ABACUS-SQL: A Text-to-SQL System Empowering Cross-Domain and Open-Domain Database Retrieval

Keyan Xu, Dingzirui Wang, Xuanliang Zhang, Qingfu Zhu, Wanxiang Che*
Harbin Institute of Technology
{kyxu, dzrwang, xuanliangzhang, qfzhu, car}@ir.hit.edu.cn

Abstract

The existing text-to-SQL systems have made significant progress in SQL query generation, but they still face numerous challenges. Existing systems often lack retrieval capabilities for open-domain databases, requiring users to manually filter relevant databases. Additionally, their cross-domain transferability is limited, making it challenging to accommodate diverse query requirements. To address these issues, we propose ABACUS-SQL. ABACUS-SQL utilizes database retrieval technology to accurately locate the required databases in an open-domain database environment. It also enhances the system cross-domain transfer ability through data augmentation methods. Moreover, ABACUS-SQL employs Pre-SQL and Self-debug methods, thereby enhancing the accuracy of SQL queries. Experimental results demonstrate that ABACUS-SQL performs excellently in multi-turn text-to-SQL tasks, effectively validating the approach's effectiveness. ABACUS-SQL is publicly accessible at https://huozi.8wss.com/abacus-sql/. [1]

1 Introduction

Text-to-SQL (Yu et al., 2019b) is a natural language processing (NLP) technique designed to automatically convert natural language queries into SQL statements, thereby lowering the barrier to data querying. This technique has been widely applied in areas such as business analytics and customer support (Liu et al., 2024; Hong et al., 2024; Katsogiannis-Meimarakis and Koutrika, 2023). However, existing text-to-SQL technologies remain challenging to use due to complex database structures, ambiguous natural language understanding, and diverse user query habits (Xue et al., 2024). To improve usability, it is essential to develop a powerful, intuitive and user-friendly text-to-SQL system capable of accurately interpreting users' diverse natural language queries and generating efficient and precise SQL statements.

Previous text-to-SQL systems (Zeng et al., 2020, 2023) have demonstrated the potential of natural language interaction with databases, with notable innovations from systems such as DB-GPT (Xue et al., 2024) and PHOTON (Zeng et al., 2020). DB-GPT possesses powerful SQL generation capabilities, while its novel Retrieval-Augmented Generation (RAG) knowledge system and adaptive learning mechanism further enhance query efficiency. PHOTON enhances the system ability to handle ambiguous and complex user inputs by integrating deep learning with a human-in-the-loop correction mechanism, thereby improving its cross-domain adaptability and robustness.

Although existing text-to-SQL systems have made significant progress in SQL query generation, they still face several limitations (Table 1). Current systems lack efficient **database retrieval capability** and struggle to automatically locate the required database in open-domain database environments, forcing users to manually filter databases, which reduces the system's generality and efficiency. Additionally, existing systems exhibit limited **cross-domain transferability**, as most require pretraining for specific domains. This constraint restricts their applicability across different domains, making it increasingly difficult to meet the query needs of specialized databases.

To address the above limitations of existing text-to-SQL systems, we develop ABACUS-SQL, focusing on enhancing multi-database retrieval performance and cross-domain transferability while introducing several innovative methods to optimize SQL generation. First, ABACUS-SQL supports retrieval in open-domain databases by leveraging beam search and query rewriting to accurately locate the required database. Second, ABACUS-SQL exhibits robust cross-domain transferability

*Corresponding author.
[1] https://github.com/starryneigh/Abacus-SQL

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations), pages 118–128

System	Multi-Turn?	Database Retrieval?	Cross-Domain?
DBGPT (Xue et al., 2024)	×	×	×
PHOTON (Zeng et al., 2020)	×	×	×
SQLChat[1]	×	×	×
Vanna[2]	✓	×	×
WrenAI[3]	✓	×	×
ABACUS-SQL	✓	✓	✓

Table 1: Comparison of ABACUS-SQL with previous systems.

by utilizing data augmentation methods to synthesize demonstrations based on domain-specific databases, enabling the system to quickly adapt to diverse domain requirements. Moreover, ABACUS-SQL integrates pre-SQL and self-debug methods, ensuring the generation of high-quality SQL even in complex query scenarios, thereby further enhancing the system's practicality and reliability.

Overall, we develop ABACUS-SQL, a robust text-to-SQL system designed for cross-domain and open-domain database environments. Our main contributions are as follows:

- **Database retrieval capability:** To address the retrieval challenges in multi-database environments, ABACUS-SQL employs open-domain database retrieval method, enabling efficient retrieval of relevant databases.
- **Cross-Domain Transferability:** To enhance cross-domain transferability, ABACUS-SQL utilizes data augmentation methods to synthesize examples from domain-specific databases, significantly improving cross-domain adaptability.
- **System Optimization:** To improve the quality of SQL query generation, ABACUS-SQL incorporates multiple innovative methods, significantly enhancing the accuracy of results.

2 Related Work

2.1 Multi-turn Text-to-SQL

Early multi-turn text-to-SQL research primarily relied on deep neural network models, improving SQL generation accuracy through specialized architectures. For example, Wang et al. (2020) proposed leveraging previous SQL queries to enhance parsing accuracy and contextual understanding, while RASAT (Qi et al., 2022) introduced a relation-aware self-attention mechanism within the Transformer structure to improve dialogue context integration. However, such models face significant challenges including high data annotation costs and complex context management (Gao et al., 2023).

