Computational Systems and Artificial Intelligence
Cutting-Edge Scientific Solutions

e-ISSN: 3108-4079
REVIEW ARTICLE
Vol.2 / No.1 / 2026 / pp.1-6
https://doi.org/10.69882/adba.csai.2026011

A Comprehensive Review of LLM-based Text-to-SQL Systems: Methods, Datasets, and Trends

Garima Singh[*,1], **Prachi Chhabra**[α,2] **and Tathagat Banerjee**[β,3]
[*,α]Department of Information Technology JSS Academy of Technical Education, Noida, India, [β]Department of Computer Science and Engineering, Indian Institute of Technology Patna, India.

ABSTRACT Translation of Natural Language to SQL queries (i.e. Text-to-SQL or NL2SQL) helps the user to easily access the relational database. It also helps in various commercial applications. In recent years, the development of Large Language has increased the performance of the NL2SQL system. It enhances the semantic understanding, schema linking, and SQL generation, even for complex and cross-domain queries. This paper reviews recently published research papers between 2018 - 2025, focusing on LLM-based methods for Text-to-SQL tasks. We examine the system pipelines covering pre-processing, translation, and post-processing stages, along with commonly used datasets and tools. We also discuss advances in schema linking, reasoning-based query generation, and the use of retrieval-augmented generation for providing additional context. Based on the surveyed literature, we summarize key trends, challenges, and future directions, aiming to provide an accessible overview for students and researchers interested in LLM-based NL2SQL systems.

KEYWORDS
SQL
Natural language processing (NLP)
LLM
Relational database

INTRODUCTION

One important method for making it easier for users to access the vast amounts of data kept in relational databases is the conversion of natural language to SQL code. For non-experts, this text-to-SQL task is a huge help. Now days, data-driven choices are increasingly important in both business and research. However, everyone's access to data is complicated and slowed down by the requirement for SQL expertise. Conventional methods like rule based, pattern matching and older neural configurations have advanced significantly. They continue to encounter issues with ambiguous language, complex database structures, and the variety of SQL features that arise in real-world scenarios. Large language models have revolutionized this field, particularly since 2023. GPT-4, Codex, and other models.

RELATED WORKS

Recent research on LLM-based NL2SQL systems demonstrates rapid progress in model architectures, pipeline design, and evaluation strategies. Survey studies provide comprehensive taxonomies and highlight the transition from PLM-based approaches to agentic and retrieval-augmented LLM frameworks, while system-oriented works emphasize workload adaptation, cost efficiency, and tighter AI–database integration. Despite significant improvements in execution accuracy and semantic understanding, current methods still face challenges related to scalability, deployment cost, schema evolution, and robustness to real-world ambiguity. Consequently, future research directions focus on efficient model customization, explainability, adaptive query generation, semantic error handling, and enterprise-ready NL2SQL deployment. To understand the pros, cons and research gaps of the existing models, we have summarised the key recent papers in the Table 1 as given below.

TAXONOMY OF LLM-BASED TEXT-TO-SQL METHODS

A precise taxonomy is necessary to map the developments in this field methodically. Pre-processing, translation, and post-processing are the three main modules that make up the lifecycle of contemporary Text-to-SQL systems.

A user writes the question in English Natural Language. The

Manuscript received: 8 November 2025,
Revised: 13 December 2025,
Accepted: 7 January 2026.

[1] garimasingh11203@gmail.com
[2] prachichhabra@jssateb.ac.in
[3] tathagat_25s21res125@iitp.ac.in (**Corresponding author**).

