Journal of Engineering Science and Technology
Special Issue on INCITEST2025 Vol. 21, No. 2 (2026) 9 - 16

EVALUATING GEMINI AND GEMMA LANGUAGE MODELS FOR INDONESIAN TEXT-TO-SQL TASKS

GALIH HERMAWAN[1,*], EDNAWATI RAINARLI[1],
RADJABOVA INOBAT RAVSHANOVNA[2]

[1]Universitas Komputer Indonesia, Bandung, Indonesia
[2]Qarshi State University, Uzbekistan
*Corresponding Author: galih.hermawan@email.unikom.ac.id

Abstract

The Text-to-SQL project converts plain human language to SQL to make databases more accessible to everyone, but there are few Indonesian resources; this research provides an Indonesian Text-to-SQL benchmark and tests modern language models. We developed a benchmark of 60 queries across six complexity types on two databases (Sakila and a cooking Resep database) and tested four models - Gemini 2.5 Flash, Gemini 2.5 Flash Lite, Gemma-3-27B-IT, and the locally run Gemma-3n-E2B, with measures of execution success, row-match accuracy, speed (ms), and token usage. The model (Gemini 2.5 Flash) achieved 100% execution success and 95% row-match correctness, with an average response time of 4,707 ms. With a moderate accuracy drop to 75%, Flash Lite reduced the delay to 1,249 ms. With 5,684 ms, Gemma-3-27B-IT achieved 81.7-82% of its accuracy. With a latency of 159,818 ms, the CPU only Gemma-3n-E2B achieved 21.7-22% accuracy. Models hosted on the cloud are highly accurate at interactive delays, although small local models running on the CPU are not. We conclude that Gemini 2.5 Flash is the most reliable overall, while Flash Lite offers a practical speed-accuracy compromise. To support reproducibility and further research impact, we release the benchmark and evaluation pipeline and outline extensions (conversational contexts, retrieval-augmented prompting, quantization, and caching) to improve the viability of lightweight local deployments.

Keywords: Gemini, Gemma, Indonesian NLP, Large language models, Text-to-SQL.

1. Introduction

Natural language database interfaces have gradually developed from rule-based approaches to neural methods to make data more accessible. Yet some of the most fundamental challenges related to schema linking and compositional reasoning still remain unsolved [1-4]. Meanwhile, Indonesian NLP is also growing fast. The studies on conversational agents and scene-text understanding show that there has been improvement in the recognition of different types of text inputs [5, 6]. The use of contextual representation models increases the robustness of Indonesian NLP tasks [7]. Effective semantic alignment between natural language and structured representations is crucial for reliable automated interpretation [8]. When viewed collectively, these studies bring out the need for dependable database interfaces in regional data ecosystems which is rising.

Earlier studies are not consistent in a single interpretation, but repeated insights could be utilized for the present study. Survey Evidence has demonstrated that Text-to-SQL systems fail when schema grounding is poor or ignored [2]. Accuracy alone can be misleading as high scores may not necessarily indicate reliabilities across different complexity of schemas and thus models cannot reliably be compared unless evaluated in standard way [9-12]. Furthermore, surface level matching studies have failed to capture the intrinsic semantic mismatches; it has been cautioned that structural similarity should not be disregarded, and more recent work point out that linguistic stability should not be overlooked for the evaluation of model robustness [13-15]. Taken together, such results suggest that performance claims should be considered in light of accuracy and service execution time [9].

Despite achieving these advances, there exist significant gaps, as there was no reusable benchmark resource in previous Indonesian work [16, 17]. To fill such a gap, we propose the first 60-query benchmark covering the schemata [18]. Our research makes three fundamental contributions. First, it represents the first openly accessible, multi-domain Indonesian Text-to-SQL benchmark that considers six analytical domains. Second, it presents an in-depth analysis of state-of-the-art large-scale language models like Gemini 2.5 and Gemma 3 in an integrated manner. Third, it presents efficiency baselines to model deployment viability through analysis of trade-offs of execution accuracy, delay, and tokens. All these aspects are expected to contribute towards rigor and clarity in any future endeavours in the realm of Indonesian Text-to-SQL research.