With the advancement of large language models (LLMs), LLM-based methods have gradually become the mainstream, achieving high performance without additional fine-tuning, thereby reducing dependence on large datasets and computational resources (Hong et al., 2024). ACT-SQL (Zhang et al., 2023) utilizes Chain-of-Thought reasoning to decompose multi-turn conversations into single-turn queries, handling dependencies through query rewriting and context completion. CoE-SQL (Zhang et al., 2024a) further optimizes this process by adopting an edit-based strategy that incrementally updates SQL queries, avoiding error accumulation caused by query rewriting, thereby improving stability and accuracy. Overall, the integration of LLMs has made multi-turn text-to-SQL more efficient and versatile, reducing resource demands while enhancing the coherence and precision of SQL generation (Zhang et al., 2024a).

2.2 Text-to-SQL System

In recent years, text-to-SQL technology has made significant advancements, leading to the emergence of various open-source tools that simplify user-database interactions and enable non-expert users to easily access the data they need. DB-GPT (Xue et al., 2024) is a framework that integrates LLMs with database interaction technologies. It supports natural language queries, efficient SQL generation, multilingual support, and incorporates privacy protection and multi-agent collaboration strategies, offering new perspectives for text-to-

[1] https://github.com/sqlchat/sqlchat
[2] https://github.com/vanna-ai/vanna
[3] https://github.com/Canner/WrenAI

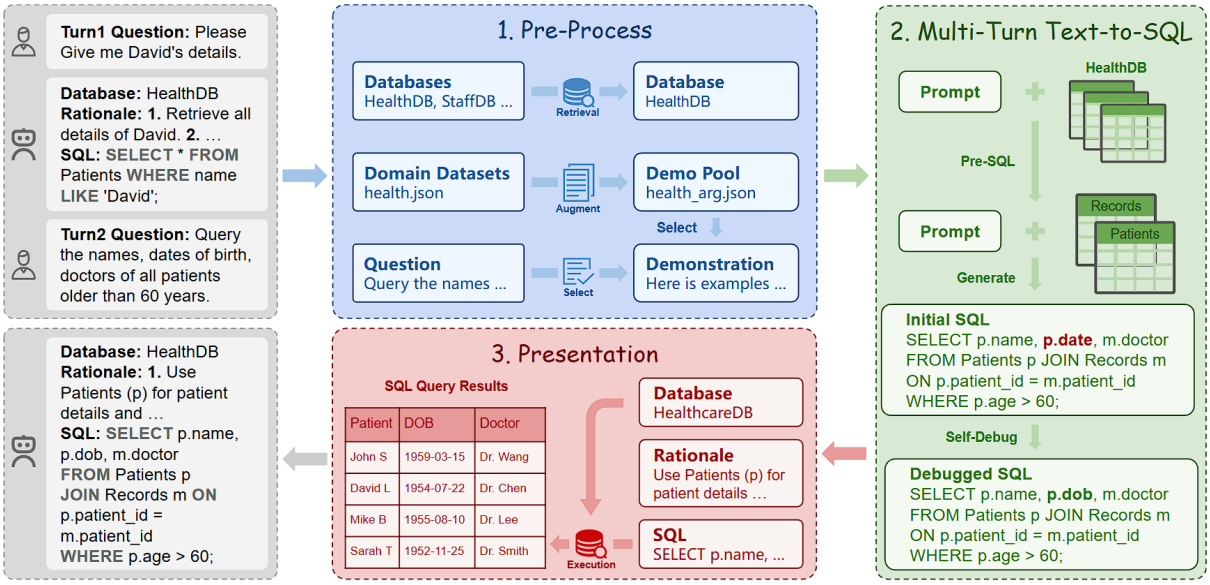

Figure 1: The illustration of ABACUS-SQL, which consists of three steps: *1. Preprocessing*: Retrieves open-domain databases and enhances cross-domain transferability with data augmentation. *2. Multi-turn Text-to-SQL*: Improves the accuracy of multi-turn SQL queries using Pre-SQL and Self-debug methods. *3. Presentation*: Shows the inference process, SQL queries, and real-time execution results to users.

SQL system development. PHOTON (Zeng et al., 2020) is a cross-domain natural language interface database system that effectively enhances the handling of complex and ambiguous queries through deep learning and a human-in-the-loop correction mechanism. SQLChat[1] adopts a conversational interaction model, enabling users to execute database operations through natural language. WrenAI[2] functions as an SQL AI agent, supporting multi-database environments and integrating semantic understanding to improve query efficiency. These tools have significantly driven the development of text-to-SQL, catering to diverse user needs and expanding the accessibility of database querying.

However, existing systems often lack retrieval functionality for open-domain databases, increasing user operation complexity and time cost. They also struggle with cross-domain transferability, making it hard to adapt to different data structures and query needs. Therefore, enhancing the system's domain transferability and adaptability in multi-database environments is a key challenge for text-to-SQL systems.

3 System Workflow

In this section, we introduce the workflow of our system, which is designed to address the limitations of previous systems, including insufficient retrieval capabilities, limited transferability, and suboptimal SQL generation. The workflow, as illustrated in the Figure 1, consists of three core phases: preprocessing, multi-turn text-to-SQL, and presentation. To overcome the shortcomings of existing systems, we implemented several optimizations. First, we employ the Murre method (Section 3.1.1) for automatic retrieval to extract databases relevant to the given query. Second, we utilize the fused method (Section 3.1.2) for data augmentation, enhancing the system's cross-domain transferability. Finally, in the SQL generation phase, we introduce Pre-SQL (Section 3.2.2) and Self-debug (Section 3.2.3) to improve the accuracy of SQL generation.

3.1 Preprocess

During the initial data preprocessing stage, we prepare for subsequent SQL generation through three key steps: open-Domain database retrieval, augmentation and selection of demonstration, and extraction of database schema information.