Table 1 Overview of recent NL2SQL papers with advantages, limitations, and future research directions

Paper	Key Contributions	Limitations	Open Research Directions
Comprehensive Survey of NL2SQL with LLMs (2025) (Liu *et al.* 2025)	Provides a lifecycle-level survey covering preprocessing, translation, and postprocessing stages; presents a detailed taxonomy of LLM-based NL2SQL architectures; offers multi-perspective evaluation and a practical deployment roadmap.	Predominantly focuses on LLM-era approaches with limited discussion of pre-LLM systems; scalability and deployment cost issues remain underexplored.	Development of cost-efficient LLM pipelines; robust handling of schema evolution; explainable and trustworthy NL2SQL systems; support for open-domain querying.
Next-Generation Database Interfaces: Survey of LLM-based Text-to-SQL (2024) (Hong *et al.* 2025)	Analyzes the transition from PLMs to LLMs; provides in-depth discussion of graph-based RAG techniques; highlights advances in schema linking and dynamic context retrieval.	High computational overhead of RAG pipelines; performance sensitivity to retrieval quality and data freshness.	Real-time graph maintenance; multilingual NL2SQL systems; explainable query generation; scalable graph construction techniques.
Employing LLMs for Text-to-SQL Tasks: Taxonomy with Prompt Engineering and Fine-tuning (2024) (Shi *et al.* 2025)	Introduces a clear taxonomy distinguishing prompt-based and fine-tuning-based approaches; discusses domain adaptation challenges and data scarcity issues.	Limited evidence of large-scale fine-tuning in real-world settings; unresolved privacy and governance concerns.	Efficient synthetic data generation; enhanced domain knowledge integration; privacy-preserving and compliant training strategies.
LLM-based NL2SQL with Distilled Customization Approach (Oracle Labs, 2025) (Liu *et al.* 2025; Corradini *et al.* 2025)	Demonstrates that distilled models can achieve near teacher-level performance with significantly fewer parameters; proposes modular customization components.	Strong dependence on powerful teacher LLMs; challenges in tuning complexity and cross-domain generalization.	Lightweight domain-adaptive NL2SQL models; automated distillation workflows; continual learning mechanisms.
TailorSQL: Workload-Tailored NL2SQL System (Amazon, 2025) (Liu *et al.* 2025; Hong *et al.* 2025; Rao *et al.* 2023)	Adapts NL2SQL systems to specific query workloads, improving efficiency and accuracy; integrates retrieval-augmented generation.	Limited generalization beyond targeted workloads; reduced flexibility across diverse domains.	Joint workload-aware and multi-domain modeling; incorporation of user feedback for continuous system improvement.
Text2SQL Is Not Enough: Unifying AI and Databases (2025) (Zhang and Zhang 2025)	Emphasizes tight AI–database integration; demonstrates the advantages of graph reasoning for complex analytical queries.	High system complexity; limited practical techniques for joint AI–database optimization.	Deeper AI–DB co-design; interactive and conversational query interfaces; robust multi-turn dialogue systems.
ASKSQL: A Cost-Effective NL2SQL Pipeline (2025) (Liu *et al.* 2025; Rao *et al.* 2023)	Proposes a pipeline optimized for cost and latency; combines lightweight retrievers with LLM-based SQL generators.	Trade-offs between accuracy and efficiency; lack of large-scale real-world deployment studies.	Adaptive cost–accuracy optimization; real-time index and retriever updates; enterprise deployment strategies.
Fine-Tuning Text-to-SQL Models with Reinforcement Learning (2025) (Zhong *et al.* 2017; Rao *et al.* 2023)	Improves execution accuracy using reward-driven fine-tuning guided by execution feedback; robust across diverse SQL structures.	Computationally expensive training; requires careful reward function design.	Scalable RL-based fine-tuning; integration with prompt- and retrieval-based methods; human-in-the-loop optimization.
Benchmark for Semantic Error Detection in NL2SQL (2025) (Rao *et al.* 2023; Lin *et al.* 2022)	Introduces benchmarks targeting semantic correctness beyond syntactic validity; proposes datasets for detecting and analyzing semantic errors.	High complexity of real-world semantic errors; limited automated correction capabilities.	Automated semantic error repair; explainable error diagnostics; multi-domain semantic robustness.
Leveraging LLMs for Adaptive Query Generation (2025) (Liu *et al.* 2025; Shi *et al.* 2025; Rao *et al.* 2023)	Proposes self-adaptive query generation using feedback loops and reranking strategies; improves robustness to query variation.	Efficiency bottlenecks; sensitivity to prompt formulation.	Efficient adaptation mechanisms; unified retrieval–generation frameworks; real-time user feedback integration.
End-to-End Text-to-SQL with Dataset Selection (2025) (Liu *et al.* 2025; Rao *et al.* 2023; Diallo *et al.* 2023)	Demonstrates the impact of systematic dataset selection on training and fine-tuning; improves generalization by reducing data noise.	Dataset bias persists; limited automation in dataset curation.	Automated dataset synthesis and expansion; domain-balanced training; continuous dataset updating.