2. Methodology and Experimental Setup

This study employs a unified workflow that integrates benchmark creation, model prompting, and evaluation. The benchmark contains 60 Indonesian Text-to-SQL queries that represent six analytical categories across the Sakila and Resep schemas, each paired with a gold SQL statement and expected outputs to reflect the range of compositional challenges discussed in prior evaluations of Text-to-SQL systems [2, 11].

A single standardized prompt template was used for all models. It includes the Indonesian query, schema DDL summaries, selected example tuples, and explicit instructions for producing executable SQL. Deterministic decoding with temperature set to 0.0 and top-p set to 1.0 ensured reproducibility. The models

evaluated were Gemini 2.5 Flash, Gemini 2.5 Flash Lite, Gemma-3-27B-IT, and the compact Gemma-3n-E2B deployed locally through Ollama in CPU-only mode (see Fig. 1).

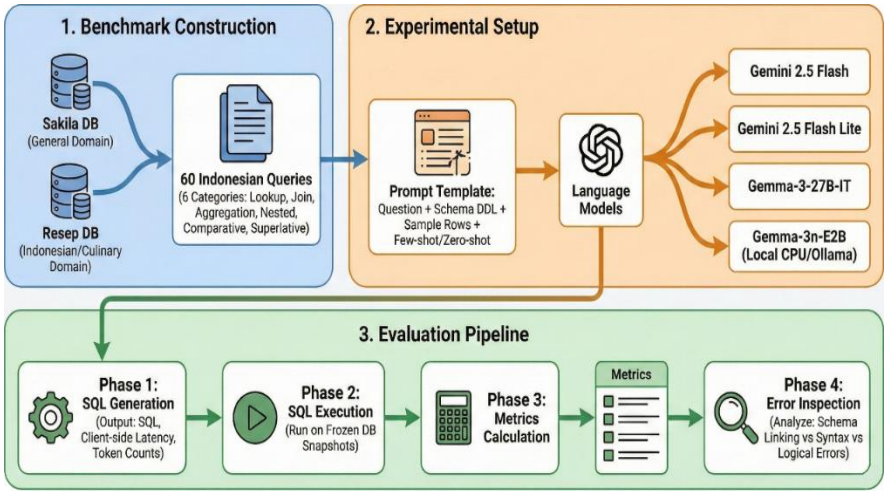

Fig. 1. Methodology and evaluation pipeline.

Each generated SQL output was examined using execution success, row-match accuracy, and structural similarity based on AST comparison [13]. Efficiency was assessed through mean and P95 client-side latency as well as total token usage, following computational considerations outlined in recent systems-level analyses [19, 20]. Error cases were manually categorized into schema-linking, structural, and logical types, consistent with the analytical practice described in earlier work on Text-to-SQL robustness [10, 21]. Proportions were reported with Wilson ninety-five percent confidence intervals, while latency metrics employed bootstrap intervals with ten thousand resamples.

3. Results and Discussion

3.1. Overview of experimental runs

In total, 240 evaluations were performed on four models and sixty scenarios. Global results are summarized in Table 1, which reports execution success, row-match accuracy, structural similarity, latency, and token usage with corresponding ninety-five percent confidence intervals. Gemini 2.5 Flash achieved perfect execution success and maintained the highest accuracy, while Flash Lite delivered lower accuracy with substantially reduced latency. Gemma-3-27B-IT reached moderate accuracy but showed higher latency in its CPU setting. The smallest model, Gemma-3n-E2B, had the lowest reliability and highest latency, confirming that compact CPU-only deployments suffer serious performance limitations. All these results point to the fact that cloud models execute flawlessly and predictably, whereas local models cannot overcome hardware throughput.

