

SIP: Aligning LLM-based TableQA with Table Semantics via SQL Logic

Zhen Yang^{1,2,3}, Ziwei Du^{1,2,3}, Minghan Zhang^{1,2,3}, Wei Du^{1,2,3}
Haodong Zou^{1,2,3}, Shicheng Tan^{1,2,3}, Jie Chen^{1,2,3}, Shu Zhao^{1,2,3*}

¹School of Computer Science and Technology, Anhui University

²Information Materials and Intelligent Sensing Laboratory of Anhui Province, Anhui University

³Anhui Provincial Key Laboratory of Security Artificial Intelligence, Anhui University

*zhaoshuzs2002@hotmail.com

Abstract

Large Language Models (LLMs) have achieved strong performance on Table Question Answering (TableQA), yet their reasoning processes often remain misaligned with the underlying table semantics. This misalignment can lead to decompositions and validation steps that fail to faithfully capture the table operations required by the question. In this paper, we introduce SIP, a SQL Logic-aligned framework for improving the semantic consistency of LLM-based table reasoning. SIP is built around SQL Logic, a natural-language abstraction of standard SQL operators that explicitly represents operations such as projection, filtering, ordering, and aggregation while avoiding the syntactic brittleness of executable SQL. SQL Logic serves as a shared operator-level representation that guides sub-question decomposition and supports operator-aware validation in both Direct Prompting and Python Agent settings. Experiments on three TableQA benchmarks show that SIP achieves state-of-the-art performance and consistently improves robustness across diverse LLMs, demonstrating the value of explicit semantic representations for structured table reasoning.

1 Introduction

Tables are a fundamental form of structured data widely used in domains such as finance, public records, and scientific analysis (Leventidis et al., 2024; Dong and Oyamada, 2022; Yu et al., 2025b; Yuan et al., 2025). Table Question Answering (TableQA) aims to answer natural-language questions through reasoning grounded in tabular structures (Xiao et al., 2025; Sui et al., 2024; Zhao et al., 2023b). With the rapid progress of Large Language Models (LLMs), recent studies have shifted from specialized table encoders to LLM-based reasoning pipelines, achieving strong performance through question decomposition, sub-question reasoning,

and self-validation (Xia et al., 2023; Li et al., 2024; Deng et al., 2024).

Despite these advances, a persistent misalignment remains between LLM reasoning and underlying table semantics. LLMs reason primarily in natural-language space, whereas table reasoning requires compositional, symbolic, and schema-dependent operations (Guan et al., 2024; Zhang et al., 2023; Pal et al., 2023; Li et al., 2023). This gap leads to several limitations. First, natural-language decomposition methods such as CoT prompting often fail to capture operator-level table semantics, producing fluent but structurally incorrect reasoning steps (Brown et al., 2020; Lewkowycz et al., 2022; Liu et al., 2022a). Second, self-validation relies heavily on the model’s own judgment and may reinforce incorrect reasoning paths when structural grounding is absent (Madaan et al., 2023; Shinn et al., 2023; Qi et al., 2024). Third, SQL-based approaches provide explicit semantics but depend on brittle executable syntax: minor variations in formatting or column names can cause execution failure even when the intended operation is correct (Cheng et al., 2023; Xie et al., 2022; Ou and Liu, 2022). These limitations suggest that existing paradigms lack a stable semantic representation that can align natural-language reasoning with structured table operations.

As illustrated in Fig. 1, questions that appear simple at the linguistic level often require operations such as projection, filtering, ordering, grouping, and aggregation. When these operations are not explicitly represented, LLM reasoning may drift from the intended table semantics, leading to inconsistent or incorrect answers. A robust TableQA system therefore needs a representation that captures SQL-style compositionality while remaining compatible with natural-language reasoning.

To address this gap, we introduce SIP, a SQL logic-inspired framework that aligns LLM reasoning with structured table semantics. At the core

*Corresponding authors

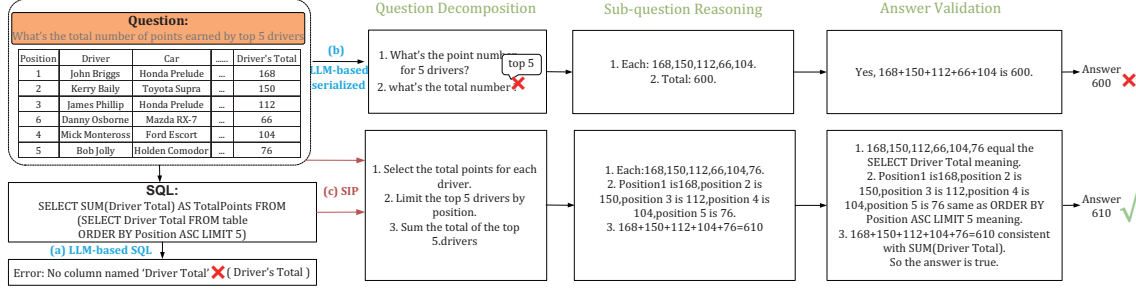


Figure 1: (a) Direct LLM-based SQL execution is sensitive to minor syntax errors. (b) Existing serialized methods directly decompose questions, but insufficient structural grounding can lead to misleading sub-questions and error propagation. (c) SIP uses SQL Logic to represent table operations, guide decomposition, and validate reasoning, thereby improving TableQA reliability.

of SIP is SQL Logic, a formally defined natural-language abstraction of SQL operators. Unlike executable SQL, SQL Logic captures the semantic structure of relational operations without relying on fragile syntax, serving as an operator-level representation of table reasoning in natural language.

SIP leverages SQL Logic in two key components. First, logic-aligned decomposition uses SQL Logic to guide the LLM in generating sub-questions that reflect the operator structure required by the question. Second, logic-based validation uses SQL Logic to verify intermediate reasoning steps across both Direct Prompting (DP) and Python-Agent (PyAgent) reasoning modes, reducing hallucination and error propagation. By grounding decomposition, reasoning, and validation in a shared logical abstraction, SIP improves reasoning consistency and stabilizes the final answer through joint inference.

We evaluate SIP on multiple TableQA benchmarks and diverse LLMs. Experimental results show that SIP consistently improves accuracy, robustness, and reasoning consistency, demonstrating the importance of explicit semantic logic for aligning LLM reasoning with structured data.

