

SANE – Schema-aware Natural-language Evaluation of Biological Data

Rolf Gattung *

Martin Krüger¹

Markus Reischl¹

Abstract

High-throughput microscopy generates large, structured datasets capturing cellular responses to pharmacological perturbations, but accessing these datasets typically requires SQL expertise. Large language models offer a natural-language alternative, yet their tendency to hallucinate raises concerns about result reliability.

We present **SANE** (Schema-Aware Natural-language Evaluation), a novel paradigm for domain-specific text-to-SQL evaluation: schema-grounded, automatically generated benchmarks tied to real and specific experimental structure. **SANE** makes evaluation more scalable, systematic, and reproducible.

Using **SANE**, we evaluate a few-shot large language model and show that, under constrained schemas with structured prompting and guardrails, accurate query generation is achievable without any model training or fine-tuning. Most failures stem from ambiguous or underspecified inputs and manifest as overly cautious clarification requests or answers to queries that should first be disambiguated, rather than incorrect SQL generation. These results indicate that few-shot large language models can provide reliable database access in well-defined domains when combined with schema-aware prompting.

1 Introduction

High-throughput drug screening using automated microscopy enables systematic analysis of cellular responses across multiple cell lines, compounds, and concentration levels. These experiments generate structured datasets

containing both raw measurements and derived analytical values such as EC_{50} , drug sensitivity scores, and morphological descriptors.

Platforms such as *Cell-Profiler* [4] provide infrastructure to process, store, and visualize such datasets. However, effective data access remains a significant challenge, as the available views in such tools are limited and querying the databases requires SQL expertise and familiarity with complex schemas.

Large language models (LLMs) offer a promising solution by enabling natural-language interaction with structured data. Text-to-SQL research has evolved from early neural parsers such as Seq2SQL [21] and SQLNet [17] to schema-aware and graph-based models like RAT-SQL [16], PICARD [13], and LGESQL [3]. Large pre-trained models (e.g., T5 [12]) have further advanced natural language interfaces to structured data. Recent works on robust LLM-based frameworks [14] and prompting strategies such as DIN-SQL [11] and DAIL-SQL [6] demonstrate improved zero- and few-shot performance through decomposition and iterative refinement, although models can still hallucinate and make mistakes [8, 1].

Benchmarks such as WikiSQL [21], Spider [20], BIRD [10], and conversational extensions like CoSQL [18] and SParC [19] evaluate cross-domain generalization, multi-table reasoning, and multi-turn interactions. However, these datasets primarily focus on human-authored queries and general schemas, leaving domain-specific biomedical databases with fixed experimental structures largely untested.

In contrast, we focus on automatically generating schema-grounded evaluation queries directly from the underlying database and experimental design, enabling scalable and reproducible benchmarking tailored to domain-specific biological dataset. In this work, we evaluate a

*Institute for Automation and Applied Informatics (IAI), Karlsruhe Institute of Technology (KIT), Hermann-von-Helmholtz-Platz 1, 76344 Eggenstein-Leopoldshafen, Germany. Email: rolf.gattung@kit.edu

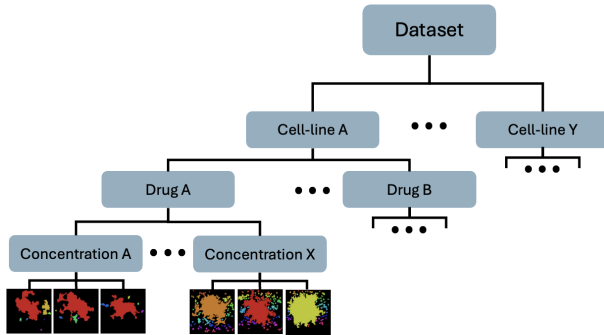


Figure 1: Hierarchical dataset structure of our high-throughput experiments. Each dataset contains multiple cell lines treated with the same drugs at identical concentration ranges. Multiple image replicates are acquired per experimental condition.

custom LLM research assistant that queries a PostgreSQL database using a 4-bit quantized LLaMA 3.1 model without any training or fine-tuning. The LLM and database are integrated into a web application planned for public release. Here, we introduce **SANE**, a custom framework that systematically derives realistic, schema-grounded test queries from high-throughput drug response databases for rigorous evaluation. Our key contributions are:

