MTabVQA: Evaluating Multi-Tabular Reasoning of Language Models in Visual Space

Anonymous ACL submission

Abstract

Vision-Language Models (VLMs) have demonstrated remarkable capabilities in interpreting visual layouts and text. However, a significant challenge remains in their ability to interpret robustly and reason over multi-tabular data presented as images, a common occurrence in real-world scenarios like web pages and digital documents. Existing benchmarks typically address single tables or non-visual data (text/structured). This leaves a critical gap: they don't assess the ability to parse diverse table images, correlate information across them, and perform multi-hop reasoning on the combined visual data. We introduce MTabVQA, a novel benchmark specifically designed for multi-tabular visual question answering to bridge that gap. MTabVQA comprises 3,745 complex question-answer pairs that necessitate multi-hop reasoning across several visually rendered table images. We provide extensive benchmark results for state-of-the-art VLMs on MTabVQA, revealing significant performance limitations. We further investigate post-training techniques to enhance these reasoning abilities and release MTabVQA-Instruct, a large-scale instruction-tuning dataset. Our experiments show that fine-tuning VLMs with MTabVQA-Instruct substantially improves their performance on visual multi-tabular reasoning. Code and dataset are available online[1].

1 Introduction

In recent years, vision language models (VLMs) and multimodal systems have demonstrated remarkable capabilities in interpreting complex visual layouts and text (Luo et al., 2024), enabling tasks ranging from document understanding (Zhang et al., 2024a), visual information extraction (Cao et al., 2023), and structured data QA (Antol et al., 2015) to interactive processes like autonomous web navigation (He et al., 2024; Zheng et al., 2024a).

[1] MTabVQA-EMNLP

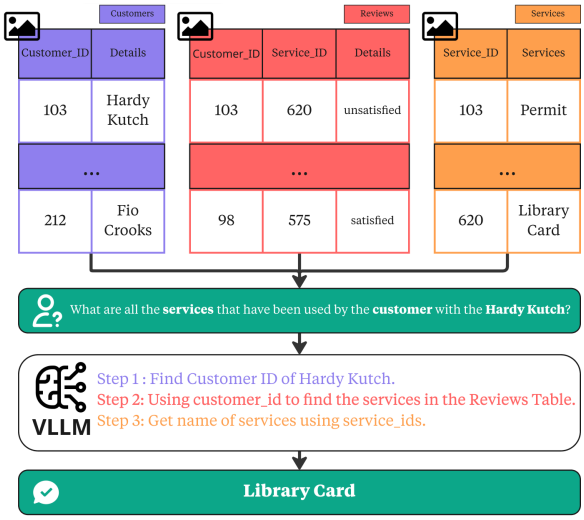


Figure 1: MTabVQA Benchmark, illustrative example showing three tables (Customers, Reviews, Services), a question requiring multi-table reasoning, the reasoning steps involved, and the final answer derived by a vision-language model.

Yet, as these models evolve into sophisticated visual agents capable of browsing screen data and performing agentic tasks, a new challenge has emerged: the robust interpretation and reasoning over multi-tabular data presented as images (Deng et al., 2024; Zheng et al., 2024b). This challenge is particularly relevant in real-world scenarios, where tables often appear as images on web pages, PDFs, or digital documents, and extracting actionable insights may require the agent to reference multiple tables simultaneously.

Traditional benchmarks (Yu et al., 2018; Chen et al., 2020; Zhong et al., 2017) in table understanding and question answering have primarily focused on single-table scenarios, often relying on textual or HTML representations (Zhu et al., 2021; Sui et al., 2024). However, such benchmarks are unable to evaluate model performance on visually complex, multi-tabular data, which requires interpreting layout and structure beyond simple text or HTML. In many practical applications, such as

financial analysis, e-commerce, and scientific research (Lautert et al., 2013), key information is distributed across several tables, each with distinct layouts and visual structures. Current benchmarks (Wu et al., 2024; Pal et al., 2023; Wu et al., 2025; Li et al., 2024c), rooted in single-table, non-visual formats (like text/HTML or relational databases), fail to assess critical capabilities: (1) understanding diverse visual table layouts presented as images, (2) parsing and correlating information across multiple, physically separate tables, and (3) executing multi-hop reasoning grounded in visual data.

To bridge this gap, we propose **M**ulti-**Tab**ular **V**isual **Q**uestion **A**nswering (**MTabVQA**), a novel benchmark specifically designed for assessing the visual reasoning capabilities of models on multi-tabular data represented as images. Distinct from prior benchmarks that primarily focus on single tables (Pasupat and Liang, 2015; Zhong et al., 2017; Zheng et al., 2024b) or utilize non-visual (textual, structured) formats for multi-table reasoning (Wu et al., 2025; Yu et al., 2018; Li et al., 2023a), MTabVQA uniquely evaluates the integration of information across multiple tables. Our benchmark, comprising **3,745** question-answer pairs, challenges models with complex queries across **14 distinct reasoning categories**. These queries are designed to necessitate multi-hop reasoning (e.g., involving aggregation, comparison, or fact-checking) by integrating information from **two to five** visually table images. MTabVQA enables a targeted evaluation of how well current models handle the process of extracting information from multiple table images and performing the multi-hop reasoning necessary to synthesize answers. Our main contributions are:

- We introduce **MTabVQA**, a novel benchmark designed to evaluate multi-hop reasoning over multiple tables presented as **images**, addressing a key gap in existing table QA benchmarks.
- We provide **extensive benchmark results** for SOTA open-source and proprietary VLMs on MTabVQA, revealing significant challenges posed by this task.
- We release **MTabVQA-Instruct**, a instruction-tuning dataset. To demonstrate its effectiveness, we introduce **TableVision**, a VLM fine-tuned on MTabVQA-Instruct, which shows significant improvements on visual multi-tabular reasoning.

2 Related Work

Research in table understanding (Wu et al., 2024) and multimodal reasoning (Zheng et al., 2024b) has advanced significantly. Initial efforts often centered on converting tables into text-based representations like Markdown or HTML (Li et al., 2024b; Zhang et al., 2023), allowing traditional language models to process them. While effective in controlled environments, this approach encounters limitations in real-world settings where tables frequently appear only as images within documents or web interfaces. Processing visually rendered tables through multi-stage text-conversion pipelines (Nassar et al., 2022) presents inherent limitations, they are complex and susceptible to OCR errors, often discard essential visual layout cues (e.g., merged cells, alignment), and risk compounding inaccuracies across stages. This highlights a critical need for models capable of interpreting and reasoning over tables directly from pixel data. Moreover, most systems rely on OCR combined with LLMs, which makes them more error-prone compared to developing a single unified model. Our work focuses squarely on the challenge of extracting information and performing reasoning directly from visual table data, addressing the complexities inherent in image-based table structures.

2.1 Table Understanding and Extraction

Effective reasoning over visual tables fundamentally relies on accurate underlying table understanding, including tasks like detection, segmentation, and structure interpretation (Bonfitto et al., 2021). These foundational challenges were often addressed by specialized methods leveraging object detection and OCR, exemplified by systems like TableFormer (Nassar et al., 2022), which improved the extraction of cell structures from images. Despite these advances, such methods frequently encountered difficulties with complex visual layouts and the semantic alignment crucial for interpreting elements like multirow headers or merged cells. Although recent large-scale datasets like MMTab (Zheng et al., 2024b) have significantly advanced benchmarking for table extraction and understanding from table images, they primarily focus on single-table scenarios. The challenge of integrating information and reasoning across multiple visually presented tables, which MTabVQA addresses, remains less explored.

Benchmark	Question Format	# Tables/Databases	# QA Pairs	Task	Modality
WTQ (Pasupat and Liang, 2015)	NL Questions	2,108	22,033	Single-table QA	Text
SQA (Iyyer et al., 2017)	NL Questions	N/A	17,553	Single-Table QA	Text
WikiSQL (Zhong et al., 2017)	SQL Query	24,241	80,000+	Single-table QA	Text
Spider (Yu et al., 2018)	NL Questions & SQL Query	200	10,181	Text-to-SQL	Text
HybridQA (Chen et al., 2020)	NL Questions	13,000	70k	Table-text QA	Text
FeTaQA (Nan et al., 2022)	NL Questions	10,330	10k	Single tables	Text
BIRD (Li et al., 2023a)	NL Questions & SQL Query	95	12,751	Text-to-SQL	Text
TableBench (Wu et al., 2024)	NL Questions	3,681	886	Single Table	Text
SPINACH (Liu et al., 2024b)	NL Questions & SQL Query	N/A	320	Text-to-SQL	Text
MMQA (Wu et al., 2025)	NL Questions & SQL Query	3,312	3,312	Text-to-SQL, Multi-table QA	Text
MMTab (Zheng et al., 2024b)	NL Questions	23K	49K	Single-Table QA	Images
MTabVQA (ours)	NL Questions	8499	3,745	Multi-Table QA	Images

Table 1: Differences between our MTabVQA and previous table QA benchmarks. We here abbreviate NL = Natural Language and SQL = Structured Query Language.