3.1.1 Open-Domain Database Retrieval

We first automatically identify and select the most relevant database based on the user's query and the uploaded databases. This process consists of two steps: database matching, which aligns the user query with database schemas and metadata to determine databases likely containing the target information; and database prioritization, which evaluates and ranks multiple relevant databases to

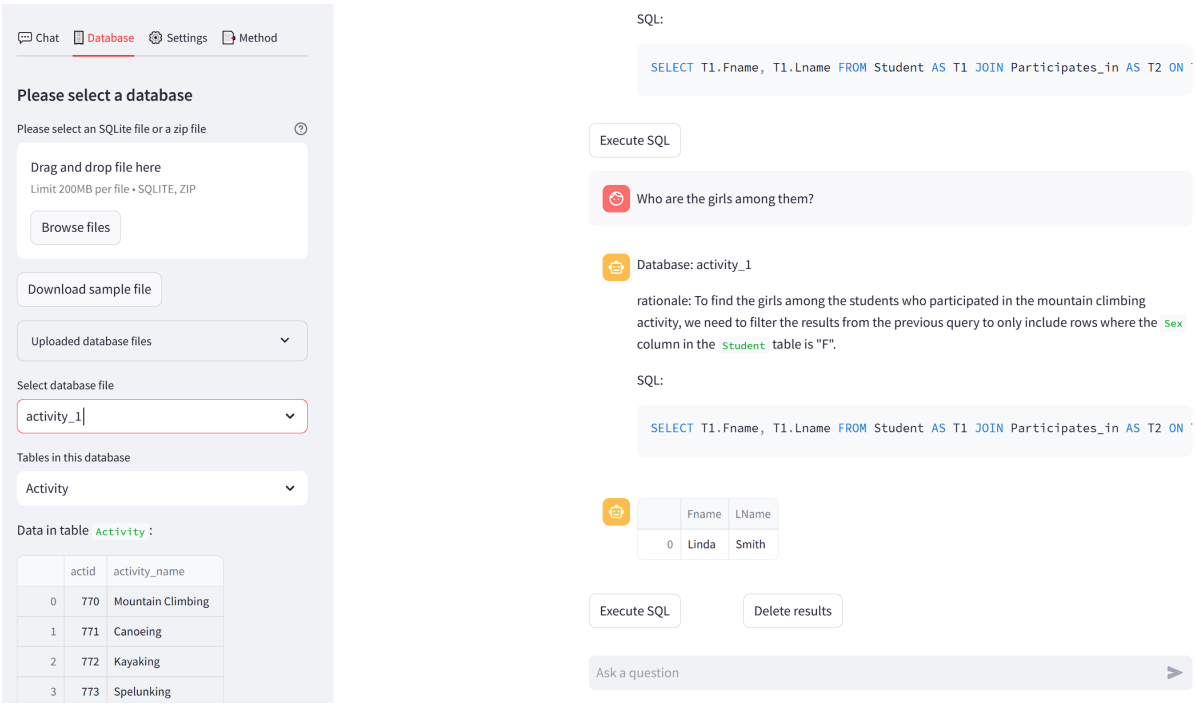

Figure 2: The interface of ABACUS-SQL: The sidebar provides various functions, such as uploading and viewing user databases, as well as switching between sessions. The main area facilitates interaction with ABACUS-SQL, allowing users to generate SQL queries and execute query results.

select the most suitable one. Specifically, we employ the Murre method from (Zhang et al., 2024b), iteratively performing database beam scarch and query-related field elimination. Detailed implementation can be found in Appendix A.

3.1.2 Demonstration Selection

We select demonstrations (Dong et al., 2024) from domain-specific datasets to help the model better align with domain characteristics for effective domain adaptation while also providing reference examples. We first employ data augmentation to expand either the default domain dataset or user-provided domain dataset, enhancing data diversity and adaptability to improve the model's cross-domain transferability. Here, we adopt the Fused method from (Wang et al., 2024b), leveraging a large language model (LLM) to iteratively update the demonstration pool (detailed implementation is provided in Appendix B). Subsequently, we perform demonstration selection using the BM25 (Robertson and Zaragoza, 2009) algorithm, which incorporates user query requirements, database schema, and dialogue context to retrieve demonstration from the predefined demonstration pool, providing valuable references for SQL generation.

3.1.3 Schema Extraction

Here, we systematically extract table schema from the previously selected database and precisely align the database structure with the user query. First, we retrieve table names, column names, data types, and their underlying relationships from the database and organize them into a format that is easily interpretable by the LLMs. Then, by aligning the fields in the user query with the database content, we ensure that the model accurately identifies the query intent, enabling the generated SQL to correctly map to the relevant tables and fields.

3.2 Multi-Turn Text-to-SQL

3.2.1 Prompt

This section aims to utilize the output from preprocessing to construct high-quality prompts (detailed in Appendix C), guiding the model in accurately generating SQL queries within multi-turn dialogue scenarios. Specifically, it includes: system prompts, which define the model's role, task, and output specifications; few-shot demonstrations, providing highly relevant references to help the model better understand query requirements; schema, which outline the database structure and relationships; and multi-turn dialogue, which leverage historical context to capture semantic associations and intent

Dataset	Method	7B		32B	
		QEX	IEX	QEX	IEX
Chase-C	Qwen2.5-Coder	40.4	11.1	46.5	18.0
	+ ABACUS-SQL	**45.5**	**15.0**	**53.5**	**23.1**
SParC	Qwen2.5-Coder	67.3	45.7	69.0	46.9
	+ ABACUS-SQL	**68.4**	**46.9**	**69.6**	**47.4**
CoSQL	Qwen2.5-Coder	69.4	40.3	72.0	41.3
	+ ABACUS-SQL	**70.6**	**42.3**	**73.1**	**42.7**

Table 2: The main experimental results with and without ABACUS-SQL. The best result under each setting is marked in **bold**.

shifts, thereby improving query accuracy.

