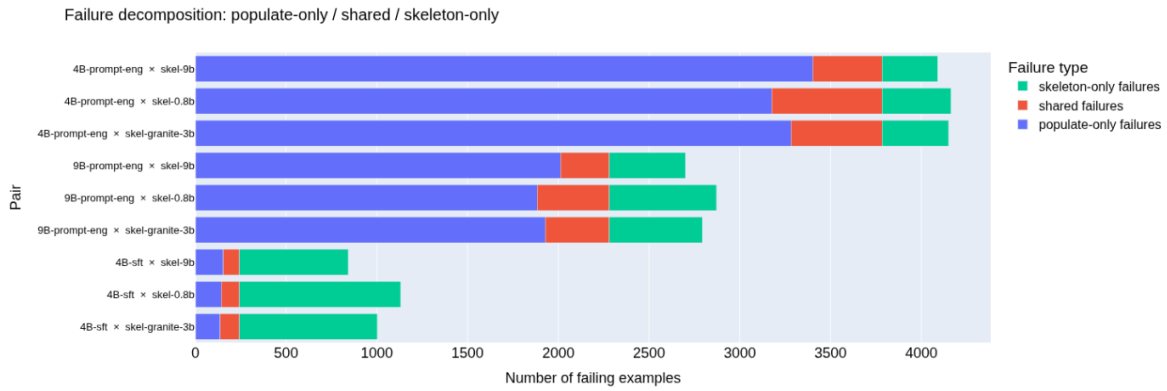

Figure 7: Failure decomposition for each completion run × skeleton checkpoint pair. Prompt-engineered completions fail more broadly; the fine-tuned 4B model fails less often, but its remaining failures overlap disproportionately with skeleton failures

A failing-overlap analysis compares each query-completion run against the three skeleton-generation checkpoints via the Jaccard similarity of their failing-example sets (Figure 6): overlap is modest throughout, indicating skeleton- and completion-stage failures are largely distinct. The direction flips, however, between prompt-engineered and fine-tuned completions (Figure 7): once completion is fine-tuned well enough to fix

most 'easy' errors, its remaining error budget concentrates disproportionately on examples also hard for skeleton generation — a minority compounding-error mode consistent with Section 6.

Generalisation to BIRD. The best skeleton checkpoint—Qwen3.5-9B with mangled schema names and no augmentation, which achieved 84.2% on the Spider test split—was evaluated without BIRD-specific training on 8,945 valid queries from 69 databases. Under the same gold-grounded protocol, execution accuracy fell to 34.9%. Only 67.1% of skeletons produced an error-free populated query, down from 93.5% on Spider, and 52.1% of those executable queries were correct.

This drop concentrates in specific SQL constructs rather than spreading uniformly across BIRD (Figure 8): accuracy is lowest for nested subqueries, DISTINCT, aggregation, and GROUP BY, degrading steadily with join count. Literal value references have almost no effect — since gold values are already supplied when populating the skeleton, the gap lies in structural reasoning, not value grounding.

These results indicate that the skeleton-generation model, trained exclusively on Spider, does not transfer well to BIRD — better explained by BIRD containing more structurally complex queries than by a generic distributional gap.

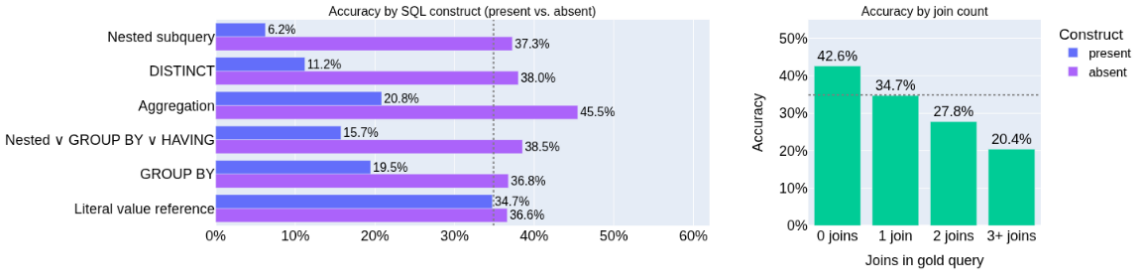

Figure 8: Zero-shot BIRD execution accuracy of Qwen3.5-9B (n = 8,945), by SQL construct and join count. The dotted line marks overall accuracy (34.9%); nested subqueries, DISTINCT and aggregation show the largest drops