- A scalable framework for domain-specific benchmarking of text-to-SQL systems on high-throughput biological data
- A large-scale evaluation with 572 non-trivial, automatically generated queries derived from real experiments
- A detailed analysis of robustness and failure modes, identifying ambiguity in natural language as the primary source of errors

2 System Context

In high-throughput microscopy, e.g. drug screenings, the datasets typically follow a hierarchical experimental design as in Figure 1. Experiments include one or more cell lines, each cell-line is tested against multiple drugs at various concentrations. Multiple images of replicates are ac-

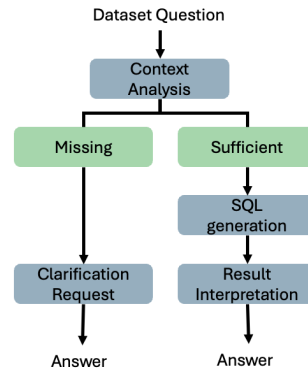


Figure 2: Three stage LLM processing pipeline. First, the model determines whether sufficient context is available to answer the query. If the context is *sufficient*, a SQL query is generated and executed. The result is then interpreted into a natural language response, while *missing* context triggers a clarification request.

quired per condition, reflecting standard drug screening protocols. From these images our webapp extracts morphological features and computes derived statistics (EC₅₀, dose-response curves, sensitivity scores), storing them in a relational database.

To query this data we can use a text-to-SQL LLM. In this work, we evaluate the quantized Llama 3.1 [7] model running on a single NVIDIA A6000 GPU with 48 GB using virtual large language model (vLLM) [9]. Crucially, the model operates in few-shot mode, no training or fine-tuning is performed. Previous works showed the capabilities of few-shot LLMs [5, 2, 15]. The LLM interprets natural-language queries about dose-response experiments, generates SQL statements, and retrieves structured data from the underlying relational database. For example, for the question: "How many datasets do I own?" the model would generate a query like "SELECT COUNT(*) FROM datasets d WHERE d.owner_id = user_id" execute it and retrieve the number of datasets of that user.

The LLM translates natural-language queries into SQL through three stages, visualized in Figure 2. First the LLM is tasked to determine whether sufficient context is provided to answer the query, outputting a binary label (*missing/sufficient*). If the label is *missing*, a clarifica-

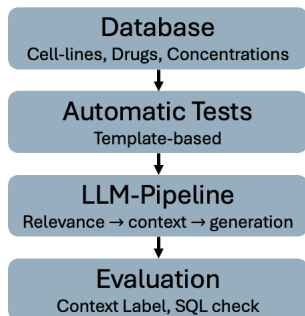


Figure 3: **SANE** benchmark generation. The database is systematically queried to extract schema structure and experimental content. Test queries are generated with corresponding ground truth SQL. The LLM is evaluated by comparing predicted context classifications (*sufficient* vs. *missing*) and generated SQL execution results against ground truth.

tion request is returned. Otherwise we generate a SQL statement with the LLM using schema-aware prompting. The returned database result is then interpreted by another LLM pass, turned into a natural language answer, bullet-point listing or table.

For SQL generation, schema-aware prompting injects database structure, domain terminology, filtering rules (e.g., excluding border artifacts), and dataset-specific context and examples to guide the generation. The LLM and database are treated as fixed, our focus is systematically evaluating accuracy, robustness, and limitations under realistic scenarios.

3 SANE

SANE is a framework that systematically generates natural language queries and corresponding ground truth SQL statements directly from database contents and experimental structure (Figure 3). The framework queries a pre-existing database from real experiments to extract schema information and representative data instances.

Based on this information, SANE constructs queries spanning multiple complexity levels, including simple retrieval, complex analytical queries, and multi-step interactions. In addition, it introduces controlled perturbations

such as typographical errors, abbreviations, and missing context to simulate realistic user interaction scenarios.

The generated queries per category are filled with randomly sampled real data, such as drug names, and are easily extendable. The respective SQL generation is non-trivial, requiring correct handling of hierarchical relationships, domain-specific terminology, and implicit constraints such as dataset scoping.

We define 69 fine-grained query categories and corresponding SQL statements, fully available in the supplementary material together with the benchmark and LLM code. For clarity, we group the questions into six top-level categories:

Simple queries involve straightforward retrieval, e.g., "What is the area of HUH7 cells for Alvocidib at $10\mu M$?".