question is then formatted into the prompt by the system. Now the formatted prompt is used by LLM model to understand the question. Then the model uses the pre defined schemas, dataset queries & filter the relevant information such as tables and queries. Now the LLM model uses prompt, schemas & provided dataset to generate an output SQL query that the user demands for. The output SQL query is then run on the system & the result is generated as tables, charts or some follow up questions are suggested (Liu *et al.* 2025; Shi *et al.* 2025; Hong *et al.* 2025; Mohammadjafari *et al.* 2025).

The following Figure 1 shows the taxonomy, as discussed in the above literature survey:

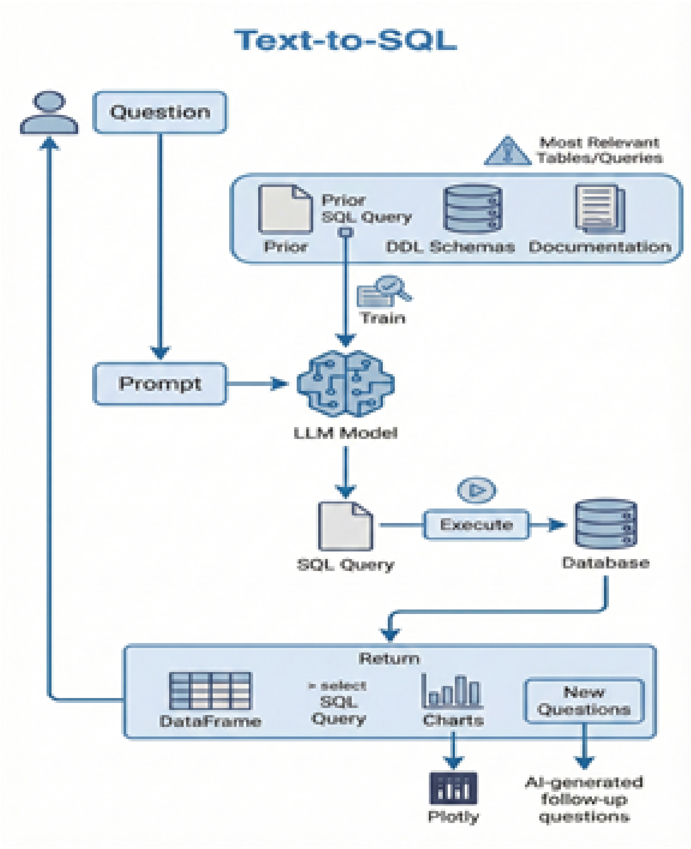

Figure 1 Taxonomy of Text-to-SQL

Research strategies ranging from breaking down intricate pipelines for error analysis and optimization to benchmarking innovations like schema augmentation, agent-based correction, and hybrid retrieval-generation workflows are supported by this modular view (Liu *et al.* 2025; Shi *et al.* 2025; Hong *et al.* 2025; Mohammadjafari *et al.* 2025).

Dataset Evolution

As per the data available the dataset did not only evolved in size but also improved the quality of schemas, multi-turn dialogue, cross-domain coverage and real business problems.

In 2017 (Zhong *et al.* 2017) WikiSQL was the largest dataset available. It had only single table queries making it simpler than the later evolved dataset.

Url: https://github.com/salesforce/WikiSQL

In 2018 (Yu *et al.* 2018) Spider was evolved, it was much smaller & complex than WikiSQL. It had multiple table joins, foreign key reasoning & cross domain generalization.