End-to-end evaluation. The 50-query Mais Ritmo subset (Section 4) ran through the full pipeline twice: cascading, with each stage consuming the previous stage's output, and with oracle schema retrieval using gold tables. All translations succeeded (mean chrF 79.3) and all requests were classified correctly. Schema retrieval was the main bottleneck (mean precision 0.10; recall 0.36): the first selection returned no tables for 14/50 queries, leaving graph expansion without a seed. Empty selections were not concentrated in queries requiring more tables: they occurred in 13/35 single-table queries, compared with 1/15 queries requiring two or three tables. Thus, table count alone does not explain retrieval failure. The ERP schema presents two plausible difficulties. First, many tables use short or cryptic identifiers, providing limited lexical signal for semantic retrieval. Second, only 438 declared relationships are available across 8,404 tables, so graph expansion cannot traverse undeclared links or recover from an empty initial selection. These observations indicate that failures arise before or during initial semantic selection, while sparse graph connectivity limits subsequent recovery; the experiment does not isolate the causal contribution of either mechanism. These failures propagated into execution accuracy: 13/50 cascading queries executed correctly (26.0%), compared with 41/50 under oracle retrieval (82.0%). With gold tables, skeleton-generation accuracy increased from 30.0% to 70.0%, and population accuracy from 98.0% to 100.0%.

6. Discussion and Limitations

Across all four pipeline stages, the results point to the same pattern: performance is governed more by task-specific specialisation than by raw model scale. Intent classification showed no consistent scale effect; schema retrieval depended more on graph-based expansion than embedding-model size; and SQL-generation accuracy degraded gracefully as the base model shrank from 9B to 0.8B parameters — consistent with the literature claim that specialisation can substitute for scale in narrow tasks (Section 2).

The end-to-end run demonstrates that the pipeline can execute on privately controlled GPU infrastructure without proprietary model APIs. The experiment used two NVIDIA A100 80 GB GPUs for batched evaluation, not as a minimum inference requirement. Because stages execute sequentially, the largest model has an approximately 18 GB bfloat16 weight footprint. This weight footprint is within the nominal 24–32 GB capacity of consumer GPUs such as the RTX 4090 or 5090, before runtime and KV-cache overhead. These cards were not

benchmarked, and peak runtime VRAM, KV-cache requirements, and stage latency were not recorded; practical single-GPU performance therefore remains to be measured.

The results also do not demonstrate operational production readiness. The only complete pipeline evaluation comprises 50 queries from a single ERP demonstration database, and its 26.0% cascading execution accuracy shows that substantial improvement is required before deployment in a production decision-support setting. The evaluation has several further limitations. Schema-retrieval experiments used a single graph merging Spider's per-database schemas; while this introduces useful cross-domain ambiguity, Spider schemas remain smaller and less densely interconnected than a real ERP system. As explained in Section 5, the graph-expansion figures use oracle-aided table-name disambiguation and are therefore upper bounds rather than deployment estimates. Translation was assessed with one model and automatic metrics; the identifier audit verifies exact pass-through rather than terminological accuracy, and the 50-query correlation is too small to exclude translation-related retrieval effects.

Finally, the Section 5 end-to-end evaluation directly identifies empty first-pass selection as a major failure mode and shows that table count alone does not explain it. Cryptic table identifiers and sparse declared relationships are plausible constraints on semantic selection and graph-based recovery, but the experiment does not isolate their individual causal contributions; moreover, with only 50 queries, its rates should be read as indicative rather than precise.

Spider's dev and test splits also informed model and configuration selection throughout development, so Spider-based figures are comparison-set results rather than unbiased estimates; the independent evidence here rests on the Mais Ritmo and BIRD evaluations, each run exactly once on frozen checkpoints.

A further limitation concerns cross-benchmark generalisation: as Section 5 details, the skeleton checkpoint's strong Spider performance (84.2%) does not transfer to BIRD (34.9%), most plausibly because Spider under-represents the structurally complex queries — nested subqueries, DISTINCT, multi-way joins, grouped aggregation — that dominate BIRD's harder cases.