2.2 Multimodal Question Answering

In parallel, multimodal question answering has made significant progress with models like LLaVA (Liu et al., 2024a), BLIP-2 (Li et al., 2023b), and GPT-4.1[2] demonstrating strong capabilities on image-based tasks. While many of these models excel in general visual understanding, they typically treat tabular content as static images, lacking the ability to navigate or reason across multiple tables. Prior benchmarks, such as WikiTableQuestions (Pasupat and Liang, 2015) and WikiSQL (Zhong et al., 2017), focus on single-table scenarios and text-based table representations. MMQA (Wu et al., 2025), a recent advancement in this area, extends the evaluation framework to multi-table and multi-hop reasoning. However, MMQA relies on textual inputs rather than raw images.

2.3 Multi-Tabular Reasoning

Reasoning across multiple tables demands correlating information from potentially disparate structures via multi-hop operations, a known challenge for current models (Pal et al., 2023). While prior work explored multi-table QA (Pal et al., 2023), summarization (Zhang et al., 2024b), and text-to-SQL (Wu et al., 2025), these efforts predominantly relied on textual or structured data representations. They often bypassed the complexities of interpreting combined visual table layouts, a critical requirement for agents interacting with screen data. MTabVQA directly addresses this research gap by focusing on **multi-tabular visual reasoning**. As in Table 1, prominent prior benchmarks like WTQ (Pasupat and Liang, 2015), WikiSQL (Zhong et al., 2017), and even multi-table focused ones such as Spider (Yu et al., 2018) and MMQA (Wu et al., 2025), primarily operate on textual or structured (e.g., SQL) representations of tables. While MMTab (Zheng et al., 2024b) introduced image-based tables, its focus remained on single-table scenarios. In contrast, MTabVQA specifically requires models to answer complex, multi-hop questions by integrating information presented across multiple table images. This necessitates visual parsing of diverse table layouts from images, a capability not comprehensively evaluated by existing benchmarks that are either non-visual or single-table centric. Thus, MTabVQA's unique combination of multi-table reasoning and image-based input directly targets this underexplored area.

3 MTabVQA Dataset

We introduce **M**ulti-**Tab**ular **V**isual **Q**uestion **A**nswering (**MTabVQA**), a benchmark specifically designed to assess the capacity of multimodal models to perform multi-hop reasoning across multiple tables presented as images. MTabVQA dataset includes four sub-datasets based on the primary source databases from which the underlying table data was derived, as detailed in Table 2. Figure 2 illustrates the multi-stage process used to construct MTabVQA, encompassing data sourcing, relational analysis, controlled data sampling, image rendering, question-answer pair generation, and rigorous verification.

3.1 Tabular Data Collection

MTabVQA utilizes tabular data from diverse sources: BIRD (Li et al., 2023a), Spider (Yu et al.,

[2] GPT 4.1

Dataset Split	Source	Sub-dataset	#QA Pairs	#Tables	Proportion (%)
MTabVQA-Eval	QFMTS (Zhang et al., 2024b)	MTabVQA-Query	2456	5541	65.7%
	Spider (Yu et al., 2018)	MTabVQA-Spider	1048	2363	27.9%
	Atis (Dahl et al., 1994)	MTabVQA-Atis	112	429	3.0%
	MiMoTable (Li et al., 2024c)	MTabVQA-Mimo	129	166	3.4%
	Total Eval Set		**3745**	**8499**	100.0%
MTabVQA-Instruct	MultiTabQA (Pal et al., 2023)	–	10,990	21,976	69.3%
	Spider (Yu et al., 2018)	–	2395	5845	15.2%
	BIRD (Li et al., 2023a)	–	1572	3144	9.9%
	Atis (Dahl et al., 1994)	–	384	1780	2.4%
	MiMoTable (Li et al., 2024c)	–	512	719	3.2%
	Full Instruct Set		**15853**	**33464**	100.0%

Table 2: Detailed composition of the MTabVQA-Eval and MTabVQA-Instruct datasets. The table shows the original data sources and provides statistics for each sub-dataset, including the number of QA pairs and unique tables.

2018), MiMoTable (Li et al., 2024c), QFMTS (Zhang et al., 2024b), and ATIS (Dahl et al., 1994). We prioritized text-to-SQL datasets as their associated complex SQL queries often involve multi-table joins, naturally lending themselves to multi-table reasoning tasks.

To ensure our benchmark targets multi-table reasoning, we first identified relevant database subsets (Figure 2, Step 1). We parsed SQL queries from the source datasets, specifically selecting those requiring multi-table join operations. This analysis confirmed rich inter-table dependencies suitable for our task. Based on this query analysis, we extracted data instances for the MTabVQA-Eval split: 1,048 multi-join queries from Spider (Yu et al., 2018) forming MTabVQA-Spider, 2,578 multi-table instances from QFMTS (Zhang et al., 2024b), and 112 and 129 multi-table pairs from ATIS (Dahl et al., 1994) and MiMoTable (Li et al., 2024c), respectively. The large and complex BIRD (Li et al., 2023a) dataset, over 7,200 join queries across 69 databases, was primarily used to generate the MTabVQA-Instruct dataset. This query-driven selection ensures that the underlying data inherently necessitates multi-table reasoning.

3.2 Data Extraction and Preprocessing

Following the identification of relevant database subsets (Section 3.1), we employed a pipeline to process the data. For each subset, the pipeline extracted the database schemata, including table definitions, column types, primary keys, and foreign key relationships defining inter-table links, and converted the relational data from its native storage (e.g., SQLite) into a standardized JSON format. Recognizing that full database tables can be excessively large for visual rendering and efficient model processing, we implemented a controlled sampling strategy. Tables exceeding a predefined row threshold ($N_{max} = 50$) were sampled down. While the proportion of excluded data varied depending on the original table sizes in each source dataset, this threshold aimed to balance visual complexity and data representativeness across the benchmark.

To preserve crucial relational information between multiple tables during sampling, we utilized a graph-based approach detailed in Algorithm 1 (Appendix A). This method ensures referential integrity by preferentially sampling rows linked across related tables via foreign keys, focusing on connections relevant to the multi-table queries identified earlier. The final output for each instance consists of the sampled table data and corresponding schemata, serialized into JSON.

3.3 Visual Table Rendering

To ensure MTabVQA evaluates visual reasoning over image-based inputs, the sampled tabular data for each QA pair was rendered into images. This step forces models to interpret visual layouts over structured text. We utilized a rendering pipeline employing dataframe_image[3] (with `selenium` or `matplotlib` backends) and custom `Pillow` scripts. This process introduced significant visual diversity by systematically varying structural aspects (e.g., column/row dimensions, relative table positioning) and appearance features (e.g., color schemes, typography, grid styles) across 10 distinct, randomly applied styling themes. This approach simulates the varied appearances of tables in real-world documents and web pages. Further details on the specific themes are provided in Appendix D.

[3] dexplo/dataframe_image

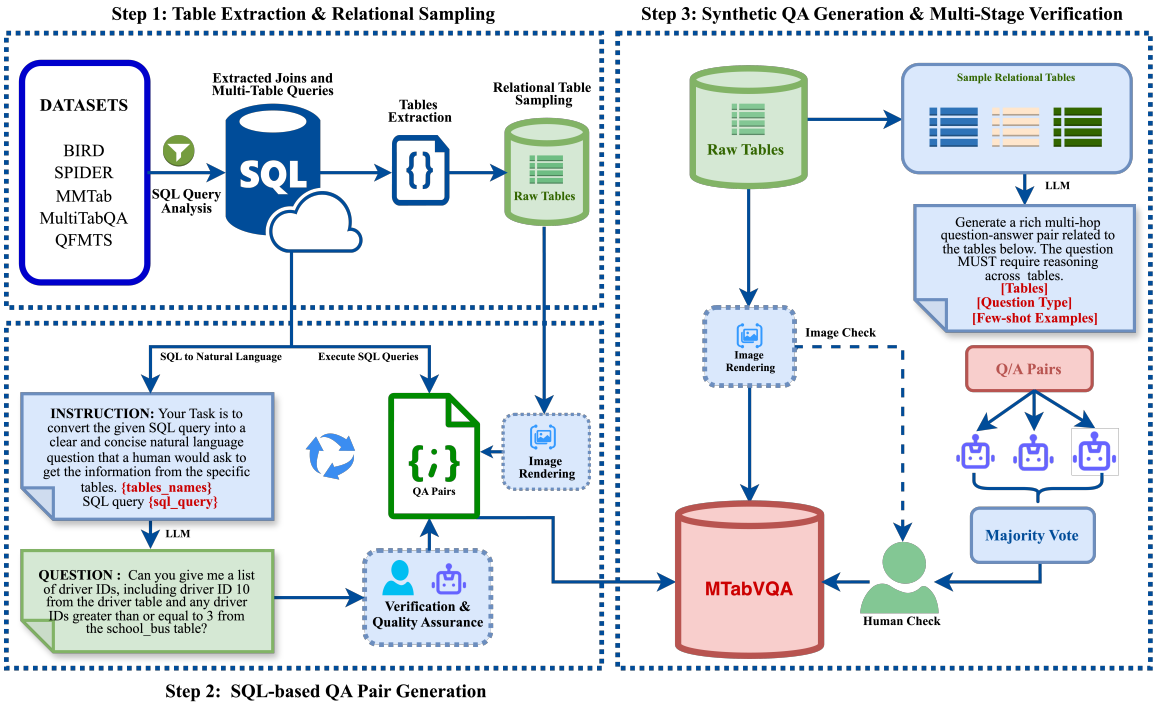


Figure 2: MTabVQA Construction Framework Overview. (1) Data Sourcing & Sampling: Identify multi-table relational data via SQL joins, extract tables, apply relational sampling. (2) Visual QA Generation: Generate multi-hop QA pairs via SQL-to-question conversion or LLM-guided generation from sampled tables/taxonomy; render tables as images. (3) Verification & Finalization: Apply automated (LLM) and human verification for quality and multi-table necessity.