Url:https://yale-lily.github.io/spider

CoSQL (Yu *et al.* 2019) in 2019 was an evolution of Spider with conversational and multi-turn queries. This dataset enabled models to handle context, have clarification & to have follow up collection.

Url: https://yale-lily.github.io/cosql

In 2023 (Rao *et al.* 2023; Diallo *et al.* 2023) BIRD was evolved with the business-based dataset. It focused on numerical reasoning, long schemas, domain-specific vocabulary and adversarial noise.

Url: https://bird-bench.github.io/

In 2024 (Corradini *et al.* 2025) BULL was evolved to challenge models with business query and real a world query workflow.

The following Figure 2 shows the growth of different NL2SQL datasets over the year 2017 - 2024.

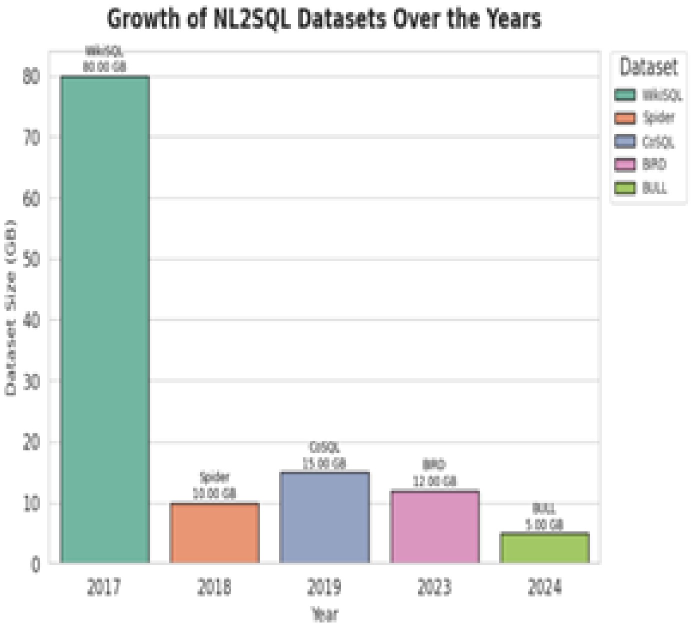

Figure 2 Timeline: Growth of Text-to-SQL Datasets (2017-2024)

Benchmark Dataset

To understand the performance of different Text-to-SQL models in real-world and to compare their accuracy across commonly used benchmarks. The Table 2 summarizes the models and the influence of their methods execution accuracy (Rao *et al.* 2023; Lin *et al.* 2022).

Pipeline Design and Key Toolkits

This section highlights about the pipeline of the existing NL2SQL. The NL2SQL model involves a lot of steps used for processing Natural Language into SQL queries which are discussed below and depicted in the Figure 3. Following are the steps involved for conversion of Natural Language to SQL queries (Liu *et al.* 2025; Shi *et al.* 2025; Hong *et al.* 2025; Mohammadjafari *et al.* 2025).

Step 1 : Data Collection And Pre-Processing

- Mostly use huge and publicly available datasets like WikiSQL (Rao *et al.* 2023), Spider (Yu *et al.* 2018), CoSQL (Guo *et al.* 2019), and BIRD (Diallo *et al.* 2023).
- Then the model is trained using the schemas, natural language & SQL answers present in the dataset.

Step 2 : Dataset Construction

Table 2 Summary of execution accuracy results for selected Text-to-SQL models on Spider and BIRD dataset benchmarks