In summary, our contributions are:

- We propose SQL Logic, a natural-language abstraction of SQL operators that provides an operator-level semantic bridge between LLM reasoning and tabular structures.
- We introduce SIP, a logic-aligned reasoning framework that uses SQL Logic to guide decomposition and validate intermediate steps across multiple reasoning modes.
- Extensive experiments show that SIP improves accuracy, consistency, and robustness

on standard TableQA benchmarks and generalizes across diverse LLMs.

2 Related Work

2.1 Table Question Answering

Tables are a fundamental form of structured data, and Table Question Answering (TableQA) requires reasoning over schemas, rows, and columns (Wang et al., 2021; Guo et al., 2021; Chen et al., 2020b; Shraga et al., 2020). Early methods typically translate natural-language questions into executable SQL or logical forms (Zhong et al., 2017; Eisen-schlos et al., 2020; Liu et al., 2022b). While these methods provide explicit execution semantics, they often rely on costly annotations and show limited generalization beyond training distributions.

2.2 LLM-based TableQA Methods

Recent TableQA methods exploit LLMs through SQL generation, retrieval, decomposition, and tool use (Xiao et al., 2025; Zhao et al., 2023b). SQL-oriented methods, such as BINDER (Cheng et al., 2023), DTE (Wang et al., 2023), and SYN-TQA (Zhang et al., 2024a), benefit from structured query semantics but remain sensitive to syntax errors, column-name mismatches, and formatting variations. Retrieval, decomposition, and refinement methods, including TACR (Wu et al., 2023), ITR (Lin et al., 2023), Cabinet (Patnaik et al., 2024), DATER (Ye et al., 2023), TIDE (Yang et al., 2025), Table-Critic (Yu et al., 2025a), RE-FOCUS (Fu et al.), TabFlash (Kim et al., 2026), and DataFactory (Wang et al., 2026), reduce table noise, improve evidence selection, decompose complex questions, or verify reasoning outputs, but their intermediate steps often lack explicit operator-level grounding. Tool-based approaches, such

Table 1: Semantic operators for single-table SQL used by SIP.

Operator	Meaning	SQL Constructs
π	Projection	SELECT, DISTINCT
σ	Selection	WHERE
γ, Σ	Grouping / Aggregation	GROUP BY, HAVING, SUM, AVG, MIN, MAX, COUNT
τ	Ordering	ORDER BY (ASC/DESC)
λ	Limiting	LIMIT, TOP
ρ	Column renaming	AS
$\odot$	Column arithmetic	+, -, *, /

Table 2: Representative operator→natural language mappings used in SIP.

Operator	Natural Language Intent	Example Expression in SQL Logic
π	Select relevant column(s)	“retrieve the column X ”; “extract each value of X ”
σ	Filter rows based on conditions	“keep rows where <i>condition holds</i> ”; “only consider records satisfying X ”
γ / Σ	Group / Aggregate values	“for each group compute <i>sum/average/max/min</i> ”; “aggregate values over X ”
τ	Order rows	“sort rows by X ascending/descending”; “rank entries by X ”
λ	Limit scope	“keep the top k entries”; “consider only the first k results”
ρ	Rename column	“refer to X as Y ”; “label this value as Y ”
$\odot$	Column arithmetic	“compute $X + Y$ ”; “use the difference between A and B ”

as CHAIN-OF-TABLE (Wang et al., 2024), ReAcTable (Zhang et al., 2024b), and Mix-SC (Liu et al., 2024a), improve flexibility through table transformations or Python execution, yet remain vulnerable to planning and validation errors induced by LLMs. Intermediate representations such as structured embeddings, pseudo-programs, or rationales (Yin et al., 2020; Liu et al., 2022c; Deng et al., 2020) improve interpretability, but they usually do not explicitly encode relational operations, making self-validation prone to assessing linguistic plausibility rather than structural correctness (Madaan et al., 2023; Shinn et al., 2023; Qi et al., 2024).

2.3 Our Work

SIP introduces SQL Logic, a natural-language abstraction of SQL operators that preserves compositional table semantics while avoiding the syntactic fragility of executable SQL. Unlike prior decomposition or rationale-based methods, SQL Logic provides an operator-level semantic reference for both sub-question generation and intermediate validation. By integrating SQL Logic with Direct Prompting and Python-Agent reasoning through

joint inference, SIP aligns natural-language reasoning with structured table operations.

3 Method

We present SIP, a logic-aligned framework for LLM-based TableQA. SIP introduces SQL Logic, an intermediate representation that expresses SQL-style table operations in natural language. SQL Logic serves as the semantic backbone of the framework, guiding decomposition, reasoning, validation, and joint inference. Fig. 2 illustrates the overall framework.

3.1 Semantic Operators for Table SQL

We focus on single-table TableQA, which covers most widely used benchmarks and avoids additional complications from joins or schema linking. SIP abstracts SQL operations into a compact operator set, including projection, selection, aggregation, ordering, limiting, renaming, and arithmetic transformations, as summarized in Table 1. These operators capture the main relational operations required by compositional table questions while decoupling reasoning from executable SQL syntax. They also define a finite semantic space for operator-aware

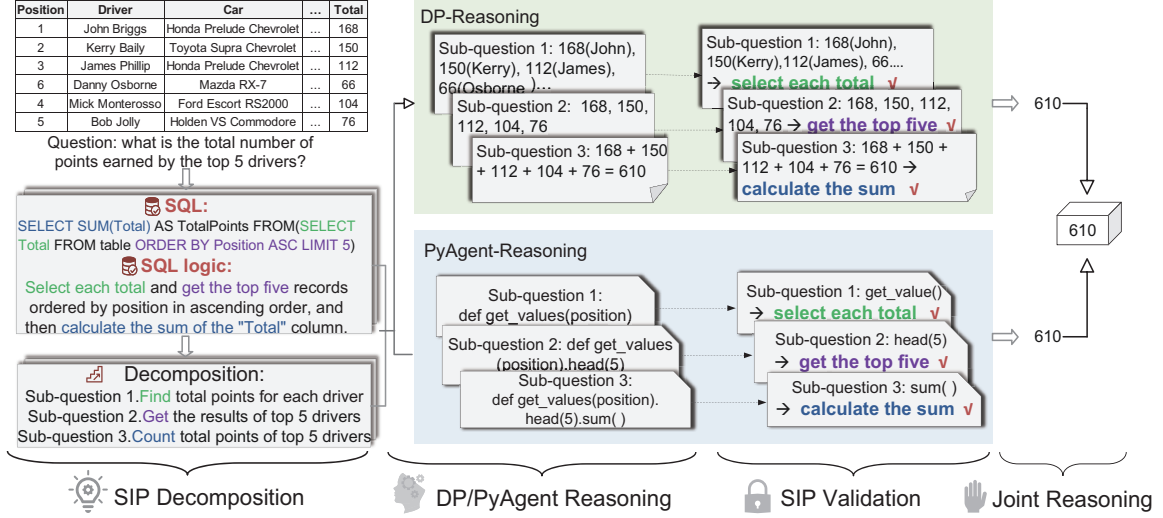


Figure 2: Overview of SIP. SQL semantics are transformed into SQL Logic, which guides question decomposition. Sub-questions and SQL Logic are then used by DP and PyAgent modules. Semantic validation checks intermediate results, and joint inference aggregates outputs into the final answer.

decomposition and validation.