Table 1. Global performance with 95% confidence intervals (N=60 scenarios per model).

Model	Gemini 2.5 Flash	Gemini 2.5 Flash Lite	Gemma-3-27B-IT	Gemma-3n-E2B
Execution Success (%)	100.0% (94.0% - 100.0%)	96.7% (88.6%-99.1%)	95.0% (86.3%-98.3%)	70.0% (57.5%-80.1%)
Row-Match Accuracy (%)	95.0% (86.3%-98.3%)	75.0% (62.8%-84.2%)	81.7% (70.1%-89.4%)	21.7% (13.1%-33.6%)
SQL Similarity (%)	62.3% (60.0%-64.4%)	60.8% (58.3%-63.2%)	62.8% (60.4%-65.0%)	51.2% (47.4%-54.4%)
Latency Mean (ms)	4,707 (3,874-5,724)	1,249 (1,175-1,330)	5,684 (5,344-6,045)	159,818 (140,518-190,504)
Latency P95 (ms)	9,935 (8,001-13,266)	1,725 (1,529-2,141)	7,983 (6,983-9,081)	188,687 (178,950-247,637)
Total Tokens (avg.)	3,452 (3,237-3,689)	2,901 (2,784-3,020)	2,637 (2,521-2,753)	1,556 (1,493-1,621)

Notes: Proportions use Wilson 95% CI. Continuous variables use percentile bootstrap 95% CI with 10,000 resamples. Latency is reported as mean and P95 (both with 95% CI) in the same column. Total tokens are the sum of prompt and completion per query.

3.2. Evaluation pipeline and error inspection

Performance across the six query categories diverges noticeably. The cloud models handle simple tasks with ease, and Gemini 2.5 Flash reaches 100% in both Lookup and Aggregation, a level the smaller models never approach. Once compositional reasoning is involved, the gap becomes difficult to ignore: Flash Lite falls to 70% on Nested queries, while Gemma-3n-E2B drops to 0% in Aggregation and only 10% in Nested tasks, underscoring how severely compact CPU-bound models still struggle beyond basic retrieval.

3.3. Efficiency evaluation

Table 1 shows a clear speed gap: Gemini 2.5 Flash Lite is about 3.8 times faster than Flash, reducing average latency from 4,707 ms to 1,249 ms, although its accuracy drops from 95.0% to 75.0%. Token usage stays relatively stable between cloud models (3,452 vs. 2,901 tokens), indicating that token count is not the main driver of latency at this scale. Local models are limited almost entirely by hardware. Gemma-3-27B-IT, despite processing fewer tokens, still averages 5,684 ms, and Gemma-3n-E2B performs worst, requiring 159,818 ms for only 1,556 tokens, which highlights the severe constraints of CPU-only inference.

3.4. Detailed error analysis

Across all models, most failures stem from logical rather than syntactic errors. Gemini 2.5 Flash remains the most reliable, reaching 95.0% success with only 5.0% row mismatches and no execution failures. Flash Lite follows with 75.0% success, where 21.7% of its errors are logical and only 3.3% involve execution failure. Local

models show a far weaker profile: Gemma-3-27B-IT achieves 81.7% success but still records 5.0% execution failure, while Gemma-3n-E2B performs the worst with 30.0% execution failure and 48.3% logical errors. These outcomes indicate that compact CPU-bound models often struggle to produce syntactically valid and logically consistent SQL without stronger schema guidance.