Complex queries require joins, aggregations, rankings, or listings, e.g., What are the top 5 drugs by DSS score for MCF7 on the circularity feature?

Errors include queries with abbreviations, typographical errors, or inconsistencies, e.g., what is the mean area for HUH7 treated with 9-IN-41 at 0.02 μM ?, where the correct drug name is 9-ING-41.

Contextless queries are underspecified and return overly large result sets. For example, "What is the EC50 for MCF7 on the area feature?" retrieves values for all drugs and should be further refined first.

Multi-step queries represent conversational scenarios with incremental refinement of information, e.g. specifying the drug in a followup message in the previous scenario.

Schema queries request structured information across tables, e.g., "How many concentration rows correspond to drug 9-ING-41?"

For each question posed to the LLM, we compare the predicted context label (*missing* or *sufficient*) with ground truth. For *missing* queries, correctness is determined by label agreement. For *sufficient* queries, correctness additionally requires that the generated SQL execution results match the reference results (result set equivalence).

This evaluation jointly assesses the model’s ability to interpret query intent and to generate executable SQL consistent with the database schema.

Table 1: Evaluation results grouped by query type.

Category	Total	Correct	False	Accuracy (%)
Simple	159	159	0	100.0
Complex	316	311	5	98.4
Errors	36	31	5	86.1
Contextless	22	18	4	81.8
Schema	32	31	1	96.9
Multi-step	7	6	1	85.7
Total	572	556	16	97.2

4 Results & Discussion

The evaluation comprises **572** automatically generated test cases spanning various database schemas and statistical attributes. Table 1 summarizes few-shot performance across the categories.

The few-shot system achieves **97.2%** overall accuracy, demonstrating reliable performance across retrieval, complex analysis, and schema-aware tasks without any training or fine-tuning. *Simple*, *Complex* and *Schema* related questions achieve a high accuracy of 100%, 98.4% and 96.9%. In categories *Contextless*, *Errors* and *Multi-step* the LLM performs slightly worse, with 81.8%, 86.1% and 85.7%. This high accuracy is achieved through domain-specific and example-driven prompting and by limiting the scope of LLM and the task. Without examples (zero-shot), solely relying on simple scheme injection, results drop to 29.9% overall accuracy.

Among the 16 few-shot failures listed in Table 1, 10 involve incorrect *missing* context label prediction, typically from unknown synonyms or misinterpreted under-specified queries. This is likely driven by the model’s primary focus on accurate SQL generation for well-defined queries. When errors or missing context are introduced, the resulting uncertainty leads to inaccurate query generation or misclassification of the context label. In 5 cases, the generated SQL is slightly incorrect, e.g. omitting drug identifiers in listings or answering overly broad questions instead of asking for clarification. One question is answered with the generic dismiss answer of the webapp because it is wrongly interpreted as not data-related (Q: What is the average EC50 for Trametinib for the area feature? A: I’m here to help! Feel free to ask about your data or the application.). However, this analysis suggests that

the effective error rate in practical usage for SQL generation is lower than the benchmark suggests, since most failure cases occur due to wrong context label prediction or omitting of names, not because of wrong numerical values.

The results demonstrate that our few-shot LLM achieves high reliability in querying complex biological databases when combined with schema-aware prompting and domain-specific constraints, validating its usage as research assistant.

Importantly, performance strongly depends on structured prompting and schema knowledge. Without domain-specific terminology and examples, the LLM cannot reliably produce valid queries. This emphasizes the critical role of prompt engineering in deploying LLMs for specialized database interfaces without requiring model training.

Failure analysis indicates future improvements should prioritize interactive query refinement and disambiguation rather than model fine-tuning as the model seems to struggle with missing context and unclear questions rather than accurate text-to-SQL generation as shown in the results. Most errors stem from ambiguous or erroneous user input, suggesting that conversational clarification mechanisms would substantially improve practical usability. Additionally, synonym expansion and domain-specific entity recognition could reduce context classification errors.

This approach significantly lowers barriers to accessing high-throughput biological data through prompting alone. By eliminating both SQL expertise requirements and the need for model training, researchers can directly interrogate experimental results, accelerating hypothesis generation and data-driven discovery.

5 Conclusion

We presented a systematic evaluation of a custom domain-specific LLM interface for natural-language querying of biological databases. Our framework **SANE** automatically generates realistic, schema-grounded test-cases from experimental designs, enabling comprehensive and reproducible assessment.