3.4 Multi-Hop QA Pair Generation

The pairs of our dataset are designed for multi-hop reasoning across table images, generated via two strategies (Figure 2, Steps 2-3):

1. SQL-to-Question (Step 2): We converted complex, multi-table SQL queries (from Sec 3.1) into natural language questions. For each SQL query, we executed it on sampled table subsets (S_A, S_B) for a ground-truth answer. An LLM[4] then paraphrased the SQL (given schemas and instructions; Figure 2, bottom-left prompt) into a question, creating QA pairs grounded in verifiable SQL logic.

2. Taxonomy-Guided Generation (Step 3): To diversify reasoning types, an LLM generated novel QA pairs from sampled table subsets and a predefined question taxonomy. This taxonomy, adapted from (Wu et al., 2024) to cover common multi-table reasoning patterns (e.g., multi-hop fact-checking, aggregation), guided the LLM (with few-shot examples; Figure 2, upper-right prompt) to create questions requiring data from $\geq$2 tables, plus answers and reasoning steps in structured JSON. Figure 3 shows the distribution of the question categories, showing that most of the questions are fact-checking, analysis, aggregation, or ranking.

[4]Gemini-2.0-Flash

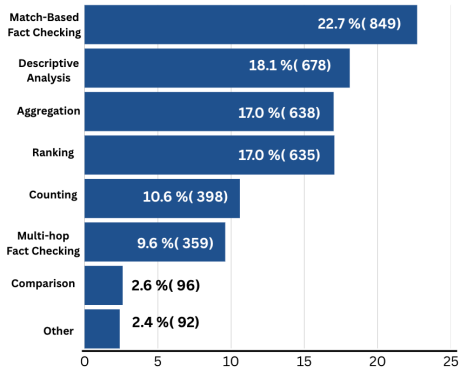


Figure 3: Distribution of Verified Question Categories in the MTabVQA dataset, "Other" includes categories like Anomaly Detection, Arithmetic Calculation, and Multi-hop Numerical Reasoning (total 3,745 QA Pairs).

3.5 Verification and Filtering

To ensure QA quality and multi-table focus, our verification process (Figure 2, Step 3) was done by automated assessment from three LLM agents[4], guided by a verification prompt (Appendix C). These agents evaluated question validity, multi-hop needs, answer accuracy, reasoning soundness, and multi-table necessity ($\geq$2 tables). LLM outputs (JSON with scores/flags) were aggregated by majority vote.

Model	MTabVQA-Spider				MTabVQA-Query				MTabVQA-ATIS				MTabVQA-MiMo				Overall	
	EM	F1	P	R	EM	F1	P	R	EM	F1	P	R	EM	F1	P	R	EM	F1
Open-Source VLMs (Zero-Shot)																		
Gemma-3-12B-IT	15.6	48.0	48.2	53.4	10.3	38.1	39.4	42.6	11.6	35.1	34.2	40.8	9.3	18.6	22.0	18.8	**11.8**	**40.1**
Qwen2.5-VL-7B	8.0	39.8	40.4	44.0	7.8	33.9	34.8	38.0	6.3	32.6	29.0	48.6	9.3	22.2	25.9	22.8	7.8	35.1
InternVL3-8B-Instruct	6.1	32.4	33.0	39.1	5.2	24.8	26.9	29.6	3.6	20.3	19.5	31.9	7.0	19.1	22.3	21.3	5.4	26.6
Phi-3.5-Vision-Instruct	2.9	26.1	25.9	39.6	2.4	22.0	22.3	34.7	1.8	15.0	15.3	24.8	0.8	3.2	3.6	3.3	2.5	22.3
LLaVA-OV-Qwen2-7B	2.2	20.0	19.5	29.3	2.3	15.7	15.9	23.6	0.0	9.2	5.9	33.8	0.7	5.5	4.3	19.1	2.1	18.4
Proprietary VLMs (Zero-Shot)																		
GPT-4.1	49.0	74.3	74.7	76.6	34.2	58.5	59.2	60.8	6.3	39.9	30.0	86.3	20.2	39.6	44.9	38.8	**37.0**	**61.7**
Gemini-2.0-Flash	42.9	68.5	69.2	71.2	31.4	57.3	58.2	60.5	22.3	36.0	37.2	37.5	24.0	42.3	49.2	41.2	34.1	59.3
Fine-tuned Model (Ours)																		
TableVision (Ours)	32.4	64.3	66.6	66.1	49.8	72.6	74.0	73.5	33.0	45.9	48.4	47.8	20.1	36.2	40.8	36.4	**43.4**	**68.2**

Table 3: Performance Comparison of VLMs on MTabVQA-Eval Splits (%), and Overall EM/F1 (%). Models categorized and sorted by overall F1 score within categories. Overall scores are weighted averages. Best overall and best open-source zero-shot overall scores are bolded. EM denotes Exact Match, P Precision, and R Recall.

Pairs meeting criteria (majority valid, confirmed multi-table use, average score $\geq$7.0) advanced to human verification using a Streamlit app (Appendix 6) for final checks on correctness, especially for complex cases. Human Validation was conducted by one annotator. Only pairs passing both automated and human checks were integrated into MTabVQA. This LLM-assisted human oversight yielded a high-quality benchmark by filtering invalid tables or incorrect QA pairs. The resulting verified data formed two entirely disjoint splits, ensuring no overlap between training and evaluation:

- **MTabVQA-Eval:** 3,745 QA pairs for benchmarking VLM performance.
- **MTabVQA-Instruct:** 15,853 instruction-following examples for post-training VLMs.

4 Experiments

This section details the experiments conducted to evaluate VLM capabilities on visual multi-tabular reasoning using our MTabVQA benchmark. Our experiments encompass three key areas:

1. **Benchmarking Current VLMs:** We first establish baseline performance by evaluating leading open-source and proprietary VLMs on the MTabVQA-Eval split and compare it with our fine-tuned model. (Section 4.1).
2. **Evaluating Post-Training Strategies:** We investigate methods to improve VLM performance for multi-table VQA. Using our MTabVQA-Instruct dataset, we explore and compare the effectiveness of different post-training techniques, such as Supervised Fine-Tuning (SFT), Chain-of-Thought (CoT), and Group Relative Policy Optimization (GRPO) (Shao et al., 2024) (Section 4.2).
3. **Investigating Impact of Post-training Data Composition:** We further analyze how VLM performance is affected by the scale and source of the data used for instruction fine-tuning (Section 4.3). Specifically, we fine-tune models on progressively larger and differently sourced subsets of MTabVQA-Instruct and evaluate their generalization on MTabVQA-Eval (Section 4.3).

4.1 Benchmarking

We conducted a comprehensive benchmarking study on MTabVQA-Eval to establish baselines for multi-table visual reasoning. We evaluated leading proprietary VLMs (GPT-4.1[2], Gemini[5] and prominent open-source alternatives (Qwen2.5 (Yang et al., 2024), Gemma-3 (Kamath et al., 2025), LLaVA-One-Vision (Li et al., 2024a), InternVL3 (Zhu et al., 2025), Phi-3.5 (Abdin et al., 2024)), alongside our fine-tuned **TableVision** model. We assessed models in a zero-shot setting across all four MTabVQA-Eval splits (Spider, Query, ATIS, and MiMo), instructing them to generate structured JSON (Appendix G.1). Generation parameters were set to a temperature of 1.0 and top-P of 1.0.

Evaluation Metrics. We primarily use EM for its strict correctness assessment, especially suitable for factual answers from tables. To capture semantic similarity and partial correctness, we also report F1 score, precision (P), and recall (R), providing a more nuanced view of answer quality.

The results (Table 3) highlight MTabVQA's difficulty. Open-source VLMs like LLaVA-One-Vision (2.2% EM, 16.7% F1 overall) and Phi-3.5-Vision struggled significantly in zero-shot, with Gemma-3 being the strongest open-source baseline (11.8%

[5] https://aistudio.google.com/

EM, 40.1% F1 overall). Even proprietary models like GPT-4.1 (37.0% EM, 61.7% F1 overall) did not achieve perfect scores and showed performance dips on certain splits (e.g., GPT-4.1 on ATIS scored 6.3% EM), indicating varied challenges within the benchmark, which shows that there is space for improvement.