3.5. Domain-specific performance

A comparison of execution accuracy and latency across the Resep and Sakila databases shows that Gemini 2.5 Flash and Gemini 2.5 Flash Lite generalize well across both schemas. Execution success remains consistent for both models: 100% across domains for Gemini 2.5 Flash and 97% on both schemas for Flash Lite. Latencies also show overlapped confidence intervals, meaning that the processing speed does not degrade independently of schema complexity. Similarly, Gemma-3-27B-IT reached high accuracy with 93% on Resep and 97% on Sakila without significant latency deviation between the two databases. In contrast, the smallest local model tested, Gemma-3n-E2B, shows marked sensitivity to domain complexity. Execution success in this model significantly drops from 90 percent on the simpler Resep database to 50 percent on the more complex Sakila schema. When Sakila queries are run, latency also increases substantially, and this rise is supported by confidence intervals that do not overlap. This pattern indicates that large cloud models do not struggle to generalize across domains, whereas compact local models that depend on CPU resources are not able to manage higher schema complexity and therefore cannot avoid considerable performance degradation.

3.6. Discussion and practical implications

Gemini 2.5 Flash is the most dependable instrument in our setup, as it is able to perform perfect execution success and highest row-match accuracy while the latency is still in an interactive range. Its profile is in line with previous Text-to-SQL research that execution fidelity and schema grounding are the main factors determining the practical utility and is also corroborated by meta-analytic work which points out schema interpretation as a recurring issue in natural language interfaces for databases [2, 10, 22, 23]. As a compromise, Gemini 2.5 Flash Lite can be used. Its response times of about 1.2 to 1.3 seconds are still good for interactive use, although its slight drop in accuracy hints that you cannot get more efficiency without some loss of fidelity. The pattern is in agreement with systems studies which show that decoder speed and batched processing together limit the number of requests that can be served concurrently.

Conversely, the local models do not avoid severe throughput constraints, and their performance reveals that CPU-only execution cannot sustain interactive latency. Although Gemma-3-27B-IT reaches an accuracy level that is not far from Flash Lite, it cannot match the latter's responsiveness, since its mean latencies fall between 5.3 and 5.9 seconds, confirming that CPU inference is not sufficient for timely execution. Surveys on efficient LLM serving further show that latency cannot be attributed simply to token counts, because hardware characteristics often exert greater influence than sequence length alone [19, 20]. The smallest model, Gemma-3n-E2B, does not escape these limitations and produces the lowest row-match accuracy together with the highest latency. If such logical mismatches are not addressed, future local deployment cannot remain viable, which is why

techniques such as retrieval augmented generation or supervised approaches that separate schema linking from SQL generation become essential for improving reliability under constrained hardware [24-27].

These observations have a few practical implications: cloud endpoints are suitable when the key drivers are accuracy and time to value, because of their high execution fidelity and predictable latency across schemas [10, 22, 28-30]. Compact variants like Flash Lite are suitable when responses below one second or of low latency are needed and small reductions in accuracy are tolerable. Only local deployment with hardware acceleration and alignment or retrieval techniques can make sense, because end-to-end latency is greatly determined by time to first token and time per output token [19, 31-33].

4. Conclusion

This study systematically evaluated four LLMs (Gemini 2.5 Flash, Gemini 2.5 Flash Lite, Gemma-3-27B-IT, and Gemma-3n-E2B) on 60 Indonesian Text-to-SQL scenarios across the Sakila and Resep databases. Gemini 2.5 Flash achieved 100% execution success and 95% row-match at interactive latency, while Flash Lite delivered roughly one third of the latency with a modest accuracy drop. Gemma-3-27B-IT reached comparable accuracy with higher CPU latency, and the local Gemma-3n-E2B lagged significantly in accuracy and exceeded practical latency. We release the first open Indonesian Text-to-SQL benchmark and a reproducible evaluation pipeline that clarifies accuracy, latency, and cost trade-offs for cloud versus local serving. Limitations include two schemas, 60 scenarios, CPU-only local tests, and fixed prompts. In practice, Flash Lite suits latency-sensitive cloud use, while viable local deployment requires GPU acceleration or targeted tuning; future work will expand scenarios and explore retrieval-augmented prompting, parameter-efficient tuning, and quantization and caching to improve local models.