3.2.2 Pre-SQL

Considering that excessive table information in multi-turn dialogues may interfere with the model's understanding of user intent, we first focus on filtering out table information that is irrelevant to the user's query. At this stage, we use a prompt as input to guide the large language model in pre-generating an SQL query (Li et al., 2024). Subsequently, we refine the generated SQL query by eliminating unnecessary table information, ensuring that only relevant tables and fields are extracted. This process not only guarantees a high degree of alignment between the SQL query and user intent but also effectively reduces redundancy, thereby enhancing query accuracy and execution efficiency.

3.2.3 Self-Debug

Self-debug (Wang et al., 2024a) refers to the process of detecting errors in the generated SQL query and then reintroducing the error information, along with table schema details and the user query, back into the model to facilitate error correction. This approach is inspired by the methodology presented in (Chen et al., 2023). During this process, the model leverages syntax error prompts, database schema information, and the original user query to generate a revised SQL query. By iteratively debugging itself, the model not only identifies and rectifies syntax errors but also improves its understanding of the query, thereby optimizing the SQL generation process.

3.3 Presentation

To enhance user experience, ABACUS-SQL provides a transparent interaction mechanism, allowing users to clearly understand the SQL generation process and obtain real-time query results.

Inference Process Visualization The system provides a step-by-step explanation of the SQL generation and refinement process to help users better understand the query.

Real-time execution results SQL query results are displayed in tabular format, allowing users to quickly verify the accuracy of the generated SQL and enhancing the interactive experience.

4 System Design

This section presents the web design of Abacus-SQL to help users better understand the system's features and how to interact with it.

4.1 Frontend

The front-end of ABACUS-SQL (Figure 2) is built using Streamlit (Streamlit, 2024), designed to provide a simple and intuitive user interface that enhances the overall user experience. As a comprehensive text-to-SQL system, ABACUS-SQL incorporates a range of core functionalities, including:

User Authentication Integrates a lightweight login system supporting account registration and encrypted password storage, along with Huozi (Huozi-Team, 2024) account login compatibility, ensuring privacy protection and seamless access.

Conversation Management Supports multi-session management, allowing users to store query history and dialogue context, thereby enhancing interaction continuity and traceability.

Database Content Visualization Provides an intuitive interface that clearly displays database tables, fields, and data, allowing users to easily browse and verify SQL queries.

Streaming output Supports real-time streaming of the SQL generation process, reducing wait time and allowing users to access partial results earlier, thereby enhancing the interactive experience.

4.2 Backend

The backend of ABACUS-SQL is built on FastAPI, providing efficient and flexible service capabilities while optimizing streaming output support. The backend utilizes Qwen2.5-Coder-7B (Hui et al., 2024) for SQL generation. Although it has not undergone fine-tuning, its strong generative capabilities are sufficient for general text-to-SQL tasks. Additionally, ABACUS-SQL supports remote LLM API services (such as GPT-4o (OpenAI et al., 2024) and DeepSeek-R1 (DeepSeek-AI et al., 2025)), allowing users to securely integrate these models via API keys to generate more precise SQL queries.

5 Experiment

5.1 Experiment Setup

Dataset The ABACUS-SQL multi-turn text-to-SQL evaluation benchmark is based on three datasets: Chase-C (Guo et al., 2021), SParC (Yu et al., 2019c), and CoSQL (Yu et al., 2019a). Chase is currently the largest cross-domain, context-dependent Chinese Text-to-SQL dataset. It consists of $5,459$ conversational turns ($17,940$ questions) spanning over 280 databases. Unlike other datasets, Chase-C features manually crafted questions based on database schemas from scratch, making it more realistic for practical applications. SParC is a cross-domain, multi-turn Text-to-SQL English dataset. It comprises approximately $12,000+$ annotated natural language question-to-SQL pairs. These questions are derived from 200 complex databases covering 138 distinct domains. CoSQL is another cross-domain, multi-turn Text-to-SQL English dataset. It contains over 3,000 conversational turns with 10,000+ annotated SQL queries. Each dialogue in CoSQL is specifically designed to simulate real-world database interaction scenarios.

Metric To evaluate the performance of ABACUS-SQL, we use two metrics: Question Execution Accuracy (QEX) and Interaction Execution Accuracy (IEX) (Zhang et al., 2024a). QEX measures the execution accuracy of single-turn SQL queries, similar to EX, but focuses on the query result for individual questions. IEX assesses the execution correctness of all SQL queries across multiple interaction turns, ensuring that the system consistently generates accurate SQL throughout the entire conversation. Together, these metrics provide a comprehensive evaluation of the system's text-to-SQL capability in multi-turn dialogue scenarios.

Model We used Qwen2.5-Coder 7B and 32B to evaluate the performance of ABACUS-SQL on multi-turn text-to-SQL tasks. Qwen2.5-Coder (Hui et al., 2024) is a code generation model based on Qwen2.5, equipped with powerful code understanding and generation capabilities. It is suitable for tasks across various programming languages, including SQL query generation. We set the inference to 3-shot with a temperature of 0.

5.2 Main Result

As shown in Table 2, ABACUS-SQL demonstrates improvements across all datasets compared to the baseline, with significant enhancement observed in the Chase-C dataset, highlighting its strong competitive edge in this domain. We also conducted ablation experiments on the pre-SQL and self-debug methods, finding that both approaches can improve system performance, with particularly more significant effects on Chinese datasets, thereby validating the effectiveness of the methods. (Appendix D). This result underscores ABACUS-SQL's exceptional ability in multi-turn dialogue understanding and SQL generation, indicating its immense potential in applications that combine database querying and natural language processing.