TableVision, our model fine-tuned using LoRA (rank 128) on MTabVQA-Instruct with Qwen2.5-VL-7B as its base, demonstrated the value of targeted training by achieving the highest overall performance (43.4% EM, 68.2% F1). Notably, TableVision surpassed all other models, including GPT-4.1, on the MTabVQA-Query (49.8% EM, 72.6% F1) and MTabVQA-ATIS splits. This shows that fine-tuning can enable smaller open-source models to outperform larger proprietary systems on complex visual multi-tabular reasoning, underscoring MTabVQA-Instruct's effectiveness.

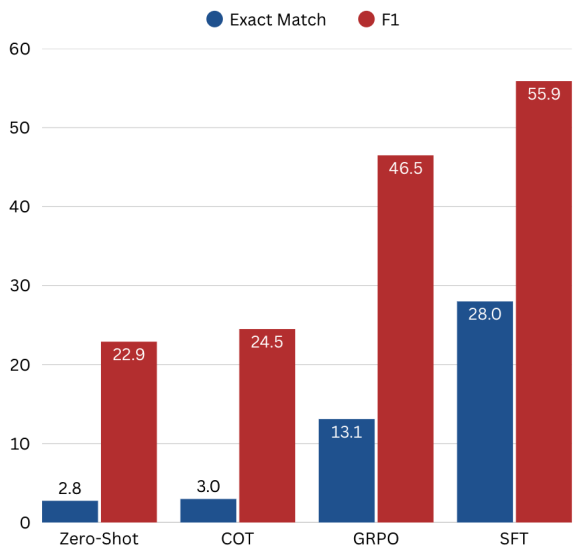


Figure 4: Performance comparison of Qwen2.5-VL-3B on the MTabVQA-Eval using different post-training strategies.

4.2 Post-training VLMs for Multi-Table Visual Reasoning

To explore methods for enhancing VLM performance on visual multi-tabular reasoning, we investigated several post-training techniques using a subset of our MTabVQA-Instruct dataset. Specifically, we utilized 2,395 QA pairs derived from the Spider data source, selected for its demonstrated fine-tuning effectiveness (Section 4.3) and manageable size for these intensive experiments. We selected the Qwen2.5-VL-3B model (Yang et al., 2024) as our base VLM, primarily due to the significant computational requirements associated with advanced post-training methods like reinforcement learning. Our investigation compared the effectiveness of different post-training techniques for multi-tabular visual reasoning. All evaluations were conducted on a corresponding MTabVQA-Eval split.

First, we established a baseline by evaluating the zero-shot performance of the 3B model. Consistent with observations for larger models (Section 4.1), the base 3B model exhibited poor initial performance on this complex multi-hop reasoning task, achieving an EM of 2.8% and an F1 score of 22.9% (Figure 4). We then evaluated the efficacy of using step-by-step reasoning through Chain-of-Thought (CoT) prompting (See Appendix G.2). While this approach encouraged structured responses, it resulted in only marginal improvements, with EM increasing slightly to 3.0% and F1 to 24.5%.

Next, recognizing the reasoning-intensive nature of multi-tabular VQA, we investigated GRPO (Shao et al., 2024), a reinforcement learning-based post-training approach using the selected 2,395-pair MTabVQA-Instruct subset. As shown in Figure 4, GRPO improved performance over the CoT baseline, achieving an EM of 13.1% and an F1 score of 46.5%.

Subsequently, we performed SFT on the same subset. For this, we employed LoRA (Hu et al., 2021) with a rank of 128 for parameter-efficient optimization. SFT yielded substantial performance gains over both CoT and GRPO, boosting EM to 28.0% and F1 to 55.9% (Figure 4). This demonstrates the strong effectiveness of targeted instruction tuning with SFT for this task in our experiments. While GRPO showed improvement, its gains did not surpass SFT with LoRA. We hypothesize that the effectiveness of GRPO in this context might be limited by the challenge of defining a more sophisticated reward function than a simple exact match/F1 score, which could better capture nuanced aspects of visual multi-tabular reasoning.

4.3 Impact of Post-training Data Scale and Source

To understand how instruction-tuning data composition affects performance, we used Qwen2.5-VL-7B as our base VLM. We then fine-tuned it on several MTabVQA-Instruct subsets, each derived from different original data sources (Table 2) to vary both data scale and origin. These models were fine-tuned using Supervised Fine-Tuning (SFT) with

Fine-tuning Subset (Source)	# Samples	MTabVQA-Spider		MTabVQA-Query		MTabVQA-ATIS		MTabVQA-MiMo		Overall	
		EM	F1	EM	F1	EM	F1	EM	F1	EM	F1
Qwen2.5-VL-7B (Zero-Shot)	0	8.0	39.8	7.8	33.9	6.3	32.6	9.3	22.2	7.8	35.1
MiMo+ATIS Subset	896	13.7	45.7	11.5	37.5	35.7	<u>46.5</u>	17.1	<u>39.7</u>	13.0	40.0
Spider Subset	2,395	26.9	59.2	49.8	71.2	13.4	22.5	17.1	31.9	41.5	65.2
MultiTabQA Subset	10,990	10.1	33.2	8.7	28.6	16.1	41.9	11.6	25.5	9.4	30.2
MTabVQA-Instruct (Full)	15,853	32.4	<u>64.3</u>	49.8	<u>72.6</u>	33.0	45.9	20.2	36.2	**43.4**	**68.2**

Table 4: Performance of fine-tuned models on dataset splits of MTabVQA-Instruct measuring the influence of dataset on the overall performance on MTabVQA-Eval. Performance is measured in EM and F1. **Bold** indicates the best overall performance. <u>Underline</u> indicates best performance for each MTabVQA-Eval subset.

LoRA (rank 128) and benchmarked on the full MTabVQA-Eval suite. Table 4 presents the EM and F1 scores across MTabVQA-Eval's sub-splits (Spider, Query, ATIS, MiMo) and overall. The fine-tuning subsets included a combined MiMo+ATIS set (896 examples), a Spider-derived set (2,395 examples), a MultiTabQA-derived set (10,990 examples), and our full MTabVQA-Instruct (15,853 examples).

The fine-tuning experiments, detailed in Table 4, reveal a complex relationship between data scale, source, and model performance. Generally, more fine-tuning data leads to better EM and F1 scores, as seen when comparing the MiMo+ATIS subset (896 examples) to the larger Spider subset (2,395 examples). The model trained on the full MTabVQA-Instruct dataset of 15,853 diverse examples achieved the highest overall F1 score (68.2%), highlighting the benefit of scale when combined with relevant and varied data.

However, the source of the fine-tuning data is critically important. The model trained only on the large MultiTabQA subset exhibited surprisingly low overall performance (30.2% F1), significantly underperforming compared to the model trained on the much smaller Spider subset and even the MiMo+ATIS subset. This suggests that the characteristics of the MultiTabQA data, while extensive, may not align well with the broader MTabVQA-Eval benchmark or could introduce a domain shift. For instance, its F1 score on MTabVQA-Query and MTabVQA-Spider was substantially lower than that achieved by TableVision or the Spider-tuned model. This highlights that a large volume of data from a single, potentially narrowly focused or misaligned source can be less effective than smaller, more targeted, or diverse datasets.

Furthermore, domain-specific alignment proves beneficial. The model fine-tuned on the Spider subset, for example, demonstrated strong performance on the MTabVQA-Spider eval split. The superior overall performance of TableVision, trained on full MTabVQA-Instruct, indicates that data diversity is crucial for generalization across varied multi-table reasoning scenarios. This shows that while scaling instruction data is generally advantageous, the relevance and diversity of this data with the target tasks is important for achieving optimal performance.

5 Conclusion

In this work, we introduce MTabVQA-Eval, a novel and challenging benchmark specifically designed to evaluate the multi-tabular reasoning capabilities of vision-language models over tables presented as images. MTabVQA-Eval, comprising 3,745 QA pairs, focuses on a critical yet underexplored area of integrating and reasoning about information distributed across several table images. This benchmark significantly contributes to bridging the gap between existing table QA benchmarks, which often rely on single or non-visual tables. We evaluated a range of SOTA open-source and proprietary VLMs on MTabVQA-Eval, revealing substantial challenges these models face with visual multi-tabular reasoning. To address this, we also release MTabVQA-Instruct, a large-scale instruction-tuning dataset. Our experiments demonstrate that our fine-tuned model, TableVision on the MTabVQA-Instruct dataset, leads to considerable performance improvements on this task. Despite these advancements, the performance of VLMs on MTabVQA-Eval indicates significant room for growth, underscoring the complexities of robust visual multi-tabular reasoning and highlighting key areas for future research in developing more capable VLMs.

In future work, we plan to explore more programmatically generated or real-world sourced table images exhibiting even greater visual diversity and degradation to more rigorously test VLM visual parsing and grounding capabilities.

Limitations

While MTabVQA represents a significant step towards evaluating visual multi-tabular reasoning, we acknowledge several limitations.