5. Acknowledgements

This study was conducted by the AkaBot Research Group with support from Universitas Komputer Indonesia.

References

1. Abbas, S.; Khan, M.U.; Lee, S.U.J.; Abbas, A.; and Bashir, A.K. (2022). A review of NLIDB with deep learning: Findings, challenges and open issues. *IEEE Access*, 10, 14927-14945.
2. Affolter, K.; Stockinger, K.; and Bernstein, A. (2019). A comparative survey of recent natural language interfaces for databases. *The VLDB Journal*, 28(5), 793-819.
3. Katsogiannis-Meimarakis, G.; and Koutrika, G. (2023). A survey on deep learning approaches for text-to-SQL. *The VLDB Journal*, 32(4), 905-936.
4. Zhang, W.; Wang, Y.; Song, Y.; Wei, V.J.; Tian, Y.; Qi, Y.; Chan, J.H.; Wong, R.C.W.; and Yang, H. (2024). Natural language interfaces for tabular data querying and visualization: A survey. *IEEE Transaction on Knowledge and Data Engineering,* 36(11), 6699-6718.

5. Luckyardi, S.; Karin, J.; Rosmaladewi, R.; Hufad, A.; and Haristiani, N. (2024). Chatbots as digital language tutors: Revolutionizing education through AI. *Indonesian Journal of Science and Technology*, 9(3), 885-908.
6. Rainarli, E.; Suprapto, S.; and Wahyono, W. (2024). Corner pixel-based method for selecting binary text in scene. *Asia-Pacific Journal of Information Technology and Multimedia*, 13(2), 221-231.
7. Yunanto, R.; Wibowo, E.P.; and Rianto, R. (2023). A BERT model to detect provocative hoax. *Journal of Engineering Science and Technology*, 18(5), 2281-2297.
8. Nursikuwagus, A.; Munir, R.; Khodra, M.L.; and Dewi, D.A. (2024). Model semantic attention (SemAtt) with hybrid learning separable neural network and long short-term memory to generate caption. *IEEE Access*. 12, 154467-154481.
9. Gao, D.; Wang, H.; Li, Y.; Sun, X.; Qian, Y.; Ding, B.; and Zhou, J. (2024). Text-to-SQL empowered by large language models: A benchmark evaluation. *VLDB Endowment,* 17(5), 1132-1145.
10. Li, B.; Luo, Y.; Chai, C.; Li, G.; and Tang, N. (2024). The dawn of natural language to SQL: Are we fully ready? *VLDB Endow,* 17(11), 3318-3331.
11. Jiang, P.; Cai, X.; Jiang, P.; and Cai, X. (2024). A survey of semantic parsing techniques. *Symmetry*, 16(9), 1201-1248.
12. Ascoli, B.G.; Kandikonda, Y.S.R.; and Choi, J.D. (2025). ETM: Modern insights into perspective on text-to-SQL evaluation in the age of large language models. *Future Internet*, 17(8), 325-340.
13. Pinna, G.; Perezhohin, Y.; Manzoni, L.; Castelli, M.; and De-Lorenzo, A. (2025). Redefining text-to-SQL metrics by incorporating semantic and structural similarity. *Scientific Reports*, 15, 22357-22366.
14. Azurmendi, I.; Zulueta, E.; García, G.; Uriarte-Arrazola, N.; Lopez-Guede, J. M.; Azurmendi, I.; Zulueta, E.; García, G.; Uriarte-Arrazola, N.; and Lopez-Guede, J.M. (2025). Beyond standard losses: Redefining text-to-SQL with task-specific optimization. *Mathematics*, 13(14), 2315-2337.
15. Hazboun, F.H.; Owda, M.; and Owda, A.Y. (2021). A natural language interface to relational databases using an online analytic processing hypercube. *AI*, 2(4), 720-737.
16. Admojo, F.T.; Lajis, A.; and Nasir, H. (2023). Systematic literature review on ontology-based Indonesian question answering system. *Knowledge Engineering and Data Science*, 6(2), 129-144.
17. Kurniawan, A.; Abdiansah, A.; and Utami, A.S. (2024). NL2SQL for chatbot with semantic parsing using rule-based methods. *Sriwijaya Journal of Informatics and Applications*, 5(1), 15-23.
18. Adora, C. (2021). Designing and using a MySQL database for human resource management. *Advances in Science, Technology and Engineering Systems Journal*, 4(6), 285-290.
19. Miao, X.; Oliaro, G.; Zhang, Z.; Cheng, X.; Jin, H.; Chen, T.; and Jia, Z. (2025). Towards efficient generative large language model serving: A survey from algorithms to systems. *ACM Computing Surveys,* 58(1), 1-37.