3.2 SQL Logic

Given a SQL query q over table T , SIP converts it into a SQL Logic sequence. Unlike executable SQL, SQL Logic is not executed; instead, it verbalizes relational operations while preserving their compositional semantics.

Operator extraction. A deterministic SQL parser extracts the ordered operator sequence:

$$O(q) = \langle o_1, o_2, \dots, o_k \rangle, \quad (1)$$

where each o_i denotes a relational transformation, such as filtering, ordering, or aggregation.

Natural-language abstraction. Each operator is mapped to a natural-language logic unit:

$$l_i = \text{NL}(o_i), \quad (2)$$

yielding the SQL Logic sequence:

$$\mathcal{L}(q) = \langle l_1, l_2, \dots, l_k \rangle. \quad (3)$$

SQL Logic thus provides an interpretable semantic scaffold for LLM reasoning without requiring syntactically valid SQL. Representative mappings are shown in Table 2.

3.3 SIP Logic-Aligned Decomposition

Given question Q , table T , and SQL Logic $\mathcal{L}(q)$, SIP generates sub-questions:

$$S = \{S_1, \dots, S_m\} = f_{\text{decomp}}(Q, T, \mathcal{L}(q)), \quad (4)$$

where each S_i is a natural-language sub-question. To align decomposition with SQL Logic, SIP enforces two conditions.

Soundness. Each sub-question must correspond to at least one logic unit:

$$\forall S_i, \exists l_j \in \mathcal{L}(q) \text{ such that } S_i \models l_j. \quad (5)$$

Completeness. The sub-question set must cover all logic units:

$$\bigcup_{i=1}^m S_i \equiv \mathcal{L}(q). \quad (6)$$

Here, $\models$ denotes semantic consistency between a sub-question and a logic unit, and $\equiv$ denotes coverage equivalence.

3.4 DP/PyAgent Reasoning

SIP reasons over each sub-question through two complementary paths.

Direct Prompting (DP). The LLM directly answers each sub-question:

$$A_i^{\text{DP}} = \text{LLM}(T, S_i). \quad (7)$$

Python Agent Execution (PyAgent). The LLM generates executable code:

$$c_i = \text{LLM}^{\text{code}}(T, S_i), \quad (8)$$

$$A_i^{\text{Py}} = \text{exec}(c_i), \quad (9)$$

where c_i is a Python snippet implementing the operation required by S_i . DP better captures linguistic cues, while PyAgent provides deterministic execution for filtering, sorting, and aggregation.

3.5 SIP Semantic Validation

For each sub-question S_i , SIP validates its intermediate answer A_i against the corresponding SQL Logic unit l_i :

$$\text{Valid}(A_i, l_i) = \begin{cases} 1, & \text{if } A_i \text{ is consistent with } l_i, \\ 0, & \text{otherwise.} \end{cases} \quad (10)$$

Validation is operator-aware: selection checks filtering conditions, ordering checks rank consistency, and aggregation verifies numerical correctness. If validation fails, SIP regenerates only the failed step, reducing error propagation without restarting the full reasoning process.

3.6 Joint Reasoning

SIP aggregates validated DP and PyAgent outputs by voting:

$$A_{\text{final}} = \arg \max_{a \in \mathcal{A}} \left(\sum_{i=1}^m \mathbf{1}(A_i^{\text{DP}} = a) + \sum_{j=1}^n \mathbf{1}(A_j^{\text{Py}} = a) \right). \quad (11)$$

This reduces variance from hallucinated reasoning or execution failures.

3.7 Case Illustration

For the question in Fig. 2, “What is the total number of points earned by the top five drivers?”, a corresponding SQL query is: `SELECT SUM(Total) FROM (SELECT Total FROM T ORDER BY Position ASC LIMIT 5)` SIP extracts $O(q) = \langle \pi, \tau, \lambda, \Sigma \rangle$, where π retrieves *Total*, τ orders rows by *Position*, λ keeps the top five rows, and Σ computes their sum. These operators are verbalized into SQL Logic units, which guide decomposition into aligned sub-questions. DP and PyAgent answer them independently, validation checks projection, ordering, and aggregation semantics, and joint inference returns the final answer: 610.

The complete algorithm is provided in Appendix A.

4 Experimental Setup

4.1 Datasets

Following prior TableQA studies (Wang et al., 2024; Liu et al., 2024b; Zhang et al., 2024b; Liu et al., 2024a), we evaluate SIP on three benchmarks: WikiTableQuestions (WTQ) (Pasupat and Liang, 2015), which contains 4,344 compositional questions involving filtering, ordering, and aggregation; TabFact (Chen et al., 2020a), which includes 2,024 table-based fact verification claims;

Table 3: Results on WTQ with GPT-3.5. SIP-DP means SIP use DP alone and SIP-PyAgent use PyAgent alone. SIP-DP&PyAgent shows the joint result.

Methods	Acc.
ReasTAP (22’ EMNLP)	58.30
TaCube (22’ EMNLP)	60.90
OmniTab (22’ NAACL)	61.30
Codex SQL (23’ ICLR)	61.10
BINDER (23’ ICLR)	64.60
DATER (23’ SIGIR)	65.90
DTE (23’ ACL)	54.20
TACR (23’ arXiv)	60.20
ITR (23’ ACL)	63.40
StructGPT (23’ EMNLP)	57.00
Liu et al. (24’ arXiv)	55.80
Cabinet (24’ ICLR)	69.10
CHAIN-OF-TABLE (24’ ICLR)	59.94
ReAcTable (24’ VLDB)	68.00
SYNTQA (24’ EMNLP)	70.40
Mix-SC(DP&Agent) (24’ NAACL)	73.65
Table-Critic (25’ ACL)	72.70
REFOCUS (25’ ICLR)	72.20
TIDE (DP&Agent) (25’ ICLR)	75.00
DataFactory (26’ IPM)	65.30
SIP-DP	71.39
SIP-PyAgent	73.65
SIP-DP&PyAgent	75.35

and FetaQA (Nan et al., 2022), which contains 2,003 free-form questions requiring reasoning over non-contiguous table segments.