6 Conclusion

We propose ABACUS-SQL, a novel multi-turn dialogue-oriented text-to-SQL system designed to enhance database retrieval, cross-domain transferability, and SQL generation accuracy and efficiency. ABACUS-SQL tackles existing challenges in current systems, such as the inability to efficiently retrieve relevant databases from open-domain database environment and the difficulty in transferring across diverse domains. By integrating the Murre method for efficient database retrieval, the Fused method to improve data generalization, and a combination of Pre-SQL and Self-debug to optimize query parsing, ABACUS-SQL demonstrates exceptional adaptability and stability in handling complex query tasks. These results validate its effectiveness in real-world applications.

Acknowledge

We gratefully acknowledge the support of the National Natural Science Foundation of China (NSFC) via grant 62236004, 62206078 and 62476073.

References

Xinyun Chen, Maxwell Lin, Nathanael Schärli, and Denny Zhou. 2023. Teaching Large Language Models to Self-Debug. *arXiv preprint*.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. 2025. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. *arXiv preprint*.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Jingyuan Ma, Rui Li, Heming Xia, Jingjing Xu, Zhiyong Wu, Tianyu Liu, Baobao Chang, Xu Sun, Lei Li, and Zhifang Sui. 2024. A Survey on In-context Learning. *arXiv preprint*.

Dawei Gao, Haibin Wang, Yaliang Li, Xiuyu Sun, Yichen Qian, Bolin Ding, and Jingren Zhou. 2023. Text-to-SQL Empowered by Large Language Models: A Benchmark Evaluation. *arXiv preprint*. ArXiv:2308.15363 [cs].

Jiaqi Guo, Ziliang Si, Yu Wang, Qian Liu, Ming Fan, Jian-Guang Lou, Zijiang Yang, and Ting Liu. 2021. Chase: A Large-Scale and Pragmatic Chinese Dataset for Cross-Database Context-Dependent Text-to-SQL. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2316–2331, Online. Association for Computational Linguistics.

Zijin Hong, Zheng Yuan, Qinggang Zhang, Hao Chen, Junnan Dong, Feiran Huang, and Xiao Huang. 2024. Next-Generation Database Interfaces: A Survey of LLM-based Text-to-SQL. *arXiv preprint*. ArXiv:2406.08426.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, Kai Dang, Yang Fan, Yichang Zhang, An Yang, Rui Men, Fei Huang, Bo Zheng, Yibo Miao, Shanghaoran Quan, Yunlong Feng, Xingzhang Ren, Xuancheng Ren, Jingren Zhou, and Junyang Lin. 2024. Qwen2.5-Coder Technical Report. *arXiv preprint*.

Huozi-Team. 2024. HIT-SCIR/huozi.

George Katsogiannis-Meimarakis and Georgia Koutrika. 2023. A survey on deep learning approaches for text-to-SQL. *The VLDB Journal*, 32(4):905–936.

Zhishuai Li, Xiang Wang, Jingjing Zhao, Sun Yang, Guoqing Du, Xiaoru Hu, Bin Zhang, Yuxiao Ye, Ziyue Li, Rui Zhao, and Hangyu Mao. 2024. PET-SQL: A Prompt-Enhanced Two-Round Refinement of Text-to-SQL with Cross-consistency. *arXiv preprint*. ArXiv:2403.09732 [cs].

Xinyu Liu, Shuyu Shen, Boyan Li, Peixian Ma, Runzhi Jiang, Yuyu Luo, Yuxin Zhang, Ju Fan, Guoliang Li, and Nan Tang. 2024. A Survey of NL2SQL with Large Language Models: Where are we, and where are we going?

OpenAI, Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, A. J. Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, Aleksander Mądry, Alex Baker-Whitcomb, Alex Beutel, Alex Borzunov, Alex Carney, Alex Chow, Alex