20. Tay, Y.; Dehghani, M.; Bahri, D.; and Metzler, D. (2022). Efficient Transformers: A Survey. *ACM Computing Serveys,* 55(6), 1-28.
21. Fan, Y.; Weng, Q.; Chen, Y.; and Wang, X.S. (2025). Rethinking data in NL2SQL: A survey of what we have and what we expect. *Vicinagearth*, 2(1), 15-26.
22. Shi, L.; Tang, Z.; Zhang, N.; Zhang, X.; and Yang, Z. (2025). A survey on employing large language models for text-to-SQL tasks. *ACM Computing Surveys*, 58(2), 1-37.
23. Kanburoğlu, A.B.; and Tek, F.B. (2024). Text-to-SQL: A methodical review of challenges and models. *Turkish Journal of Electrical Engineering and Computer Sciences*, 32(3), 403-419.
24. Klesel, M.; and Wittmann, H.F. (2025). Retrieval-augmented generation (RAG). *Business and Information Systems Engineering*, 67(4), 551-561.
25. Wang, Z.; Zhu, M. X.; Li, G.; and Kong, S. (2025). Retrieval-augmented Chinese text-to-SQL generation for conversational bibliographic search. *Plos One*, 20(10), e0334965-e0334974.
26. Wen, W.; Zhang, Y.; Pan, S.; Sun, Y.; Lu, P.; Ding, C.; 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), 3489-3508.
27. Han, M.; Park, S.; Kim, S.; and Kim, H. (2024). Bridging the gap between text-to-SQL research and real-world applications: A unified all-in-one framework for text-to-SQL. *Knowledge-Based Systems*, 306, 112697-112713.
28. Gugulothu, M. (2025). Securing data endpoints in google cloud: A data-driven approach to anomaly detection and threat mitigation. *Journal Publication of International Research for Engineering and Management (JOIREM)*, 5(6), 1-4.
29. Manchana, R. (2020). Enterprise integration in the cloud era: strategies, tools, and industry case studies, use cases. *International Journal of Science and Research (IJSR)*, 9(11), 1738-1747.
30. Biswas, S. (2025). Artificial intelligence-enhanced cybersecurity frameworks for real-time threat detection in cloud and enterprise. *ASRC Procedia: Global Perspectives in Science and Scholarship*, 1(01), 737-770.
31. Mijuskovic, A.; Chiumento, A.; Bemthuis, R.; Aldea, A.; and Havinga, P. (2021). Resource management techniques for cloud/fog and edge computing: An evaluation framework and classification. *Sensors*, 21(5), 1832-1841.
32. Muttaqin, H.R.; and Setiyadi, A. (2024). Cloud computing development of PT Divistant Teknologi Indonesia CRM system using Microsoft azure. *International Journal of Informatics, Information System and Computer Engineering (INJIISCOM)*, 5(2), 276-287.
33. Hadiana, A.I. (2020). Fog computing architecture for indoor disaster management. *International Journal of Informatics, Information System and Computer Engineering (INJIISCOM)*, 1(1), 79-90.