4.2 Evaluation Metrics and Implementation Details

Following prior work (Cheng et al., 2023; Ye et al., 2023), we report exact match accuracy for WTQ and TabFact, and BLEU (Papineni et al., 2002) for FetaQA. GPT-3.5 is used as the default LLM with temperature 0.8 for comparability with existing studies (Cheng et al., 2023; Ye et al., 2023; Liu et al., 2024a). To evaluate generality, we further test SIP with open-source LLaMA 2 models and closed-source LLMs, including GPT-4, GLM-4, Gemini 1.5, and Claude 3.5, all using temperature 0.8. SIP is implemented with HuggingFace Transformers and vLLM; open-source models are run on RTX 4090 GPUs with FP16 precision, while closed-source models are accessed via official APIs. Prompts for decomposition, reasoning, and valida-

tion are kept consistent across LLMs.

Due to dataset-specific prompting and reporting protocols, some baselines cannot be reliably reproduced across all benchmarks. Therefore, compared methods vary by dataset following their original evaluation settings.

Table 4: Results on TabFact with GPT-3.5. SIP-DP means SIP use DP alone and SIP-PyAgent use PyAgent alone. SIP-DP&PyAgent shows the joint reasoning result.

Methods	Acc.
SaMoE (22’ ACL)	86.70
SASP (22’ ACL)	77.00
T5-3B (22’ EMNLP)	83.68
Codex SQL (23’ ICLR)	80.70
BINDER (23’ ICLR)	85.10
DATER (23’ SIGIR)	85.60
StructGPT (23’ EMNLP)	87.30
CHAIN-OF-TABLE (24’ ICLR)	80.20
ReAcTable (24’ VLDB)	86.10
Mix-SC(DP&PyAgent) (24’ ACL)	88.50
TIDE (DP&Agent) (25’ ICLR)	89.82
Table-Critic (25’ ACL)	88.30
REFOCUS (25’ ICLR)	87.10
DataFactory (26’ IPM)	80.10
SIP-DP	78.51
SIP-PyAgent	87.90
SIP-DP&PyAgent	89.33

4.3 Baselines

To evaluate SIP, we compare it with two groups of baselines: pre-training methods and LLM-based serialized methods.

Pre-training Methods. We compare with SASP (Ou and Liu, 2022), T5-3B (Xie et al., 2022), OmniTab (Jiang et al., 2022), PeasTAP (Zhao et al., 2022), TaCube (Zhou et al., 2022a), SaMoE (Zhou et al., 2022b), Task Configs (Chen et al., 2023), TARGET (Ji et al., 2025), TabCot (Chen, 2023), TAG-QA (Zhao et al., 2023a), and UniTabPT (Sarkar and Lausen, 2023).

LLM-based Serialized Methods. We compare with Codex SQL (Cheng et al., 2023), BINDER (Cheng et al., 2023), DTE (Wang et al., 2023), TACR (Wu et al., 2023), ITR (Lin et al., 2023), Cabinet (Patnaik et al., 2024), DATER (Ye et al., 2023), StructGPT (Jiang et al., 2023), Liu et al. (Liu et al., 2024b), CHAIN-OF-TABLE (Wang

Table 5: Results on FetaQA dataset.

Methods	Acc.
Task Configs (23’ ACL)	27.80
T5-large (23’ SIGIR)	30.54
TARGET (24’ NeurIPS)	24.13
Codex (23’ SIGIR)	27.96
DATER (23’ SIGIR)	30.92
TabCot (23’ EACL)	29.36
ReAcTable (24’ VLDB)	30.43
DataFactory (26’ IPM)	30.64
SIP-DP	31.11
SIP-PyAgent	23.92
SIP-DP&PyAgent	31.33

et al., 2024), ReAcTable (Zhang et al., 2024b), SYNTQA (Zhang et al., 2024a), Mix-SC (Liu et al., 2024a), TIDE (Yang et al., 2025), Table-Critic (Yu et al., 2025a), REFOCUS (Fu et al.) and DataFactory (Wang et al., 2026).

5 Experimental Results

5.1 Main Results

Table 6: Number of Answer Selections in SIP Joint Reasoning.

PyAgent	DP	WTQ	TabFact	FetaQA
1	1	64.99	84.34	27.15
3	3	71.82	87.99	29.55
5	5	73.30	87.25	31.07
1	5	67.03	79.74	28.46
5	1	74.24	88.78	29.84
3	5	70.35	80.63	30.77
5	3	75.35	89.33	31.33

Tables 3–5 report the main results. Baseline results are taken from the original papers, while SIP results are averaged over multiple runs. On WikiTableQuestions, SIP achieves 75.35% accuracy, outperforming strong baselines such as TIDE (Yang et al., 2025) and Mix-SC (Liu et al., 2024a). Both SIP-DP and SIP-PyAgent obtain competitive performance, showing that SQL Logic provides effective structural grounding for different reasoning modes, while joint inference further improves performance by combining DP’s linguistic reasoning with PyAgent’s executable computation.

Table 7: Ablation results on datasets.

Methods	WTQ	TabFact	FetaQA
SIP-DP	71.39	78.51	31.11
w/o Decomposition	60.74 (↓ 10.65)	73.27 (↓ 5.24)	29.47 (↓ 1.64)
w/o Validation	63.33 (↓ 8.06)	75.46 (↓ 3.05)	30.15 (↓ 0.96)
SIP-PyAgent	73.65	87.90	23.92
w/o Decomposition	61.41 (↓ 12.24)	83.63 (↓ 4.27)	21.01 (↓ 2.91)
w/o Validation	65.47 (↓ 8.18)	84.78 (↓ 3.12)	22.42 (↓ 1.50)
SIP-DP&PyAgent	75.35	89.33	31.33
w/o Decomposition	63.43 (↓ 11.92)	84.51 (↓ 4.82)	29.88 (↓ 1.45)
w/o Validation	68.29 (↓ 7.06)	86.93 (↓ 2.40)	31.18 (↓ 0.15)

On TabFact, SIP reaches 89.33% accuracy, outperforming most baselines and ranking second only to TIDE. This suggests that SQL Logic is effective for fact verification, although TIDE’s triple-based representation may better capture some implicit comparisons or fine-grained inter-row relations. On FetaQA, SIP achieves the best performance among all compared methods, indicating that SQL Logic-aligned decomposition and validation help maintain consistency in long-form table reasoning.