English-Only. The current iteration of MTabVQA is primarily English-centric. Its underlying tabular data, generated questions, and answers are predominantly in English, which limits the benchmark's applicability for evaluating VLMs on multi-tabular reasoning in other languages. Extending MTabVQA to include multilingual tables and queries would be a valuable contribution, allowing for a more comprehensive assessment of VLM capabilities across diverse linguistic contexts and promoting research in multilingual visual document understanding.

Synthetic Table Layout. While MTabVQA tasks require multi-hop reasoning across table images and incorporate varied visual renderings, the scope of this visual complexity could be further expanded. Real-world documents often contain tables with highly unconventional layouts, extensive cell merging/spanning, embedded charts or icons within cells, and varying image quality (e.g., scanned documents with noise), which makes the task even more challenging for LLMs.

Limited Annoation. To verify that the QA pairs were correct, we used only one annotator to verify the judgments of the LLM's agent. Although the annotation was carried out carefully, there may have been minimal errors in the data annotation, as there was no double-checking by two people.

References

Marah I Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat S. Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sébastien Bubeck, Martin Cai, Caio César Teodoro Mendes, Weizhu Chen, Vishrav Chaudhary, Parul Chopra, and 68 others. 2024. Phi-3 Technical Report: A Highly Capable Language Model Locally on Your Phone. *CoRR*, abs/2404.14219.

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. 2015. VQA: Visual Question Answering. *CoRR*, abs/1505.00468.

Sara Bonfitto, Elena Casiraghi, and Marco Mesiti. 2021. Table understanding approaches for extracting knowledge from heterogeneous tables. *WIREs Data Mining Knowl. Discov.*, 11(4).

Panfeng Cao, Ye Wang, Qiang Zhang, and Zaiqiao Meng. 2023. GenKIE: Robust Generative Multimodal Document Key Information Extraction. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 14702–14713. Association for Computational Linguistics.

Wenhu Chen, Hanwen Zha, Zhiyu Chen, Wenhan Xiong, Hong Wang, and William Yang Wang. 2020. HybridQA: A dataset of multi-hop question answering over tabular and textual data. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1026–1036, Online. Association for Computational Linguistics.

Deborah A. Dahl, Madeleine Bates, Michael Brown, William Fisher, Kate Hunicke-Smith, David Pallett, Christine Pao, Alexander Rudnicky, and Elizabeth Shriberg. 1994. Expanding the scope of the ATIS task: The ATIS-3 corpus. In *Human Language Technology: Proceedings of a Workshop held at Plainsboro*.

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.

Hongliang He, Wenlin Yao, Kaixin Ma, Wenhao Yu, Yong Dai, Hongming Zhang, Zhenzhong Lan, and Dong Yu. 2024. WebVoyager: Building an end-to-end web agent with large multimodal models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6864–6890, Bangkok, Thailand. Association for Computational Linguistics.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. 2021. LoRA: Low-Rank Adaptation of Large Language Models. *CoRR*, abs/2106.09685.

Mohit Iyyer, Wen-tau Yih, and Ming-Wei Chang. 2017. Search-based neural structured learning for sequential question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1821–1831, Vancouver, Canada. Association for Computational Linguistics.

Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, Louis Rouillard, Thomas Mesnard, Geoffrey Cideron, Jean-Bastien Grill, Sabela Ramos, Edouard Yvinec, Michelle Casbon, Etienne Pot, Ivo Penchev, Gaël Liu, and 79 others. 2025. Gemma 3 Technical Report. *CoRR*, abs/2503.19786.

Larissa R. Lautert, Marcelo M. Scheidt, and Carina F. Dorneles. 2013. Web table taxonomy and formalization. *SIGMOD Rec.*, 42(3):28–33.

Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Yanwei Li, Ziwei Liu, and Chunyuan Li. 2024a. LLaVA-OneVision: Easy Visual Task Transfer. *CoRR*, abs/2408.03326.

Jinyang Li, Binyuan Hui, Ge Qu, Binhua Li, Jiaxi Yang, Bowen Li, Bailin Wang, Bowen Qin, Rongyu Cao, Ruiying Geng, Nan Huo, Xuanhe Zhou, Chenhao Ma, Guoliang Li, Kevin Chen-Chuan Chang, Fei Huang, Reynold Cheng, and Yongbin Li. 2023a. Can LLM Already Serve as A Database Interface? A BIg Bench for Large-Scale Database Grounded Text-to-SQLs. *CoRR*, abs/2305.03111.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven C. H. Hoi. 2023b. BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models. *CoRR*, abs/2301.12597.

Peng Li, Yeye He, Dror Yashar, Weiwei Cui, Song Ge, Haidong Zhang, Danielle Rifinski Fainman, Dongmei Zhang, and Surajit Chaudhuri. 2024b. Table-GPT: Table Fine-tuned GPT for Diverse Table Tasks. *Proc. ACM Manag. Data*, 2(3).

Zheng Li, Yang Du, Mao Zheng, and Mingyang Song. 2024c. MiMoTable: A Multi-scale Spreadsheet Benchmark with Meta Operations for Table Reasoning. *CoRR*, abs/2412.11711.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. 2024a. Improved Baselines with Visual Instruction Tuning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle.*, pages 26286–26296. IEEE.

Shicheng Liu, Sina Semnani, Harold Triedman, Jialiang Xu, Isaac Dan Zhao, and Monica Lam. 2024b. SPINACH: SPARQL-based information navigation for challenging real-world questions. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 15977–16001, Miami, Florida, USA. Association for Computational Linguistics.

Chuwei Luo, Yufan Shen, Zhaoqing Zhu, Qi Zheng, Zhi Yu, and Cong Yao. 2024. LayoutLLM: Layout Instruction Tuning with Large Language Models for Document Understanding. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15630–15640.

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.

Ahmed S. Nassar, Nikolaos Livathinos, Maksym Lysak, and Peter W. J. Staar. 2022. TableFormer: Table Structure Understanding with Transformers. *CoRR*, abs/2203.01017.

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)*, pages 6322–6334, Toronto, Canada. Association for Computational Linguistics.

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*, pages 1470–1480, Beijing, China. Association for Computational Linguistics.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *CoRR*, abs/2402.03300.

Yuan Sui, Mengyu Zhou, Mingjie Zhou, Shi Han, and Dongmei Zhang. 2024. Table meets llm: Can large language models understand structured table data? a benchmark and empirical study. In *The 17th ACM International Conference on Web Search and Data Mining (WSDM '24) Mérida, Mexico*.

Jian Wu, Linyi Yang, Dongyuan Li, Yuliang Ji, Manabu Okumura, and Yue Zhang. 2025. MMQA: evaluating llms with multi-table multi-hop complex questions. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore*. OpenReview.net.

Xianjie Wu, Jian Yang, Linzheng Chai, Ge Zhang, Jiaheng Liu, Xinrun Du, Di Liang, Daixin Shu, Xianfu Cheng, Tianzhen Sun, Guanglin Niu, Tongliang Li, and Zhoujun Li. 2024. TableBench: A Comprehensive and Complex Benchmark for Table Question Answering. *CoRR*, abs/2408.09174.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 22 others. 2024. Qwen2.5 Technical Report. *CoRR*, abs/2412.15115.

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. 2018. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3911–3921, Brussels, Belgium. Association for Computational Linguistics.

Jiaxin Zhang, Wentao Yang, Songxuan Lai, Zecheng Xie, and Lianwen Jin. 2024a. DocKylin: A Large Multimodal Model for Visual Document Understanding with Efficient Visual Slimming. *CoRR*, abs/2406.19101.

Tianshu Zhang, Xiang Yue, Yifei Li, and Huan Sun. 2023. TableLlama: Towards Open Large Generalist Models for Tables. *CoRR*, abs/2311.09206.

Weijia Zhang, Vaishali Pal, Jia-Hong Huang, E. Kanoulas, and Maarten de Rijke. 2024b. QFMTS: Generating Query-Focused Summaries over Multi-Table Inputs. In *European Conference on Artificial Intelligence*.

Boyuan Zheng, Boyu Gou, Jihyung Kil, Huan Sun, and Yu Su. 2024a. GPT-4V(ision) is a Generalist Web Agent, if Grounded. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria*.

Mingyu Zheng, Xinwei Feng, Qingyi Si, Qiaoqiao She, Zheng Lin, Wenbin Jiang, and Weiping Wang. 2024b. Multimodal Table Understanding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, pages 9102–9124. Association for Computational Linguistics.

Victor Zhong, Caiming Xiong, and Richard Socher. 2017. Seq2SQL: Generating Structured Queries from Natural Language using Reinforcement Learning. *CoRR*, abs/1709.00103.

Fengbin Zhu, Wenqiang Lei, Youcheng Huang, Chao Wang, Shuo Zhang, Jiancheng Lv, Fuli Feng, and Tat-Seng Chua. 2021. TAT-QA: A question answering benchmark on a hybrid of tabular and textual content in finance. 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 3277–3287, Online. Association for Computational Linguistics.