Method / Model	Approach	Spider 1.0 (Exec. Acc.)	BIRD (Exec. Acc.)	Spider 2.0 (Exec. Acc.)	Key Insights
RESDSQL-3B + NatSQL (Rao *et al.* 2023; Scholak *et al.* 2021; Wang *et al.* 2020)	Fine-tuned PLM	83.9%	N/A	N/A	Strongest PLM-based method on Spider 1.0
DAIL-SQL (GPT-4) (Liu *et al.* 2025; Hong *et al.* 2025; Rao *et al.* 2023)	Prompt-based LLM	∼80.6%	66–82%	2.2%	Highlights the extreme difficulty of Spider 2.0
Claude-3.7 Sonnet (Liu *et al.* 2025; Shi *et al.* 2025; Rao *et al.* 2023)	In-context LLM	∼80–86%	∼80%	24.5–25.8%	State-of-the-art on most open benchmarks
Chat2DB-Agent + Claude-4 (Liu *et al.* 2025; Hong *et al.* 2025; Zhang and Zhang 2025)	Fine-tuned LLM	N/A	N/A	44.1%	Winner of Spider 2.0 / Snow benchmark
ByteBrain-Agent (Liu *et al.* 2025; Hong *et al.* 2025; Rao *et al.* 2023)	Multi-agent LLM	N/A	N/A	60.9%	Top-performing system on Spider 2.0 leaderboard
Gemini-2.0 Pro (Liu *et al.* 2025; Shi *et al.* 2025; Rao *et al.* 2023)	LLM API	N/A	N/A	13.9%	Moderate Spider 2.0 performance
GPT-4o (Liu *et al.* 2025; Shi *et al.* 2025; Rao *et al.* 2023)	LLM API	86.6%	N/A	10.1–12.9%	Best on Spider 1.0, weaker on Spider 2.0
SQLCoder (Defog AI, 2024) (Liu *et al.* 2025; Rao *et al.* 2023; Pourreza and Rafiei 2023)	Open-source LLM	$< 80\%$	N/A	$< 15\%$	Strong logical reasoning; struggles with complex schemas

- cite the supporting literature completely rather than select a subset of citations;
- provide important background citations, including relevant review papers (to help orient the non-specialist reader);
- to cite similar work in other chaos theory and applications.

Step 3 : Open Source And Closed Source LLM model The dataset constructed in the previous step is fed into two types of LLM models - Open Source LLM models - These models are fully customizable as per our requirements such as LLAMA2.

Closed Source LLM models - These model can be accessed only through their APIs such as ChatGPT.

Step 4 : Training Dataset To train the model PEFT techniques like LoRA/QLoRA has been used. They also fine tune the open source model efficiently. DeepSeed Optimization speeds up the training and reduce the memory usage.

Step 5 : Output Prediction In this step natural language has been taken as an input question and which will generates SQL queries as output.The predictor uses Adaptive parallelism to generate efficient queries and Reranking or refinement to identify the best SQL query.

Step 6 : Performance And Evaluation The performance of the output SQL queries is evaluated using three primary metrics: Execution Accuracy (EX) measures result consistency between predicted and gold queries; Exact Match (EM) checks if the query strings are identical; and Yield-Expected-Semantic (YES) assesses their semantic equivalence. These are calculated as:

$$EX = \frac{1}{n}\sum_{i=1}^{n} I(\text{Exec}(Q_{\text{pred}}) = \text{Exec}(Q_{\text{gold}})) \tag{1}$$

$$EM = \frac{1}{n}\sum_{i=1}^{n} I(Q_{\text{pred}} = Q_{\text{gold}}) \tag{2}$$

$$YES = \frac{1}{n}\sum_{i=1}^{n} I(\text{Sem}(Q_{\text{pred}}) = \text{Sem}(Q_{\text{gold}})) \tag{3}$$

The overall architecture as shown in the Figure 3 describes the steps of the LLM-based Text-to-SQL systems, from dataset creation to training, prediction, and final assessment.

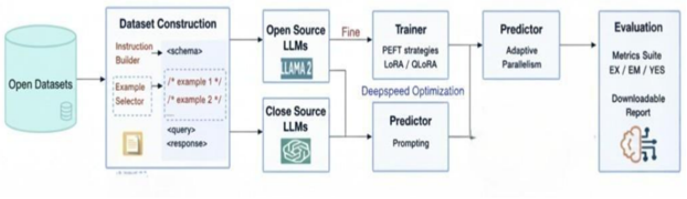

Figure 3 General Pipeline of Text-to-SQL

b) Latest And Popular Tools Used

In the following table an overview of the most popular tools in the recent years and their applications has been discussed (Liu *et al.* 2025; Hong *et al.* 2025; Zhang and Zhang 2025; Zhong *et al.* 2017; Diallo *et al.* 2023).