Kirillov, Alex Nichol, Alex Paino, Alex Renzin, Alex Tachard Passos, Alexander Kirillov, Alexi Christakis, Alexis Conneau, Ali Kamali, Allan Jabri, Allison Moyer, Allison Tam, Amadou Crookes, Amin Tootoochian, Amin Tootoonchian, Ananya Kumar, Andrea Vallone, Andrej Karpathy, Andrew Braunstein, Andrew Cann, Andrew Codispoti, Andrew Galu, Andrew Kondrich, Andrew Tulloch, Andrey Mishchenko, Angela Baek, Angela Jiang, Antoine Pelisse, Antonia Woodford, Anuj Gosalia, Arka Dhar, Ashley Pantuliano, Avi Nayak, Avital Oliver, Barret Zoph, Behrooz Ghorbani, Ben Leimberger, Ben Rossen, Ben Sokolowsky, Ben Wang, Benjamin Zweig, Beth Hoover, Blake Samic, Bob McGrew, Bobby Spero, Bogo Giertler, Bowen Cheng, Brad Lightcap, Brandon Walkin, Brendan Quinn, Brian Guarraci, Brian Hsu, Bright Kellogg, Brydon Eastman, Camillo Lugaresi, Carroll Wainwright, Cary Bassin, Cary Hudson, Casey Chu, Chad Nelson, Chak Li, Chan Jun Shern, Channing Conger, Charlotte Barette, Chelsea Voss, Chen Ding, Cheng Lu, Chong Zhang, Chris Beaumont, Chris Hallacy, Chris Koch, Christian Gibson, Christina Kim, Christine Choi, Christine McLeavey, Christopher Hesse, Claudia Fischer, Clemens Winter, Coley Czarnecki, Colin Jarvis, Colin Wei, Constantin Koumouzelis, Dane Sherburn, Daniel Kappler, Daniel Levin, Daniel Levy, David Carr, David Farhi, David Mely, David Robinson, David Sasaki, Denny Jin, Dev Valladares, Dimitris Tsipras, Doug Li, Duc Phong Nguyen, Duncan Findlay, Edede Oiwoh, Edmund Wong, Ehsan Asdar, Elizabeth Proehl, Elizabeth Yang, Eric Antonow, Eric Kramer, Eric Peterson, Eric Sigler, Eric Wallace, Eugene Brevdo, Evan Mays, Farzad Khorasani, Felipe Petroski Such, Filippo Raso, Francis Zhang, Fred von Lohmann, Freddie Sulit, Gabriel Goh, Gene Oden, Geoff Salmon, Giulio Starace, Greg Brockman, Hadi Salman, Haiming Bao, Haitang Hu, Hannah Wong, Haoyu Wang, Heather Schmidt, Heather Whitney, Heewoo Jun, Hendrik Kirchner, Henrique Ponde de Oliveira Pinto, Hongyu Ren, Huiwen Chang, Hyung Won Chung, Ian Kivlichan, Ian O'Connell, Ian O'Connell, Ian Osband, Ian Silber, Ian Sohl, Ibrahim Okuyucu, Ikai Lan, Ilya Kostrikov, Ilya Sutskever, Ingmar Kanitscheider, Ishaan Gulrajani, Jacob Coxon, Jacob Menick, Jakub Pachocki, James Aung, James Betker, James Crooks, James Lennon, Jamie Kiros, Jan Leike, Jane Park, Jason Kwon, Jason Phang, Jason Teplitz, Jason Wei, Jason Wolfe, Jay Chen, Jeff Harris, Jenia Varavva, Jessica Gan Lee, Jessica Shieh, Ji Lin, Jiahui Yu, Jiayi Weng, Jie Tang, Jieqi Yu, Joanne Jang, Joaquin Quinonero Candela, Joe Beutler, Joe Landers, Joel Parish, Johannes Heidecke, John Schulman, Jonathan Lachman, Jonathan McKay, Jonathan Uesato, Jonathan Ward, Jong Wook Kim, Joost Huizinga, Jordan Sitkin, Jos Kraaijeveld, Josh Gross, Josh Kaplan, Josh Snyder, Joshua Achiam, Joy Jiao, Joyce Lee, Juntang Zhuang, Justyn Harriman, Kai Fricke, Kai Hayashi, Karan Singhal, Katy Shi, Kavin Karthik, Kayla Wood, Kendra Rimbach, Kenny Hsu, Kenny Nguyen, Keren Gu-Lemberg, Kevin Button, Kevin Liu, Kiel Howe, Krithika Muthukumar, Kyle Luther, Lama Ahmad, Larry Kai, Lauren Itow, Lauren Workman, Leher Pathak, Leo Chen, Li Jing, Lia Guy, Liam Fedus, Liang Zhou, Lien Mamitsuka, Lilian Weng, Lindsay McCallum, Lindsey Held, Long Ouyang, Louis Feuvrier, Lu Zhang, Lukas Kondraciuk, Lukasz Kaiser, Luke Hewitt, Luke Metz, Lyric Doshi, Mada Aflak, Maddie Simens, Madelaine Boyd, Madeleine Thompson, Marat Dukhan, Mark Chen, Mark Gray, Mark Hudnall, Marvin Zhang, Marwan Aljubeh, Mateusz Litwin, Matthew Zeng, Max Johnson, Maya Shetty, Mayank Gupta, Meghan Shah, Mehmet Yatbaz, Meng Jia Yang, Mengchao Zhong, Mia Glaese, Mianna Chen, Michael Janner, Michael Lampe, Michael Petrov, Michael Wu, Michele Wang, Michelle Fradin, Michelle Pokrass, Miguel Castro, Miguel Oom Temudo de Castro, Mikhail Pavlov, Miles Brundage, Miles Wang, Minal Khan, Mira Murati, Mo Bavarian, Molly Lin, Murat Yesildal, Nacho Soto, Natalia Gimelshein, Natalie Cone, Natalie Staudacher, Natalie Summers, Natan LaFontaine, Neil Chowdhury, Nick Ryder, Nick Stathas, Nick Turley, Nik Tezak, Niko Felix, Nithanth Kudige, Nitish Keskar, Noah Deutsch, Noel Bundick, Nora Puckett, Ofir Nachum, Ola Okelola, Oleg Boiko, Oleg Murk, Oliver Jaffe, Olivia Watkins, Olivier Godement, Owen Campbell-Moore, Patrick Chao, Paul McMillan, Pavel Belov, Peng Su, Peter Bak, Peter Bakkum, Peter Deng, Peter Dolan, Peter Hoeschele, Peter Welinder, Phil Tillet, Philip Pronin, Philippe Tillet, Prafulla Dhariwal, Qiming Yuan, Rachel Dias, Rachel Lim, Rahul Arora, Rajan Troll, Randall Lin, Rapha Gontijo Lopes, Raul Puri, Reah Miyara, Reimar Leike, Renaud Gaubert, Reza Zamani, Ricky Wang, Rob Donnelly, Rob Honsby, Rocky Smith, Rohan Sahai, Rohit Ramchandani, Romain Huet, Rory Carmichael, Rowan Zellers, Roy Chen, Ruby Chen, Ruslan Nigmatullin, Ryan Cheu, Saachi Jain, Sam Altman, Sam Schoenholz, Sam Toizer, Samuel Miserendino, Sandhini Agarwal, Sara Culver, Scott Ethersmith, Scott Gray, Sean Grove, Sean Metzger, Shamez Hermani, Shantanu Jain, Shengjia Zhao, Sherwin Wu, Shino Jomoto, Shirong Wu, Shuaiqi, Xia, Sonia Phene, Spencer Papay, Srinivas Narayanan, Steve Coffey, Steve Lee, Stewart Hall, Suchir Balaji, Tal Broda, Tal Stramer, Tao Xu, Tarun Gogineni, Taya Christianson, Ted Sanders, Tejal Patwardhan, Thomas Cunninghman, Thomas Degry, Thomas Dimson, Thomas Raoux, Thomas Shadwell, Tianhao Zheng, Todd Underwood, Todor Markov, Toki Sherbakov, Tom Rubin, Tom Stasi, Tomer Kaftan, Tristan Heywood, Troy Peterson, Tyce Walters, Tyna Eloundou, Valerie Qi, Veit Moeller, Vinnie Monaco, Vishal Kuo, Vlad Fomenko, Wayne Chang, Weiyi Zheng, Wenda Zhou, Wesam Manassra, Will Sheu, Wojciech Zaremba, Yash Patil, Yilei Qian, Yongjik Kim, Youlong Cheng, Yu Zhang, Yuchen He, Yuchen Zhang, Yujia Jin, Yunxing Dai, and Yury Malkov. 2024. GPT-4o System Card. *arXiv preprint*.