Jinguo Zhu, Weiyun Wang, Zhe Chen, Zhaoyang Liu, Shenglong Ye, Lixin Gu, Hao Tian, Yuchen Duan, Weijie Su, Jie Shao, Zhangwei Gao, Erfei Cui, Xuehui Wang, Yue Cao, Yangzhou Liu, Xingguang Wei, Hongjie Zhang, Haomin Wang, Weiye Xu, and 32 others. 2025. InternVL3: Exploring Advanced Training and Test-Time Recipes for Open-Source Multimodal Models. *Preprint*, arXiv:2504.10479.

A Relational Table Sampling

Algorithm 1 details our method for creating smaller, interconnected samples from large databases. We start by randomly selecting a limited number of rows (up to a maximum, $N_{max} = 50$) from one initial table. Then, using the database's foreign keys, we identify other tables linked to this first one. When sampling from these linked tables, the crucial step is to find and prioritize rows that are directly related to the rows already chosen from the previous table. This is achieved by matching values in the specific columns that link the tables. This process of finding related data and sampling continues as the algorithm explores outwards to other connected tables, ensuring the final set of sampled tables forms a related subset of the original database.

Algorithm 1 Relational Table Sampling

Input:
$\mathcal{D}$: Input database (collection of tables)
$\mathcal{R}$: Set of foreign key relationships between tables in $\mathcal{D}$
N_{max}: Maximum number of rows per sampled table
(V: Set of table identifiers derived from $\mathcal{D}$)
($G = (V, E)$: Relationship graph derived from $\mathcal{D}$ and $\mathcal{R}$)
Output:
$\mathcal{S}$: Set of pairs (t, S_t), where S_t is the sampled row subset for table $t \in V$

$\mathcal{S} \leftarrow \emptyset$; $\mathcal{P} \leftarrow \emptyset$ ▷ $\mathcal{S}$: Output samples, $\mathcal{P}$: Processed tables set
$t_{start} \leftarrow$ SelectSeed(V, G) ▷ Select a starting table (e.g., highest degree)
$S_{t_{start}} \leftarrow$ Sample(t_{start}, N_{max}) ▷ Sample initial rows for t_{start}
$\mathcal{S} \leftarrow \{(t_{start}, S_{t_{start}})\}$; $\mathcal{P} \leftarrow \{t_{start}\}$ ▷ Update output set and processed set
Initialize Q; Q.Enqueue(t_{start}) ▷ Q: Queue for Breadth-First Search (BFS)
while Q is not empty **do** ▷ Perform BFS traversal
    $t_{curr} \leftarrow Q$.Dequeue$()$ ▷ t_{curr}: Current table being processed
    for each $t_{rel} \in$ Neighbors$(t_{curr}, G) \setminus \mathcal{P}$ **do** ▷ t_{rel}: Related, unprocessed neighbor table
        $R_{linked} \leftarrow$ GetLinkedRows$(t_{rel}, t_{curr}, S_{t_{curr}}, \mathcal{R})$ ▷ Get rows in t_{rel} linked to sampled rows
        $S_{t_{rel}} \leftarrow$ SampleSubset(R_{linked}, N_{max}) ▷ Sample a subset from linked rows, max size N_{max}
        $\mathcal{S} \leftarrow \mathcal{S} \cup \{(t_{rel}, S_{t_{rel}})\}$ ▷ Add the new sample to the output
        $\mathcal{P} \leftarrow \mathcal{P} \cup \{t_{rel}\}$; Q.Enqueue(t_{rel}) ▷ Mark t_{rel} as processed and add to queue
    end for
end while
return $\mathcal{S}$ ▷ Return the final set of sampled table subsets

B Data Sourcing: Join and Filter Details

This section provides a detailed breakdown of the process used to identify and filter data instances requiring multi-table join operations from the source datasets, as mentioned in Section 3.1. This formed the basis for constructing both the MTabVQA-Eval and MTabVQA-Instruct splits, ensuring a focus on multi-tabular reasoning. The primary method involved parsing SQL queries associated with text-to-SQL datasets to detect explicit join clauses (e.g., 'JOIN', 'INNER JOIN', 'LEFT JOIN'). For datasets without explicit SQL, we relied on provided metadata or question characteristics indicative of multi-table requirements.

B.1 Spider Dataset

The Spider dataset (Yu et al., 2018) is a large-scale text-to-SQL benchmark. We analyzed its train, development (dev), and test splits to identify questions whose corresponding SQL queries involved joins.

- **Train Split:**
  - Total Questions: 7,000
  - Questions with SQL Joins: 2,771
  - Selected for MTabVQA-Instruct (after filtering and processing): 2,395 instances.
- **Development (Dev) Split:**
  - Total Questions: 1,034
  - Questions with SQL Joins: 408
- **Test Split:**
  - Total Questions: 2,147
  - Questions with SQL Joins: 862
- **MTabVQA-Eval (from Spider Dev/Test):**
  - Combined Join Questions from Dev & Test: 408 (Dev) + 862 (Test) = 1,270
  - Selected for MTabVQA-Eval (MTabVQA-Spider-Eval split): 1,048 instances. These were chosen from the 1,270 join questions based on criteria ensuring clear multi-hop reasoning paths, unambiguous answers from sampled data, and visual representability.

B.2 QFMTS Dataset

The QFMTS dataset (Zhang et al., 2024b) focuses on query-focused multi-document summarization with tables. We identified instances requiring information synthesis across multiple tables.

- Total Questions/Instances: 4,908
- Instances Identified as Requiring Multi-Table Reasoning (e.g., via SQL joins or inherent task nature): 2,578
- Selected for MTabVQA-Eval (MTabVQA-Query-Eval split, primarily from QFMTS): 2,456 instances. Filtering ensured complexity and suitability for our visual QA benchmark.

B.3 BIRD Dataset

BIRD (Li et al., 2023a) is another challenging text-to-SQL benchmark designed to evaluate robustness on large databases and complex queries.

- Total Identified SQL Join Queries (approx.): 7,900
- Generated QA pairs for MTabVQA-Instruct: 1,572 instances. These were generated from a diverse selection of the join queries, focusing on creating complex multi-hop reasoning scenarios suitable for instruction tuning.

B.4 MultiTabQA Dataset

The MultiTabQA dataset (Pal et al., 2023) is specifically designed for question answering over multiple tables.

- Total QA pairs involving joins/multi-table lookups utilized: 10,990
- These were directly incorporated into the MTabVQA-Instruct dataset due to their inherent multi-table nature.

B.5 ATIS Dataset

The Air Travel Information System (ATIS) dataset (Dahl et al., 1994) contains spoken language queries related to flight information, often mapped to relational database queries.

- Total Questions Analyzed: 496
- Instances identified/selected for MTabVQA-Eval (MTabVQA-Atis split): 112
- Instances selected/generated for MTabVQA-Instruct: 384 (See Table 2).

B.6 MiMoTable Dataset

The MiMoTable dataset (Li et al., 2024c) focuses on multimodal table understanding.

- Total Questions/Instances: 1,636
- Questions Identified with Multi-Table Requirements (e.g., from problem descriptions or metadata indicating cross-table information needed): 641
- Selected for MTabVQA-Instruct: 512 instances.
- Selected for MTabVQA-Eval: 129 instances.

B.7 Overall Summary

Across all source datasets, we identified approximately **26,826** potential questions or instances that involved multi-table join operations or inherently required multi-table reasoning. Through our processing, filtering, and generation pipeline, a total of **19,608** high-quality, multi-tabular visual question-answering instances were curated to form the MTabVQA-Eval (3,745 pairs) and MTabVQA-Instruct (15,853 pairs, with some overlap in underlying source tables but disjoint QA pairs) datasets. The filtering criteria included ensuring genuine multi-hop reasoning, clarity of questions and answers, visual representability of the involved tables, and overall quality for benchmarking and instruction tuning.

C Verification Prompt

The following prompt was provided to the verification LLMs-based verification agents during the automated assessment phase described in Section 3.5.

```
You are a verification agent for table-based question answering.
You need to verify if the answer and reasoning for the given
question are correct based ONLY on the provided table data.

[Tables Used]
[Sampled Table Data (JSON Format)]

[Question-Answer Pair]
Question: [Generated Question Text]
Answer: [Generated Answer (JSON Format)]
Reasoning Steps: [Generated Reasoning Steps]
Question Type: [Designated Question Type]

Your task:
1. Check if the question is well-formed and genuinely requires multi-hop reasoning across
MULTIPLE provided tables. Single-table questions are invalid.
2. Verify if the answer is accurate based only on the information present in the given tables.
If the answer is incorrect, 'is_valid' must be 'false'.
3. Check if the 'tables_used' field correctly lists relevant tables and if at least
two tables were necessary.
4. Validate if the reasoning steps are logical, coherent, and correctly lead from the table
data to the answer.

Respond with ONLY a valid JSON object (no introductory text, markdown formatting,
or code blocks outside the JSON structure) containing the following keys:
{{
    "is_valid": true/false,
    "verification_comments": "Your detailed verification comments
                             explaining the validity/issues and
                             multi-table requirement.",
    "score": <an integer score from 0 to 10, where 10 is
              perfect adherence to all criteria>,
    "uses_multiple_tables": true/false
}}
```

Figure 5: LLM prompt for automated QA pair verification. Placeholders like '[Generated Question Text]' represent the actual data provided to the model.