Challenges And Future Scope

One of the biggest issues in the Natural Language to SQL system is language ambiguity. Firstly, Those who are not familiar with the querying language often ask unclear and informal questions, making the system unable to understand the intent and generate the correct query. Secondly, Real-world databases often have many connected tables, making the database complex. It makes identifying the table, column for a query very difficult. Thirdly, having

Table 3 Popular Framework

Tool / Framework	Brief Description	Applications
DAIL-SQL (Liu *et al.* 2025; Hong *et al.* 2025; Rao *et al.* 2023)	GPT-4-based pipeline with dynamic prompt/data augmentation and hybrid reasoning	Spider / BIRD / real-world, few-shot / retrieval
MAC-SQL (Liu *et al.* 2025; Hong *et al.* 2025; Zhang and Zhang 2025)	Multi-agent (Selector, Decomposer, Refiner) with agentic SQL generation, correction, and refinement	Large schema, multi-step pipelines
DIN-SQL (Liu *et al.* 2025; Shi *et al.* 2025; Mohammadjafari *et al.* 2025)	Four-stage pipeline for schema linking, classification, SQL generation, and correction	Robust SQL generation, Spider, BIRD
CHASE-SQL (Hong *et al.* 2025; Rao *et al.* 2023)	Multi-agent, divide-and-conquer with CoT / pathways, iterative self-correction	Large / complex SQL, BIRD, Spider 2.0
SuperSQL (Shi *et al.* 2025; Lin *et al.* 2022)	Consistency-driven LLM with SQL majority voting and schema-aware reranking	Consistent SQL outputs
Alpha-SQL (Liu *et al.* 2025; Hong *et al.* 2025; Zhang and Zhang 2025)	Agent-based, planning-centric MCTS for strategic module activation, domain-agnostic	Autonomous, adaptive SQL generation
SQLfuse (Shi *et al.* 2025; Rao *et al.* 2023; Pourreza and Rafiei 2023)	Critic module, candidate reranking, schema linking with open-source LLMs, and few-shot learning	Open-source, BIRD, Spider

large schemas and poor documentation also affect the accuracy of the query generated. Last but not the least the publicly available datasets do not reflect the real-world business query thus making the model struggle with advanced operations such as nested queries, joins, and aggregations. Collecting the real-world data is costly, time-consuming, and slow. Therefore,pretrained models often do not produce a correct query for the unseen schemas or a new querying style. Thus, making the system unreliable for noisy input or schema changes, and their error-handling mechanisms are still under development. Even after the cost of deploying and maintaining of LLM models is expensive.

In the coming time, improvements in semantic understanding and context awareness are expected to enhance the system's performance. Novel approaches such as retrieval-augmented generation, multi-agent pipelines, and human-in-the-loop feedback can improve accuracy, robustness, and adaptability. As databases continue to integrate AI features like embeddings and vector search, Text-to-SQL systems are likely to become more practical, reliable, and accessible to industries worldwide (Liu *et al.* 2025; Shi *et al.* 2025; Zhang and Zhang 2025).

CONCLUSION

The paper presented an in-depth review of the evolution of Large Language Model (LLM)–based Text-to-SQL systems. A systematically organized existing approaches into a clear and structured taxonomy, capturing the key design across different generations of models. In addition, it has been traced that the evolution of benchmark datasets from 2017 to 2024, highlighting how increasing complexity has shaped model development. The study provides a comparative analysis of evaluation metrics and benchmark results, extracting critical insights across multiple datasets and methodologies. Finally, we tried to summarize the common system pipelines, widely adopted tools and datasets, major challenges, and promising future research directions to offer a comprehensive reference for researchers and practitioners.