7. Conclusion

This work set out to determine whether a modular NL2SQL architecture, built entirely from small, open-weight, locally deployable language models, can meet the demands of enterprise deployment — large, poorly documented schemas, multilingual input, and data-sovereignty constraints — without depending on the largest proprietary models.

Evidence gathered across four independently evaluated stages, plus an end-to-end test on held-out enterprise queries, gives both component-level and whole-pipeline support: intent classification, schema retrieval, and SQL generation each showed that task-specific specialisation, rather than scale, drives performance (Section 5).

Propagating the pipeline's own outputs end-to-end exposes schema retrieval as the largest source of error. Only 26.0% of 50 held-out queries executed correctly with retrieved tables, compared with 82.0% when correct tables were supplied; generation given those tables degraded only modestly. Generation itself is not enterprise-ready either: the same skeleton checkpoint, evaluated zero-shot on BIRD, reached only 34.9%, with the largest drops on structurally complex queries — so both schema retrieval and out-of-distribution generation remain open problems, schema retrieval being the larger gap here.

Taken together, the results show that modularity and per-stage specialisation can support locally deployable, privacy-sovereign multilingual NL2SQL components without relying on one very large model. The complete system is not production-ready: its only end-to-end evaluation used 50 queries from one ERP demonstration database and achieved 26.0% execution accuracy. Future work should evaluate the complete pipeline across independent production-scale ERP schemas and query sets, target schema retrieval directly — its 28% empty-selection rate is the highest-leverage lever identified so far — and close the BIRD transfer gap on structurally complex queries through targeted fine-tuning, data augmentation, or execution-feedback-driven query repair.

Ethics Declaration: No human participants or personal data were involved. ERP metadata came from a vendor demonstration database, not production customer data.

AI tool usage Declaration: Large language models generated synthetic table descriptions for schema retrieval. Claude (Anthropic) was also used as a coding assistant and to review the manuscript for consistency.

References

Demirkıran, F., Coşkun, A.K., Kömeçoğlu, Y., Buluz Kömeçoğlu, B. and Güven, R., 2024. Enhancing text-to-SQL conversion in Turkish: an analysis of LLMs with schema context. In: UBMK 2024 – 9th International Conference on Computer Science and Engineering. IEEE, pp.255–259. https://doi.org/10.1109/UBMK63289.2024.10773403

Finkelstein, M., Caswell, I., Domhan, T., Peter, J.-T., Juraska, J., Riley, P., Deutsch, D., Kovacs, G., Dilanni, C., Cherry, C., Briakou, E., Nielsen, E., Luo, J., Black, K., Mullins, R., Agrawal, S., Xu, W., Kats, E., Jaskiewicz, S., Freitag, M. and Vilar, D., 2026. TranslateGemma Technical Report. arXiv preprint arXiv:2601.09012. https://doi.org/10.48550/arXiv.2601.09012

Gemma Team, 2025. Gemma 3. [online] Kaggle. Available at: https://goo.gle/Gemma3Report [Accessed 30 July 2026].

Gemma Team, 2026. Gemma 4 Technical Report. arXiv preprint arXiv:2607.02770. https://doi.org/10.48550/arXiv.2607.02770

Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L. and Chen, W., 2021. LoRA: low-rank adaptation of large language models. arXiv preprint arXiv:2106.09685. https://doi.org/10.48550/arXiv.2106.09685

Jiang, G., Li, W., Yu, C., Zhu, Z. and Li, W., 2025. FGCSQL: a three-stage pipeline for large language model-driven Chinese text-to-SQL. Electronics, 14(6), p.1214. https://doi.org/10.3390/electronics14061214

Juraska, J., Deutsch, D., Finkelstein, M. and Freitag, M., 2024. MetricX-24: the Google submission to the WMT 2024 metrics shared task. In: Proceedings of the Ninth Conference on Machine Translation. Association for Computational Linguistics, pp.492–504. https://doi.org/10.18653/v1/2024.wmt-1.35