D Visual Table Rendering Details

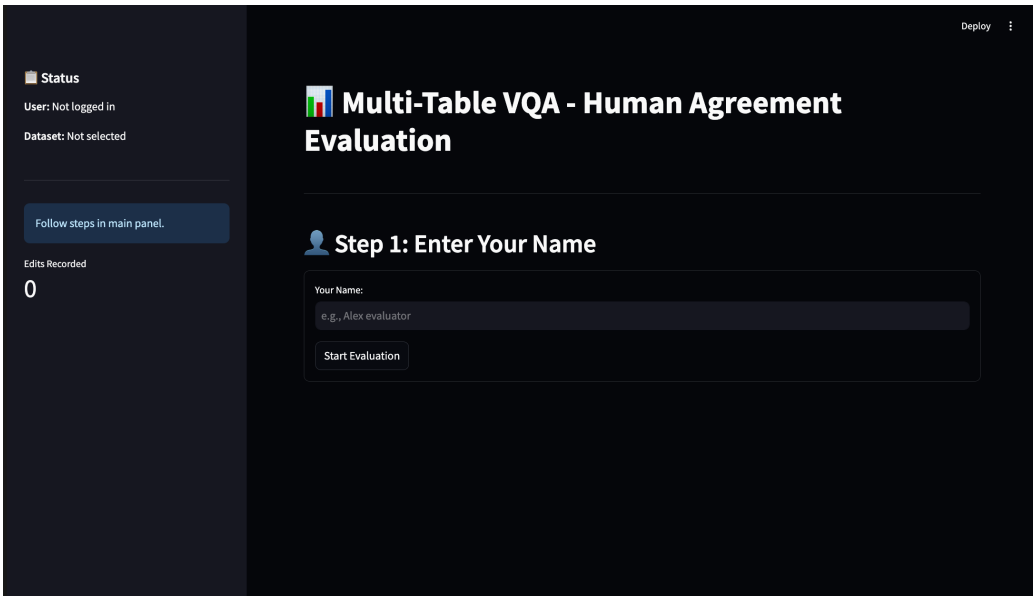
As described in Section 3.3, MTabVQA table images were generated with significant visual diversity to mimic real-world appearances. For each QA pair, the rendering process introduced controlled variations across several dimensions using 10 distinct styling themes, randomly selected per table. These themes systematically varied:

- **Structure and Layout:**
  - Column widths and row heights were adapted to content to ensure readability while introducing natural variations.
  - The relative positioning of multiple table images within the final visual context presented to the model was also varied (e.g., tables rendered side-by-side, stacked vertically, or with other layout configurations).
- **Appearance (Themes, Fonts, Styles):** The 10 distinct styling themes systematically manipulated the following:
  - *Color Schemes:* This included variations in header background colors (e.g., using specific hex codes like `#4CAF50` (green), `#1E88E5` (blue), `#333` (dark grey)), cell background colors, text colors (e.g., white text on dark headers, black text on light backgrounds), and alternating row shading ('zebra striping' with colors like `#f2f2f2`).
  - *Typography:* Different font families (e.g., common serif and sans-serif fonts) were used. Font weights were varied (e.g., bold headers, normal weight for cell content). Font sizes were adjusted within themes (e.g., a base size of **12pt** in one theme, with relative adjustments for headers).
  - *Styling Elements:* The presence, style, and color of grid lines were varied (e.g., solid lines, dashed lines, varying thickness, or minimalist themes with no grid lines). Cell padding was adjusted to control spacing within cells. Border styles for the overall table and individual cells were also diversified (e.g., `1px solid black`, `2px solid #000`, or no borders).

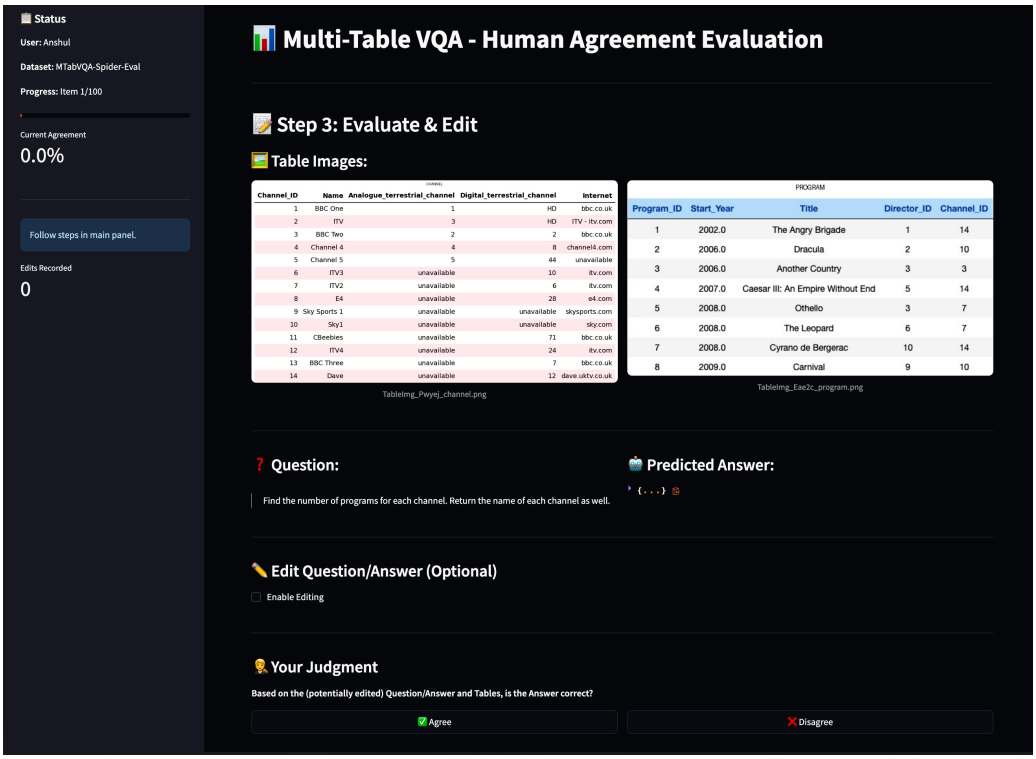
This deliberate introduction of visual diversity is key to challenging models on robust OCR and layout understanding across varied presentations before they engage in multi-tabular reasoning.

E Human Verification Interface

Figure 6 shows the interface of the Streamlit application used for the final human verification stage (Section 3.5). This tool displayed the rendered table images, the generated question, the LLM-generated answer and reasoning, and the automated verification scores, allowing reviewers to make the final acceptance decision.



(a) Initial login screen for evaluator identification.

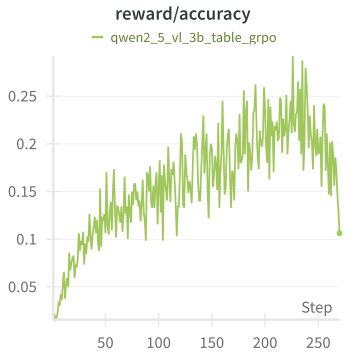


(b) Main evaluation screen displaying table images, question, predicted answer, and reviewer judgment options.

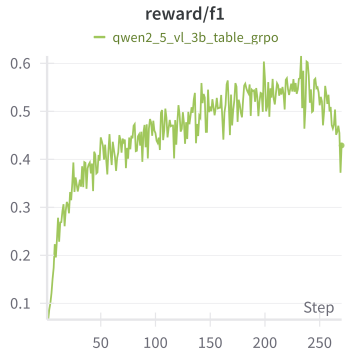
Figure 6: Screenshots of the Streamlit application interface used for human verification. Panel (a) shows the user login step, and panel (b) presents the core evaluation interface with table images and QA details.

F GRPO Training Details

This section provides additional details on the Group Relative Policy Optimization (GRPO) (Shao et al., 2024) experiments discussed in Section 4.2 for fine-tuning the Qwen2.5-VL-3B model. We utilized the EasyR1 framework[6] for these experiments, training for a total of 270 steps. The training was conducted on a subset of MTabVQA-Instruct derived from the Spider dataset (2,395 examples).



(a) Reward: Accuracy (EM)



(b) Reward: F1 Score

(c) Reward: Structural Format

(d) Reward: JSON Format

(e) Reward: Overall

Figure 7: GRPO training reward component curves for Qwen2.5-VL-3B over 270 training steps. These plots illustrate the learning progress for content accuracy (EM, F1), structural format adherence, JSON validity, and the combined overall reward.

Reward Function: The reward function for GRPO was designed to encourage both semantic correctness and proper output formatting. It was a composite score derived from:

- **Content Correctness:** Assessed by the weighted sum of Exact Match (EM) and F1 score between the generated answer and the ground truth.
- **Format Adherence:** This included two components:
  - *Structural Format Score:* A binary score indicating whether the model's output correctly included the required '<think>' and '<answer>' tags.
  - *JSON Format Score:* A binary score indicating whether the content within the '<answer>' tags was valid JSON.