Jiexing Qi, Jingyao Tang, Ziwei He, Xiangpeng Wan, Yu Cheng, Chenghu Zhou, Xinbing Wang, Quanshi Zhang, and Zhouhan Lin. 2022. RASAT: Integrating Relational Structures into Pretrained Seq2Seq Model for Text-to-SQL. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language*

Processing, pages 3215–3229, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Stephen Robertson and Hugo Zaragoza. 2009. The Probabilistic Relevance Framework: BM25 and Beyond. *Information Retrieval*, 3(4).

A Streamlit. 2024. Streamlit: A faster way to build and share data apps. *faster way to build and share data apps*.

Dingzirui Wang, Longxu Dou, Xuanliang Zhang, Qingfu Zhu, and Wanxiang Che. 2024a. DAC: Decomposed Automation Correction for Text-to-SQL. *arXiv preprint*. ArXiv:2408.08779 [cs].

Dingzirui Wang, Longxu Dou, Xuanliang Zhang, Qingfu Zhu, and Wanxiang Che. 2024b. Improving Demonstration Diversity by Human-Free Fusing for Text-to-SQL. *arXiv preprint*.

Run-Ze Wang, Zhen-Hua Ling, Jing-Bo Zhou, and Yu Hu. 2020. Tracking Interaction States for Multi-Turn Text-to-SQL Semantic Parsing. *arXiv preprint*.

Siqiao Xue, Caigao Jiang, Wenhui Shi, Fangyin Cheng, Keting Chen, Hongjun Yang, Zhiping Zhang, Jianshan He, Hongyang Zhang, Ganglin Wei, Wang Zhao, Fan Zhou, Danrui Qi, Hong Yi, Shaodong Liu, and Faqiang Chen. 2024. DB-GPT: Empowering Database Interactions with Private Large Language Models. *arXiv preprint*.

Tao Yu, Rui Zhang, He Yang Er, Suyi Li, Eric Xue, Bo Pang, Xi Victoria Lin, Yi Chern Tan, Tianze Shi, Zihan Li, Youxuan Jiang, Michihiro Yasunaga, Sungrok Shim, Tao Chen, Alexander Fabbri, Zifan Li, Luyao Chen, Yuwen Zhang, Shreya Dixit, Vincent Zhang, Caiming Xiong, Richard Socher, Walter S. Lasecki, and Dragomir Radev. 2019a. CoSQL: A Conversational Text-to-SQL Challenge Towards Cross-Domain Natural Language Interfaces to Databases. *arXiv preprint*.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. 2019b. Spider: A Large-Scale Human-Labeled Dataset for Complex and Cross-Domain Semantic Parsing and Text-to-SQL Task. *arXiv preprint*. ArXiv:1809.08887 [cs].

Tao Yu, Rui Zhang, Michihiro Yasunaga, Yi Chern Tan, Xi Victoria Lin, Suyi Li, Heyang Er, Irene Li, Bo Pang, Tao Chen, Emily Ji, Shreya Dixit, David Proctor, Sungrok Shim, Jonathan Kraft, Vincent Zhang, Caiming Xiong, Richard Socher, and Dragomir Radev. 2019c. SParC: Cross-Domain Semantic Parsing in Context. *arXiv preprint*.

Jichuan Zeng, Xi Victoria Lin, Caiming Xiong, Richard Socher, Michael R. Lyu, Irwin King, and Steven C. H. Hoi. 2020. Photon: A Robust Cross-Domain Text-to-SQL System. *arXiv preprint*. ArXiv:2007.15280 [cs].

Lu Zeng, Sree Hari Krishnan Parthasarathi, and Dilek Hakkani-Tur. 2023. N-Best Hypotheses Reranking for Text-to-SQL Systems. In *2022 IEEE Spoken Language Technology Workshop (SLT)*, pages 663–670.

Hanchong Zhang, Ruisheng Cao, Lu Chen, Hongshen Xu, and Kai Yu. 2023. ACT-SQL: In-Context Learning for Text-to-SQL with Automatically-Generated Chain-of-Thought. *arXiv preprint*.

Hanchong Zhang, Ruisheng Cao, Hongshen Xu, Lu Chen, and Kai Yu. 2024a. CoE-SQL: In-Context Learning for Multi-Turn Text-to-SQL with Chain-of-Editions. *arXiv preprint*.