Lan, Z. and Li, S., 2024. PS-SQL: phrase-based schema-linking with pre-trained language models for text-to-SQL parsing. In: 2024 6th International Conference on Natural Language Processing (ICNLP). IEEE, pp.31–35. https://doi.org/10.1109/ICNLP60986.2024.10692964

Li, J., Hui, B., Qu, G., Yang, J., Li, B., Li, B., Wang, B., Qin, B., Geng, R., Huo, N., Zhou, X., Ma, C., Li, G., Chang, K.C.C., Huang, F., Cheng, R. and Li, Y., 2023. 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, pp.42330–42357. https://doi.org/10.48550/arXiv.2305.03111

Li, X., You, J., Li, H., Peng, J., Chen, X. and Guo, Z., 2025. CM-SQL: a cross-model consistency framework for text-to-SQL. Neurocomputing, 658, p.131708. https://doi.org/10.1016/j.neucom.2025.131708

Liu, J. and Chu, S., 2025. Text2SQL business intelligence system based on retrieval-augmented generation (RAG). Engineering Reports, 7(6). https://doi.org/10.1002/eng2.70249

Liu, Y. and Zhang, T., 2025. Graph-Fusion RESDSQL: enhancing NL2SQL robustness with dynamic schema linking and contextual integration. In: 2025 IEEE 7th International Conference on Communications, Information System and Computer Engineering (CISCE). IEEE, pp.221–225. https://doi.org/10.1109/CISCE65916.2025.11065889

Llama Team, AI @ Meta, 2024. The Llama 3 herd of models. arXiv preprint arXiv:2407.21783. https://doi.org/10.48550/arXiv.2407.21783

Long, H. and Cao, D., 2021. BERT-based text-to-SQL generation method with question-table content enhancement and template filling. In: 2021 2nd International Conference on Information Science and Education (ICISE-IE). IEEE, pp.969–973. https://doi.org/10.1109/ICISE-IE53922.2021.00222

Mao, Y., Ge, Y., Fan, Y., Xu, W., Mi, Y., Hu, Z. and Gao, Y., 2025. A survey on LoRA of large language models. Frontiers of Computer Science, 19(7), pp.1–19. https://doi.org/10.1007/s11704-024-40663-9

Marimo, 2024. marimo: a reactive Python notebook. [online] Available at: https://marimo.io [Accessed 30 July 2026].

Newman, M.E.J., 2003. The structure and function of complex networks. SIAM Review, 45(2), pp.167–256. https://doi.org/10.1137/S003614450342480

Qwen Team, February 2026. Qwen3.5: towards native multimodal agents. [online] Available at: https://qwen.ai/blog?id=qwen3.5 [Accessed 30 July 2026].

Razuvayevskaya, O., Wu, B., Leite, J.A., Heppell, F., Srba, I., Scarton, C., Bontcheva, K. and Song, X., 2024. Comparison between parameter-efficient techniques and full fine-tuning: a case study on multilingual news article classification. PLoS ONE, 19(5), p.e0301738. https://doi.org/10.1371/journal.pone.0301738

Wen, W., Zhang, Y., Pan, S., Sun, Y., Lu, P. and Ding, C., 2025. LR-SQL: a supervised fine-tuning method for Text2SQL tasks under low-resource scenarios. Electronics, 14(17), p.3489. https://doi.org/10.3390/electronics14173489

Yu, C., Yao, Y., Zhang, X., Zhu, G., Guo, Y., Shao, X., Shibasaki, M., Hu, Z., Dai, L., Guan, Q. and Shibasaki, R., 2025. Monkuu: a LLM-powered natural language interface for geospatial databases with dynamic schema mapping. International Journal of Geographical Information Science. https://doi.org/10.1080/13658816.2025.2533322

Yu, T., Zhang, R., Yang, K., Yasunaga, M., Wang, D., Li, Z., Ma, J., Li, I., Yao, Q., Roman, S., Zhang, Z. and Radev, D., 2018. Spider: a large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, pp.3911–3921. https://doi.org/10.18653/v1/D18-1425

Zhan, G., Shi, G., Liu, X. and Wang, N., 2025. Natural language to SQL using large language models. In: 2025 21st International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD). IEEE, pp.495–502. https://doi.org/10.1109/ICNC-FSKD67701.2025.11198017