Achieving 97.2% accuracy on 572 queries through prompting alone—without any training or fine-

tuning—the system demonstrates strong performance. This highlights its potential for real-world use in constrained, domain-specific environments, particularly for reliable extraction of numerical values. This is especially useful for large databases that otherwise require custom scripts or software and do not offer an easy, interactive access to the data, as in the high-throughput cell-image analysis.

Future work could focus on interactive disambiguation mechanisms for the LLM for ambiguous questions and extending the **SANE** framework to broader biomedical domains, including clinical databases and multi-omics repositories.

References

- [1] Aisha Alansari et al. *Large Language Models Hallucination: A Comprehensive Survey*. 2026. arXiv: 2510.06265.
- [2] Tom B. Brown et al. *Language Models are Few-Shot Learners*. 2020. arXiv: 2005.14165.
- [3] Ruisheng Cao et al. *LGESQL: Line Graph Enhanced Text-to-SQL Model with Mixed Local and Non-Local Relations*. 2021. arXiv: 2106.01093.
- [4] Anne E. Carpenter et al. “CellProfiler: image analysis software for identifying and quantifying cell phenotypes”. In: *Genome Biology* 7.10 (2006), R100. DOI: 10.1186/gb-2006-7-10-r100.
- [5] Mia Cate. “Understanding Zero-Shot and Few-Shot Learning in LLMs”. In: (June 2023).
- [6] Dawei Gao et al. *Text-to-SQL Empowered by Large Language Models: A Benchmark Evaluation*. 2023. arXiv: 2308.15363.
- [7] Aaron Grattafiori et al. *The Llama 3 Herd of Models*. 2024. arXiv: 2407.21783.
- [8] Adam Tauman Kalai et al. *Why Language Models Hallucinate*. 2025. arXiv: 2509.04664.
- [9] Woosuk Kwon et al. *Efficient Memory Management for Large Language Model Serving with PagedAttention*. 2023. arXiv: 2309.06180.
- [10] Jinyang Li et al. *Can LLM Already Serve as A Database Interface? A BIG Bench for Large-Scale Database Grounded Text-to-SQLs*. 2023. arXiv: 2305.03111.
- [11] Mohammadreza Pourreza et al. *DIN-SQL: Decomposed In-Context Learning of Text-to-SQL with Self-Correction*. 2023. arXiv: 2304.11015.
- [12] Colin Raffel et al. *Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer*. 2023. arXiv: 1910.10683.
- [13] Torsten Scholak et al. *PICARD: Parsing Incrementally for Constrained Auto-Regressive Decoding from Language Models*. 2021. arXiv: 2109.05093.
- [14] Xiaodong Su et al. “A robust natural language text-to-SQL generation framework with dynamic strategies based on LLMs”. In: *Scientific Reports* 16 (2026), p. 7892. DOI: 10.1038/s41598-026-39128-9.
- [15] Hisaya Sudo et al. “Evaluation of Few-Shot AI-Generated Feedback on Case Reports in Physical Therapy Education: Mixed Methods Study”. In: *JMIR Med Educ* 11 (Dec. 2025), e85614. ISSN: 2369-3762. DOI: 10.2196/85614.
- [16] Bailin Wang et al. *RAT-SQL: Relation-Aware Schema Encoding and Linking for Text-to-SQL Parsers*. 2021. arXiv: 1911.04942.
- [17] Xiaojun Xu et al. *SQLNet: Generating Structured Queries From Natural Language Without Reinforcement Learning*. 2017. arXiv: 1711.04436.
- [18] Tao Yu et al. *CoSQL: A Conversational Text-to-SQL Challenge Towards Cross-Domain Natural Language Interfaces to Databases*. 2019. arXiv: 1909.05378.
- [19] Tao Yu et al. *SParC: Cross-Domain Semantic Parsing in Context*. 2019. arXiv: 1906.02285.
- [20] Tao Yu et al. *Spider: A Large-Scale Human-Labeled Dataset for Complex and Cross-Domain Semantic Parsing and Text-to-SQL Task*. 2019. arXiv: 1809.08887.
- [21] Victor Zhong et al. *Seq2SQL: Generating Structured Queries from Natural Language using Reinforcement Learning*. 2017. arXiv: 1709.00103.