The overall reward signal aimed to maximize these components, guiding the model towards generating accurate and well-formatted responses.

Figure 7 shows the progression of various reward components during the GRPO training process. The plots for 'reward/accuracy' (EM) and 'reward/f1' show a general upward trend, indicating learning of

[6] https://github.com/hiyouga/EasyR1

content correctness. The ‘reward/format‘ and ‘reward/json‘ plots demonstrate that the model quickly learned to adhere to the specified output structure. The ‘reward/overall‘ plot reflects the combined learning signal. The final checkpoint used for evaluation was selected based on the highest ‘reward/overall‘ achieved during training. These settings were chosen to balance training stability, computational efficiency, and exploration during the reinforcement learning process for the multi-tabular visual question answering task, aiming for both accurate content and correctly formatted output. Key GRPO training parameters are summarized in Table 5.

Parameter	Value
Core Algorithm	
Advantage Estimator	GRPO
KL Coefficient (λ_{KL})	0.01
Training Setup	
Base Model	Qwen/Qwen2.5-VL-3B-Instruct
Training Data	MTabVQA-Instruct (Spider Subset) (2,395 ex.)
Max Training Steps	270
Total Epochs	15
Rollout Batch Size	128
Actor Model (Qwen2.5-VL-3B)	
Learning Rate	1e-06
Optimizer	AdamW (BF16)
Global Update Batch Size	32
Rollout Generation	
Temperature (Training)	1.0
Top-p (Training)	0.99
Num. Generations per Prompt (n)	5

Table 5: GRPO Hyperparameters for Qwen2.5-VL-3B Fine-tuning.

G Model Evaluation and Generation Prompts

This section details the system prompts used for evaluating and generating responses from the Vision-Language Models (VLMs) in different experimental settings.

G.1 Standard Zero-Shot Evaluation Prompt

For standard zero-shot evaluations of VLMs (Section 4.1), including proprietary models and open-source baselines before specific post-training, the following system prompt was used. This prompt instructs the model on how to interpret multi-tabular image data, reason about the question, and provide an answer strictly in the specified JSON format.

System Prompt: Zero-Shot Evaluation

```
You are an intelligent assistant capable of understanding and reasoning about multi-tabular data
given as images, each table is one image. You will be presented with one or more tables containing
information on a specific topic.
You will then be asked a question that requires you to analyze the data in the table(s) and provide a
correct answer in strict required format.

Your task is to:

1. Carefully examine the provided table(s) Pay close attention to the column headers, the data types
within each column, and the relationships between tables if multiple tables are given.
2. Understand the question being asked. Identify the specific information being requested and
determine which table(s) and columns are relevant to answering the question.
3. Extract the necessary information from the table(s). Perform any required filtering, joining,
aggregation, or calculations on the data to arrive at the answer.
4. Formulate a clear and concise answer in natural language. The answer should be directly responsive
to the question and presented in a human-readable format. It may involve listing data, presenting
a single value, or explaining a derived insight.
5. Do not include any SQL queries in the answer. But you can use it internally, to come up with answer.
6. Be accurate and avoid hallucinations. Your answer should be completely based on the data
in the provided table(s). Do not introduce any external information or make assumptions not supported
by the data.
7. Be specific and follow the instructions in the question. If the question ask to get specific
columns, return only mentioned columns.
8. If the question is unanswerable based on the provided tables, state "The question cannot be
answered based on the provided data.
9. Please provide only the answer which has been asked, without any additional text (try to use
few tokens). However, take the time to think and reason before
giving your answer. Also, try to provide an answer even if you are unsure.
10. Provide the answer in JSON format with given response schema as given
[['ans1','ans2'],['ans3','ans4']]. Respond only with valid JSON format.

Take your time to understand the question. Break it down into smaller steps. Come up with
an answer and examine your reasoning. Finally, verify your answer.
you need to extract answers based on the given multi-hop question [Question] and given multiple tables
[TABLE1], and [TABLE2]. Please only output the results without any other words.
Return the answer in the following JSON format.

Return the answer in JSON schema: : {
              "type": "json_schema",
              "json_schema": {
                  "name": "Response",
                  "type": "object",
                  "properties": {
                      "data": {
                          "type": "array",
                          "items": {"type": "array", "items": {"type": "string"}},
                      }
                  },
                  "required": ["data"],
                  "additionalProperties": False,
              },
          }
```

G.2 Chain-of-Thought (CoT) Evaluation Prompt

For the Chain-of-Thought (CoT) prompting experiments (Section 4.2), a modified system prompt was used. This prompt explicitly instructs the model to first generate a step-by-step reasoning process (the chain of thought) and then provide the final answer.

System Prompt: Chain-of-Thought (CoT)

````
You are an intelligent assistant capable of understanding and reasoning about multi-tabular data
given as images, each table potentially being one image. You will be presented with one or more tables
containing information on a specific topic. You will then be asked a question that requires you to
analyze the data in the table(s) and provide a correct answer in the strictly required format.

Your task is to:

1. Carefully examine the provided table(s): Pay close attention to the column headers, the
data types within each column, and the relationships between tables if multiple tables are given.
2. Understand the question being asked: Identify the specific information being requested and
determine which table(s) and columns are relevant to answering the question.
3. Reason step-by-step (Chain of Thought): Before generating the final answer,
formulate a clear chain of thought outlining how you identified the relevant data,
performed necessary operations (filtering, joining, aggregation, calculations),
and arrived at the result. This reasoning is crucial and MUST be included in the final output.
4. Extract the necessary information from the table(s): Perform any required filtering, joining,
aggregation, or calculations on the data based on your chain of thought to arrive at the answer.
5. Do not include any SQL queries in the final answer JSON. You can use SQL logic internally during
your reasoning (Chain of Thought), but the final output should not contain raw SQL code.
6. Be accurate and avoid hallucinations: Your answer must be completely based on the data in
the provided table(s).
. Provide the output strictly in the specified JSON format: The output must be a single JSON object
containing two keys: `chain_of_thought` (a string detailing your reasoning steps) and `data`
(an array of arrays containing the answer).

Your entire response must be ONLY a valid JSON string conforming to the schema below.
JSON Schema:
```json
{
  "type": "object",
  "properties": {
    "chain_of_thought": {
      "type": "string",
      "description": "A detailed step-by-step explanation of the reasoning process
      used to arrive at the answer."
    },
    "data": {
      "type": "array",
      "items": {
        "type": "array",
        "items": {
          "type": "string"
        }
      },
      "description": "The result data, formatted as an array of arrays, where each
      inner array represents a row."
    }
  },
  "required": [
    "chain_of_thought",
    "data"
  ],
  "additionalProperties": False
}
```

Take your time to understand the question and the data. Break the problem down using Chain of Thought.
Construct the final JSON containing both your reasoning and the extracted data. Verify
your answer and the format before outputting. Remember to output ONLY the JSON string.
````

G.3 GRPO Thinking Prompt

For the Group Relative Policy Optimization (GRPO) training and generation (Section 4.2 and Appendix F), the prompt was used. This prompt is similar to the CoT prompt in that it requires an internal reasoning process (‘<think>...</think>‘) before the final answer, but it is specifically tailored for the GRPO framework, which often involves distinct markers for thought processes versus final outputs used in reward calculation. The final answer is expected within ‘<answer>...</answer>‘ tags in a specific JSON format.

System Prompt: GRPO Thinking Prompt

```
You are an intelligent assistant capable of understanding and reasoning about multi-tabular data
given as images, each table is one image. You will be presented with one or more tables containing
information on a specific topic.
You will then be asked a question that requires you to analyze the data in the table(s) and
provide a correct answer in strict required format using multi-hop reasoning.

Your task is to:

1. Carefully examine the provided table(s) Pay close attention to the column headers, the data types
within each column, and the relationships between tables if multiple tables are given.
2. Understand the question being asked. Identify the specific information being requested and
determine which table(s) and columns are relevant to answering the question.
3. Extract the necessary information from the table(s). Perform any required filtering, joining,
aggregation, or calculations on the data to arrive at the answer.
4. Formulate a clear and concise answer in natural language. The answer should be directly
responsive to the question and presented in a human-readable format.
It may involve listing data, presenting a single value, or explaining a derived insight.
5. Do not include any SQL queries in the answer. But you can use it internally, to come up with answer.
6. Be accurate and avoid hallucinations. Your answer should be completely based on the data in the
provided table(s). Do not introduce any external information or make assumptions not
supported by the data.
7. Provide the answer in JSON format with given response schema as given
[['ans1','ans2'],['ans3','ans4']].Respond only with valid JSON format, as shown in the example above.

Strictly, Give answer in this format, using the example below as reference:

You FIRST think about the reasoning process as an internal monologue and then provide the final answer.
The reasoning process MUST BE enclosed within <think> </think> tags. You will be presented with one
or more tables containing information on a specific topic.You will then be asked a question that
requires you to  analyze the data in the table(s) and provide a correct answer.The final answer MUST BE
put in <answer> </answer> in json format.
Example JSON format inside  <answer>{"data": [[‘ans1’, ‘ans2’], [‘ans3’, ‘ans4’]]}</answer>.
```