5.2 Effect of Answer Selection Frequency

Table 6 shows that increasing the number of sampled answers generally improves performance, as voting reduces the impact of occasional reasoning failures. The best results are obtained with five PyAgent and three DP selections, suggesting that joint inference benefits from a PyAgent-skewed balance: PyAgent provides stable execution for numerical and structural operations, while DP helps resolve linguistic ambiguity. Configurations dominated by a single reasoning mode are less effective, confirming the complementarity of the two pathways.

5.3 Ablation Study

We ablate SIP’s two core components: SQL logic-guided decomposition and semantic validation. Results are shown in Table 7.

Removing SIP decomposition. We replace logic-guided decomposition with standard Chain-of-Thought decomposition, allowing the LLM to generate sub-questions without SQL Logic. This variant leads to the largest performance drops across all settings, showing that unguided decomposition often fails to capture the true operator structure and causes cascading reasoning errors.

Removing SIP validation. Following (Huang et al., 2024), we replace logic-based validation with self-validation. Removing validation also reduces accuracy, confirming its role in correcting intermediate reasoning drift. The drop is generally larger for DP than PyAgent, since executable code already imposes partial structural constraints.

5.4 Experiment Analysis

Comparison with direct SQL execution. Table 8 compares SIP with direct SQL execution methods. SQL-oriented models are sensitive to syntax errors, column-name mismatches, and formatting variations, which can cause execution failures even when the intended operation is correct. SIP avoids this issue by abstracting SQL intent into SQL Logic, preserving relational semantics without requiring executable SQL. As a result, SIP achieves 75.35% accuracy, outperforming the strongest SQL baseline by over 15 points.

Table 8: Comparison of direct SQL execution.

Models	Accuracy(%)
WizardCoder	50.60
SQLCoder	52.30
Codex-SQL	60.20
SIP	75.35

Generalization across models. Table 9 shows that SIP improves both open-source and proprietary LLMs, indicating that its gains are model-agnostic. The improvements also hold for strong models such as GPT-4, Claude 3.5, and DeepSeek R1, suggesting that SQL Logic remains useful beyond base model reasoning ability. Larger or stronger models often obtain larger gains, implying that they can better exploit SIP’s operator-level structure.

Table 9: Comparison of LLMs with and without SIP.

Models	Accuracy(%)
<i>Open-source LLMs Methods</i>	
Llama 2-7b	41.85
Llama 2-7b + SIP	44.16 (2.31 ↑)
Llama 2-13b	50.78
Llama 2-13b + SIP	53.37(1.59 ↑)
DeepSeek R1	78.18
DeepSeek R1 + SIP	80.23 (2.05 ↑)
<i>Close-source LLMs Methods</i>	
GLM 4	65.43
GLM 4 + SIP	72.23(6.80 ↑)
GPT 4	67.84
GPT 4 + SIP	75.33(7.49 ↑)
Gemini 1.5	61.94
Gemini 1.5 + SIP	65.84 (3.90 ↑)
Claude 3.5	71.47
Claude 3.5 + SIP	75.80 (4.33 ↑)

API call efficiency. Table 10 compares SIP with Chain-of-Table in API call complexity. Chain-of-Table repeatedly invokes the LLM for step prediction, argument generation, and table transformation, leading to $3N$ calls. SIP performs SQL Logic extraction and decomposition once, and then reuses the logic units for validation, resulting in $N + 2$ calls. This reduces API cost and limits error accumulation from repeated serialized interactions.

Table 10: Comparison of API Call Complexity

Methods	API Call Operations
Chain-of-Table	(Next step + Argument + Transform) $\times$ Iterations $N = 3N$
SIP	(SQL logic + Decompose) + Valid $\times$ Iterations $N = N + 2$

Impact of question complexity. Fig. 3 shows that decomposition granularity affects DP and PyAgent differently. SIP-DP benefits more from questions with five or more sub-questions, where linguistic scaffolding helps capture multi-hop dependencies. SIP-PyAgent performs better on questions with fewer but more functional operations, such as filtering or aggregation. Joint inference combines these strengths across different complexity levels.

Impact of table size. Fig. 4 shows that SIP-

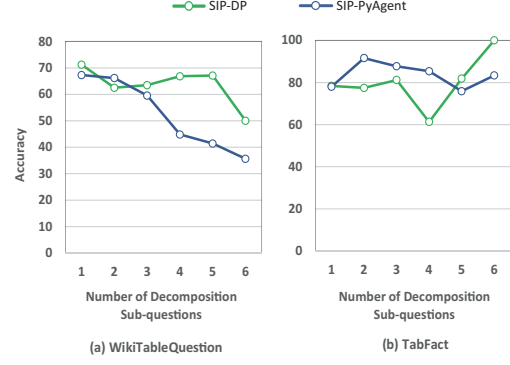


Figure 3: Impact of the number of decomposition Sub-questions on TableQA.

PyAgent becomes more effective as table size increases. For small tables, DP and PyAgent perform similarly, but DP degrades on larger tables due to context truncation and attention dilution. PyAgent remains more stable by executing filtering, sorting, and aggregation over full table contents.

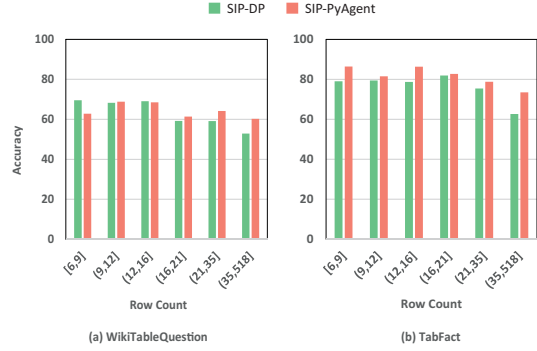


Figure 4: Impact of table size on TableQA performance.

Data contamination and Error Analysis. We further report data-contamination checks and sampled error analysis in Appendix B and C.