Ethical standard

The authors have no relevant financial or non-financial interests to disclose.

Availability of data and material

The data that support the findings of this study are available from the corresponding author upon reasonable request.

Conflicts of interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Declaration of generative AI and AI-assisted technologies in the writing process

The authors declare that generative artificial intelligence (AI) tools were used during the preparation of this manuscript. Specifically, AI assistance was utilized for language editing, text refinement, and formatting purposes. The authors take full responsibility for the content and have carefully reviewed and verified all AI-assisted outputs.

LITERATURE CITED

Corradini, F., M. Leonesi, and M. Piangerelli, 2025 State of the Art and Future Directions of Small Language Models: A Systematic Review. Big Data and Cognitive Computing **9**: 189.

Diallo, I., L. Yao, X. Du, and H. Wang, 2023 Harnessing Large Language Models for Business Analytics: Text-to-SQL for Enterprise Data. arXiv preprint .

Guo, J., Z. Zhang, X. Dong, X. Sun, and Q. Zhang, 2019 Towards Complex Text-to-SQL in Cross-Domain Databases with Intermediate Representation. arXiv preprint .

Hong, Z., Z. Yuan, Q. Zhang, H. Chen, J. Dong, *et al.*, 2025 Next-Generation Database Interfaces: A Survey of LLM-Based Text-to-SQL .

Lin, X. V., C. Li, M. Yasunaga, I. Moreno, L. He, *et al.*, 2022 PICARD: Executing SQL Using Constrained Decoding with Large Language Models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Liu, X., S. Shen, B. Li, P. Ma, R. Jiang, *et al.*, 2025 A Survey of Text-to-SQL in the Era of LLMs: Where Are We, and Where Are We Going? arXiv preprint .

Mohammadjafari, A., A. S. Maida, and R. Gottumukkala, 2025 From Natural Language to SQL: Review of LLM-Based Text-to-SQL Systems .

Pourreza, M. and D. Rafiei, 2023 ValueNet: A Neural Text-to-SQL Architecture Incorporating Values. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Rao, J., Z. Hu, W. Zhang, Y. Zhao, and W. Chen, 2023 Benchmarking LLMs for Text-to-SQL: Are We There Yet? arXiv preprint .

Scholak, T., X. V. Li, and D. Bahdanau, 2021 RAT-SQL: Relation-Aware Schema Encoding for Text-to-SQL Parsers. Transactions of the Association for Computational Linguistics **9**: 351–367.

Shi, L., Z. Tang, N. Zhang, X. Zhang, and Z. Yang, 2025 A Survey on Employing Large Language Models for Text-to-SQL Tasks. arXiv preprint .

Wang, B., R. Shin, X. Liu, O. Polozov, M. Richardson, *et al.*, 2020 RAT-SQL + BERT: Enhancing Relation-Aware Encoding for Text-to-SQL Parsing. arXiv preprint .

Yu, T., R. Zhang, H. Er, S. Li, E. Xue, *et al.*, 2019 CoSQL: A Conversational Text-to-SQL Challenge Towards Cross-Domain Natural Language Interfaces to Databases. In *Proceedings of EMNLP-IJCNLP*, pp. 1962–1979.

Yu, T., R. Zhang, K. Yang, M. Yasunaga, S. Li, *et al.*, 2018 Spider: A Large-Scale Human-Labeled Dataset for Complex and Cross-Domain Semantic Parsing and Text-to-SQL Task. arXiv preprint .

Zhang, Y. and X. Zhang, 2025 Text2SQL Is Not Enough: Unifying AI and Databases with TAG .

Zhong, V., C. Xiong, and R. Socher, 2017 Seq2SQL: Generating Structured Queries from Natural Language Using Reinforcement Learning. arXiv preprint arXiv:1709.00103 .

How to cite this article: Singh, G., Chhabra, P. and Banerjee, T. A Comprehensive Review of LLM-based Text-to-SQL Systems: Methods, Datasets, and Trends. *Computational Systems and Artificial Intelligence*, 2(1),1-6, 2026.