Xuanliang Zhang, Dingzirui Wang, Longxu Dou, Qingfu Zhu, and Wanxiang Che. 2024b. Multi-Hop Table Retrieval for Open-Domain Text-to-SQL. *arXiv preprint*.

A Murre

In terms of implementation, we adopted the multi-round retrieval method proposed by (Zhang et al., 2024b), with the following steps:

Retrieval First, we extract table information from all databases in a multi-database environment and analyze the relevance of each table to the user's query. These tables are then ranked, and the top-k tables with the highest scores are retrieved.

Removal Next, a large language model (LLM) is used to rephrase the user's query, removing information related to the tables retrieved in the previous step. This step aims to eliminate tables that, although similar to the previously retrieved ones, are not relevant to the user's query.

Continue The retrieval process is then repeated from step 1, continuing until all relevant tables from the related databases have been retrieved, ensuring comprehensive coverage of all information pertinent to the user's query.

By employing this multi-round retrieval and information removal strategy, ABACUS-SQL can efficiently locate and extract the most relevant table information from the databases, thereby generating more accurate SQL queries.

B Fused

Regarding the specific implementation of data augmentation, we adopted the FUSED method (Wang et al., 2024b) to augment the dataset. Our specific implementation process is as follows:

Qwen2.5-Coder	Chase-C				SParC				CoSQL			
	QEX		IEX		QEX		IEX		QEX		IEX	
	7B	32B	7B	32B	7B	32B	7B	32B	7B	32B	7B	32B
ABACUS-SQL	45.5	53.5	15.0	23.1	68.4	69.6	46.9	47.4	70.6	73.1	42.3	42.7
- Pre-SQL	45.5	51.9	14.3	21.7	67.1	69.3	45.5	46.9	70.4	72.7	41.3	42.0
Δ	−0.0	−1.6	−0.7	−1.4	−1.3	−0.3	−1.4	−0.5	−0.2	−0.4	−1.0	−0.7
- Self-Debug	41.9	48.7	12.3	19.8	67.7	69.1	45.8	47.1	69.8	72.2	40.3	42.0
Δ	−3.6	−4.8	−2.7	−3.3	−0.7	−0.5	−1.1	−0.3	−0.8	−0.9	−2.0	−0.7

Table 3: Ablation studies removing Pre-SQL or Self-Debug, The Δ row represents the differences with respect to ABACUS-SQL.

<|im_start|>system
You are an expert in the field of databases and SQL, specializing in analyzing and writing SQL queries.
You can provide detailed, efficient, and accurate suggestions to improve database architecture and write optimized SQL statements for various database management systems.
Your tasks are:
1. Analyze the user's requirements carefully based on their questions and generate a reasoning process to assist in creating SQL.
2. Based on the reasoning process, generate an SQL query to answer the user's question.
3. Focus on the information provided in multi-turn conversations and use it to better generate SQL queries.
4. Extract general methods for generating SQL queries from the given examples and apply them to the user's questions, but do not directly copy the information from the examples.
The output format is as follows:
Rationale: {rationale step by step}
SQL:
```sql
{SQL}
```<|im_end|>
<|im_start|>user
Answer the question using the following examples:
[Demonstrations]
Generate an SQL to answer the question with the given schema:
[Schema]<|im_end|>
[Chat History]
<|im_start|>user
[Question]<|im_end|>
<|im_start|>assistant

Figure 3: The prompt used in Multi-Turn text-to-SQL

User data upload The dataset uploaded by the user must include database schema and example SQL queries along with their corresponding natural language question descriptions. The system will validate the format of the uploaded data to ensure it meets the basic requirements for augmentation. If no user data is uploaded, the system will use a default dataset for demonstration augmentation.

Sample sampling and clustering The system clusters the demonstrations based on structural features of the SQL queries (such as keywords, operators, etc.), forming different semantic categories. It then randomly samples demonstrations from each category, ensuring that the demonstrations input into the augmentation process exhibit significant diversity, thus avoiding overly similar demonstrations.

Sample fusion Using a large language model (LLM), the sampled demonstrations are used as inputs to generate new demonstrations through few-shot learning. The newly generated demonstrations combine features from multiple demonstrations while maintaining differences from existing ones, thereby enhancing the diversity of the overall demonstration pool.

Verification and filtering The system performs semantic consistency verification on the generated SQL queries and question descriptions, ensuring that the generated demonstrations are consistent with the database schema and the intended query. Low-quality or redundant demonstrations are removed through automated testing.

Demonstration pool update The augmented dataset is automatically added to the demonstra-

tion pool and merged with the existing dataset. The merged Demonstration pool is used for subsequent model inference and training, further improving the accuracy and adaptability of the generated SQL queries.

C Prompt

The prompt for ABACUS-SQL, shown in Figure 3, mainly consists of the following components: system prompts, few-shot examples, schema, and multi-turn dialogue.

D Ablation Studies

As shown in Table 3, we conduct ablation experiments on Pre-SQL and Self-debug methods, drawing the following conclusions:

Both methods improve system performance. Pre-SQL reduces the interference of irrelevant tables, decreasing complexity and improving query efficiency. Self-debug addresses post-generation errors, reducing mistakes caused by input ambiguity or understanding bias, further optimizing accuracy.

The results are particularly significant on Chinese datasets. Experiments show that when testing the Chase-C dataset on the Qwen2.5-Coder 32b model, Pre-SQL improves by 1.4 points on the IEX metric, while the Self-Debug method enhances the IEX metric by 5.1 points. Due to the ambiguity and complexity of the Chinese language, the system's semantic understanding requirements are higher. Pre-SQL helps reduce interference from irrelevant information, while the Self-Debug method corrects understanding biases. The synergy between these two methods significantly improves query accuracy and reliability, demonstrating a distinct advantage in handling Chinese natural language queries.