6 Conclusion

We presented SIP, a SQL Logic-aligned framework that bridges natural-language reasoning and structured table semantics for LLM-based TableQA. By using compositional natural-language logic units to guide decomposition and validation, SIP avoids executable SQL’s syntactic fragility and consistently improves performance and robustness across datasets and LLMs.

Limitations

SIP depends on the quality of the SQL used to derive SQL Logic. Although SQL Logic mitigates

the brittleness of direct SQL execution, semantic errors in generated SQL, such as missing constraints, incorrect operators, or mismatched column semantics, may propagate into SQL Logic and misguide decomposition or validation. Moreover, SQL Logic abstracts explicit relational operations but may overlook implicit constraints or nuanced interpretations in natural-language questions. Future work will explore more robust SQL generation and hybrid symbolic-neural refinement to improve the reliability of SQL Logic.

Acknowledgements

Our work is supported by the National Natural Science Foundation of China (62476003), Anhui Province Excellent Scientific Research and Innovation Team (2024AH010004), Anhui Provincial Natural Science Foundation - Water Science Joint Fund (2408055US006), the University Synergy Innovation Program of Anhui Province (GXXT-2023-050), and SMP-Zhipu.AI Large Model Cross-Disciplinary Fund (SMP-Zhipu20240210). We also acknowledge the support from Zhipu AI-Anhui University Joint Research Center, and the High-Performance Computing Platform of Anhui University.

References

- Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901.
- Jifan Chen, Yuhao Zhang, Lan Liu, Rui Dong, Xinchu Chen, Patrick Ng, William Yang Wang, and Zhiheng Huang. 2023. [Improving cross-task generalization of unified table-to-text models with compositional task configurations](#). In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 5523–5539. Association for Computational Linguistics.
- Wenhu Chen. 2023. [Large language models are few\(1\)-shot table reasoners](#). In *Findings of the Association for Computational Linguistics: EACL 2023, Dubrovnik, Croatia, May 2-6, 2023*, pages 1090–1100. Association for Computational Linguistics.
- Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyu Zhou, and William Yang Wang. 2020a. [Tabfact: A large-scale dataset for table-based fact verification](#). In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.
- Zhiyu Chen, Mohamed Trabelsi, Jeff Hefflin, Yinan Xu, and Brian D. Davison. 2020b. [Table search using a deep contextualized language model](#). In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pages 589–598. ACM.
- Zhoujun Cheng, Tianbao Xie, Peng Shi, Chengzu Li, Rahul Nadkarni, Yushi Hu, Caiming Xiong, Dragomir Radev, Mari Ostendorf, Luke Zettlemoyer, Noah A. Smith, and Tao Yu. 2023. [Binding language models in symbolic languages](#). In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.
- Naihao Deng, Zhenjie Sun, Ruiqi He, Aman Sikka, Yulong Chen, Lin Ma, Yue Zhang, and Rada Mihalcea. 2024. [Tables as images? exploring the strengths and limitations of llms on multimodal representations of tabular data](#). *CoRR*, abs/2402.12424.
- Xiang Deng, Huan Sun, Alyssa Lees, You Wu, and Cong Yu. 2020. [Turl: table understanding through representation learning](#). *Proceedings of the VLDB Endowment*, 14(3):307–319.
- Yuyang Dong and Masafumi Oyamada. 2022. [Table enrichment system for machine learning](#). In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*, pages 3267–3271. ACM.
- Julian Martin Eisenschlos, Syrine Krichene, and Thomas Müller. 2020. [Understanding tables with intermediate pre-training](#). In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 281–296. Association for Computational Linguistics.
- Xingyu Fu, Minqian Liu, Zhengyuan Yang, John Corring, Yijuan Lu, Jianwei Yang, Dan Roth, Dinei A. F. Florêncio, and Cha Zhang. [Refocus: Visual editing as a chain of thought for structured image understanding](#). In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of Machine Learning Research*.
- Che Guan, Mengyu Huang, and Peng Zhang. 2024. [MFORT-QA: multi-hop few-shot open rich table question answering](#). In *Proceedings of the 2024 10th International Conference on Computing and Artificial Intelligence, ICCAI 2024, Bali Island, Indonesia, April 26-29, 2024*, pages 434–442. ACM.
- Huifeng Guo, Wei Guo, Yong Gao, Ruiming Tang, Xiquang He, and Wenzhi Liu. 2021. [Scalefreectr: Mixcache-based distributed training system for CTR models with huge embedding table](#). In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*,

- Virtual Event, Canada, July 11-15, 2021, pages 1269–1278. ACM.
- Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. 2024. [Large language models cannot self-correct reasoning yet](#). In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.
- Xingyu Ji, Parker Glenn, Aditya G. Parameswaran, and Madelon Hulsebos. 2025. [TARGET: benchmarking table retrieval for generative tasks](#). *CoRR*, abs/2505.11545.
- Jinhao Jiang, Kun Zhou, Zican Dong, Keming Ye, Xin Zhao, and Ji-Rong Wen. 2023. [Structgpt: A general framework for large language model to reason over structured data](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 9237–9251. Association for Computational Linguistics.
- Zhengbao Jiang, Yi Mao, Pengcheng He, Graham Neubig, and Weizhu Chen. 2022. [Ommitab: Pretraining with natural and synthetic data for few-shot table-based question answering](#). In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 932–942. Association for Computational Linguistics.
- Jongha Kim, Minseong Bae, Sanghyeok Lee, Jinsung Yoon, and Hyunwoo J Kim. 2026. [Tabflash: Efficient table understanding with progressive question conditioning and token focusing](#). In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 22573–22581.
- Aristotelis Leventidis, Martin Pekár Christensen, Matteo Lissandrini, Laura Di Rocco, Katja Hose, and Renée J. Miller. 2024. [A large scale test corpus for semantic table search](#). In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2024, Washington DC, USA, July 14-18, 2024*, pages 1142–1151. ACM.
- Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay V. Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. 2022. [Solving quantitative reasoning problems with language models](#). In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.
- Hongxin Li, Jingran Su, Yuntao Chen, Qing Li, and Zhaoxiang Zhang. 2023. [Sheetcopilot: Bringing software productivity to the next level through large language models](#). In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.
- Peng Li, Yeye He, Dror Yashar, Weiwei Cui, Song Ge, Haidong Zhang, Danielle Rifinski Fainman, Dongmei Zhang, and Surajit Chaudhuri. 2024. [Table-gpt: Table fine-tuned GPT for diverse table tasks](#). *Proc. ACM Manag. Data*, 2(3):176.
- Weizhe Lin, Rexhina Blloshmi, Bill Byrne, Adrià de Gispert, and Gonzalo Iglesias. 2023. [An inner table retriever for robust table question answering](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 9909–9926. Association for Computational Linguistics.
- Jiacheng Liu, Skyler Hallinan, Ximing Lu, Pengfei He, Sean Welleck, Hannaneh Hajishirzi, and Yejin Choi. 2022a. [Rainier: Reinforced knowledge introspector for commonsense question answering](#). In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 8938–8958. Association for Computational Linguistics.
- Qian Liu, Bei Chen, Jiaqi Guo, Morteza Ziyadi, Zeqi Lin, Weizhu Chen, and Jian-Guang Lou. 2022b. [TAPEX: table pre-training via learning a neural SQL executor](#). In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.
- Qian Liu, Bei Chen, Jiaqi Guo, Morteza Ziyadi, Zeqi Lin, Weizhu Chen, and Jian-Guang Lou. 2022c. [TAPEX: table pre-training via learning a neural SQL executor](#). In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.
- Tianyang Liu, Fei Wang, and Muhao Chen. 2024a. [Re-thinking tabular data understanding with large language models](#). In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), NAACL 2024, Mexico City, Mexico, June 16-21, 2024*, pages 450–482. Association for Computational Linguistics.
- Yujian Liu, Jiabao Ji, Tong Yu, Ryan A. Rossi, Sungchul Kim, Handong Zhao, Ritwik Sinha, Yang Zhang, and Shiyu Chang. 2024b. [Augment before you try: Knowledge-enhanced table question answering via table expansion](#). *CoRR*, abs/2401.15555.
- Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegrefe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder,

- Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. [Self-refine: Iterative refinement with self-feedback](#). In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.
- Linyong Nan, Chiachun Hsieh, Ziming Mao, Xi Victoria Lin, Neha Verma, Rui Zhang, Wojciech Kryscinski, Hailey Schoelkopf, Riley Kong, Xiangru Tang, Mutethia Mutuma, Ben Rosand, Isabel Trindade, Renusree Bandaru, Jacob Cunningham, Caiming Xiong, and Dragomir R. Radev. 2022. [Fetaqa: Free-form table question answering](#). *Trans. Assoc. Comput. Linguistics*, 10:35–49.
- Suixin Ou and Yongmei Liu. 2022. [Learning to generate programs for table fact verification via structure-aware semantic parsing](#). In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 7624–7638. Association for Computational Linguistics.
- Vaishali Pal, Andrew Yates, Evangelos Kanoulas, and Maarten de Rijke. 2023. [Multitabqa: Generating tabular answers for multi-table question answering](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 6322–6334. Association for Computational Linguistics.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. [Bleu: a method for automatic evaluation of machine translation](#). In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA*, pages 311–318. ACL.
- Panupong Pasupat and Percy Liang. 2015. [Compositional semantic parsing on semi-structured tables](#). In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 1470–1480. The Association for Computer Linguistics.
- Sohan Patnaik, Heril Changwal, Milan Aggarwal, Sumit Bhatia, Yaman Kumar, and Balaji Krishnamurthy. 2024. [CABINET: content relevance-based noise reduction for table question answering](#). In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.
- Zhenting Qi, Mingyuan Ma, Jiahang Xu, Li Lyna Zhang, Fan Yang, and Mao Yang. 2024. [Mutual reasoning makes smaller llms stronger problem-solvers](#). *CoRR*, abs/2408.06195.
- Soumajyoti Sarkar and Leonard Lausen. 2023. [Testing the limits of unified sequence to sequence LLM pretraining on diverse table data tasks](#). *CoRR*, abs/2310.00789.
- Noah Shinn, Beck Labash, and Ashwin Gopinath. 2023. [Reflexion: an autonomous agent with dynamic memory and self-reflection](#). *CoRR*, abs/2303.11366.
- Roe Shraga, Haggai Roitman, Guy Feigenblat, and Mustafa Canim. 2020. [Web table retrieval using multimodal deep learning](#). In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pages 1399–1408. ACM.
- Yuan Sui, Jiaru Zou, Mengyu Zhou, Xinyi He, Lun Du, Shi Han, and Dongmei Zhang. 2024. [TAP4LLM: table provider on sampling, augmenting, and packing semi-structured data for large language model reasoning](#). In *Findings of the Association for Computational Linguistics: EMNLP 2024, Miami, Florida, USA, November 12-16, 2024*, pages 10306–10323. Association for Computational Linguistics.
- Bing Wang, Yan Gao, Zhoujun Li, and Jian-Guang Lou. 2023. [Know what i don’t know: Handling ambiguous and unknown questions for text-to-sql](#). In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 5701–5714. Association for Computational Linguistics.
- Fei Wang, Kexuan Sun, Muhao Chen, Jay Pujara, and Pedro A. Szekely. 2021. [Retrieving complex tables with multi-granular graph representation learning](#). In *SIGIR ’21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, pages 1472–1482. ACM.
- Tong Wang, Chi Jin, Yongkang Chen, Huan Deng, Xiaohui Kuang, and Gang Zhao. 2026. Datafactory: Collaborative multi-agent framework for advanced table question answering. *Information Processing & Management*, 63(6):104723.
- Zilong Wang, Hao Zhang, Chun-Liang Li, Julian Martin Eisenschlos, Vincent Perot, Zifeng Wang, Lesly Miculicich, Yasuhisa Fujii, Jingbo Shang, Chen-Yu Lee, and Tomas Pfister. 2024. [Chain-of-table: Evolving tables in the reasoning chain for table understanding](#). In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.
- Jian Wu, Yicheng Xu, Yan Gao, Jian-Guang Lou, Börje F. Karlsson, and Manabu Okumura. 2023. [TACR: A table-alignment-based cell-selection and reasoning model for hybrid question-answering](#). *CoRR*, abs/2305.14682.
- Renqiu Xia, Bo Zhang, Haoyang Peng, Ning Liao, Peng Ye, Botian Shi, Junchi Yan, and Yu Qiao. 2023. [Structchart: Perception, structuring, reasoning for visual chart understanding](#). *CoRR*, abs/2309.11268.

- Bin Xiao, Burak Kantarci, Jiawen Kang, Dusit Niyato, and Mohsen Guizani. 2025. [Efficient prompting for llm-based generative internet of things](#). *IEEE Internet Things J.*, 12(1):778–791.
- Tianbao Xie, Chen Henry Wu, Peng Shi, Ruiqi Zhong, Torsten Scholak, Michihiro Yasunaga, Chien-Sheng Wu, Ming Zhong, Pengcheng Yin, Sida I. Wang, Victor Zhong, Bailin Wang, Chengzu Li, Connor Boyle, Ansong Ni, Ziyu Yao, Dragomir Radev, Caiming Xiong, Lingpeng Kong, and 4 others. 2022. [Unified-skg: Unifying and multi-tasking structured knowledge grounding with text-to-text language models](#). In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 602–631. Association for Computational Linguistics.
- Zhen Yang, Ziwei Du, Minghan Zhang, Wei Du, Jie Chen, Zhen Duan, and Shu Zhao. 2025. [Triples as the key: Structuring makes decomposition and verification easier in llm-based tableqa](#). In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.
- Yunhu Ye, Binyuan Hui, Min Yang, Binhua Li, Fei Huang, and Yongbin Li. 2023. [Large language models are versatile decomposers: Decomposing evidence and questions for table-based reasoning](#). In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*, pages 174–184. ACM.
- Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. 2020. Tabert: Pretraining for joint understanding of textual and tabular data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8413–8426.
- Peiying Yu, Guoxin Chen, and Jingjing Wang. 2025a. Table-critic: A multi-agent framework for collaborative criticism and refinement in table reasoning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers): ACL*, pages 17432–17451.
- Yangyang Yu, Haohang Li, Zhi Chen, Yuechen Jiang, Yang Li, Jordan W. Suchow, Denghui Zhang, and Khaldoun Khashanah. 2025b. [Finmem: A performance-enhanced LLM trading agent with layered memory and character design](#). *IEEE Trans. Big Data*, 11(6):3443–3459.
- Ziqiang Yuan, Kaiyuan Wang, Shoutai Zhu, Ye Yuan, Jingya Zhou, Yanlin Zhu, and Wenqi Wei. 2025. [Finllms: A framework for financial reasoning dataset generation with large language models](#). *IEEE Trans. Big Data*, 11(5):2264–2277.
- Le Zhang, Yihong Wu, Fengran Mo, Jian-Yun Nie, and Aishwarya Agrawal. 2023. [Moqagpt : Zero-shot multi-modal open-domain question answering with large language model](#). In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pages 1195–1210. Association for Computational Linguistics.
- Siyue Zhang, Anh Tuan Luu, and Chen Zhao. 2024a. [Syntqa: Synergistic table-based question answering via mixture of text-to-sql and E2E TQA](#). In *Findings of the Association for Computational Linguistics: EMNLP 2024, Miami, Florida, USA, November 12-16, 2024*, pages 2352–2364. Association for Computational Linguistics.
- Yunjia Zhang, Jordan Henkel, Avriela Floratou, Joyce Cahoon, Shaleen Deep, and Jignesh M. Patel. 2024b. [Reactable: Enhancing react for table question answering](#). *Proc. VLDB Endow.*, 17(8):1981–1994.
- Wenting Zhao, Ye Liu, Yao Wan, Yibo Wang, Zhongfen Deng, and Philip S. Yu. 2023a. [Localize, retrieve and fuse: A generalized framework for free-form question answering over tables](#). In *Findings of the Association for Computational Linguistics: IJCNLP-AACL 2023 - Findings, Nusa Dua, Bali, November 1-4, 2023*, pages 1–12. Association for Computational Linguistics.
- Yilun Zhao, Linyong Nan, Zhenting Qi, Rui Zhang, and Dragomir Radev. 2022. [Reastap: Injecting table reasoning skills during pre-training via synthetic reasoning examples](#). In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 9006–9018. Association for Computational Linguistics.
- Yilun Zhao, Haowei Zhang, Shengyun Si, Linyong Nan, Xiangru Tang, and Arman Cohan. 2023b. Investigating table-to-text generation capabilities of llms in real-world information seeking scenarios. *arXiv preprint arXiv:2305.14987*.
- Victor Zhong, Caiming Xiong, and Richard Socher. 2017. [Seq2sql: Generating structured queries from natural language using reinforcement learning](#). *CoRR*, abs/1709.00103.
- Fan Zhou, Mengkang Hu, Haoyu Dong, Zhoujun Cheng, Fan Cheng, Shi Han, and Dongmei Zhang. 2022a. [Tacube: Pre-computing data cubes for answering numerical-reasoning questions over tabular data](#). In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 2278–2291. Association for Computational Linguistics.
- Yuxuan Zhou, Xien Liu, Kaiyin Zhou, and Ji Wu. 2022b. [Table-based fact verification with self-adaptive mixture of experts](#). In *Findings of the Association for Computational Linguistics: ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 139–149. Association for Computational Linguistics.

A Algorithm

Algorithm 1 SIP: Logic-Aligned Joint Reasoning

Require: Question Q , Table T , DP iterations n ,
PyAgent iterations m

Ensure: Final answer A_{final}

Generate SQL Logic $\mathcal{L}(q)$ using Eq. (1)–(3)

Generate sub-questions $S = \{S_1, \dots, S_k\}$ using
Eq. (4)

for $i = 1$ to n **do**

 Compute and validate DP outputs using
 Eq. (7) and Eq. (10)

end for

for $i = 1$ to m **do**

 Compute and validate PyAgent outputs using
 Eq. (8)–(9) and Eq. (10)

end for

Compute A_{final} using Eq. (11)

return A_{final}

B Data contamination

Data contamination. Table 11 compares SIP with direct QA prompting. Direct QA achieves only 48.70%, while SIP reaches 75.35%, suggesting that SIP’s gains are unlikely to come from memorization alone. Instead, SQL Logic guides the model to derive answers from the given table context.

Table 11: Comparison of direct QA for data contamination.

Models	Accuracy(%)
Direct QA (Cheng et al., 2023)	48.70
SIP	75.35

C Error Analysis

Error Analysis under Explainability. SIP exposes sub-questions, reasoning traces, and validation results, enabling fine-grained error analysis. We sample 50 erroneous cases from DP and PyAgent, respectively. DP errors mainly involve missing or incorrect records, misunderstanding special rows, counting mistakes, and answer-format mismatches. PyAgent errors mainly come from incorrect function generation, execution failures, and special-row handling. These findings suggest that future improvements should target record selection, special-row recognition, and more reliable function generation.