RealHiTBench: A Comprehensive Realistic Hierarchical Table Benchmark for Evaluating LLM-Based Table Analysis

Pengzuo Wu[1*], Yuhang Yang[1*], Guangcheng Zhu[1], Chao Ye[1], Hong Gu[2], Xu Lu[1], Ruixuan Xiao[1], Bowen Bao [1], Yijing He[1], Liangyu Zha[3], Wentao Ye[1], Junbo Zhao[1], Haobo Wang[1†]

[1]Zhejiang University [2]vivo Mobile Communication Co., Ltd

[3]Institute of Computing Innovation, Zhejiang University

{wupengzuo, yangyuhang, wanghaobo}@zju.edu.cn

Abstract

With the rapid advancement of Large Language Models (LLMs), there is an increasing need for challenging benchmarks to evaluate their capabilities in handling complex tabular data. However, existing benchmarks are either based on outdated data setups or focus solely on simple, flat table structures. In this paper, we introduce **RealHiTBench**, a comprehensive benchmark designed to evaluate the performance of both LLMs and Multimodal LLMs (MLLMs) across a variety of input formats for complex tabular data, including LaTeX, HTML, and PNG. RealHiTBench also includes a diverse collection of tables with intricate structures, spanning a wide range of task types. Our experimental results, using **25** state-of-the-art LLMs, demonstrate that RealHiTBench is indeed a challenging benchmark. Moreover, we also develop TreeThinker, a tree-based pipeline that organizes hierarchical headers into a tree structure for enhanced tabular reasoning, validating the importance of improving LLMs' perception of table hierarchies. We hope that our work will inspire further research on tabular data reasoning and the development of more robust models. The code and data are available at https://github.com/cspzyy/RealHiTBench.

1 Introduction

In recent years, there has been a lot of work on table analysis (Li et al., 2024b; Zhao et al., 2024). Specifically, an increasing number of studies discuss table question answering (TableQA) tasks. (Yang et al., 2024b; Sarkar and Lausen, 2023; Zhou et al., 2024; Nguyen et al., 2024; Li et al., 2024a). Additionally, the introduction of large language models (LLMs), including close-source models such as GPT-4o (OpenAI, 2023) and Gemini-1.5-Pro (Anil et al., 2023), and table-oriented models like TableGPT (Su et al., 2024)

For Children under 18, which exact activity of Household costs the most time of Mothers when Mother part time and Father full time?

Activity	Average hours per day				Activity	Average hours per day			
	Both full time		Mother part time & Father full time			Both full time		Mother part time & Father full time	
	Mothers	Fathers	Mothers	Fathers		Mothers	Fathers	Mothers	Fathers
Children under 18					Children 7-12				
Total	36.31	10.34	12.42	9.99	Total	11.47	10.29	12.15	9.77
PersonCare	9.01	8.62	9.10	8.45	PersonCare	9.04	8.67	9.08	8.34
Sleeping	8.20	8.04	8.34	7.91	Sleeping	8.29	8.12	8.37	7.84
Household	2.00	1.35	2.62	1.21	Household	1.85	1.23	2.38	1.10
Housework	0.87	0.27	1.13	0.23	Housework	0.78	0.29	1.03	0.26
Food	0.78	0.35	1.01	0.29	Food	0.78	0.36	0.99	0.30
Lawn	0.08	0.23	0.12	0.24	Lawn	0.07	0.17	0.09	0.19
Purchasing	0.58	0.37	0.70	0.33	Purchasing	0.58	0.39	0.69	0.33
Grocery	0.13	0.06	0.17	0.06	Grocery	0.14	0.07	0.15	0.07
Consumer	0.35	0.24	0.41	0.22	Consumer	0.34	0.25	0.41	0.21
Children under 6					Children 13-17				
Total	13.00	11.20	13.29	10.74	Total	11.84	10.51	12.73	10.24
PersonCare	9.87	9.12	9.52	8.81	PersonCare	8.99	8.59	9.10	8.54
Sleeping	9.02	8.88	8.96	8.26	Sleeping	8.14	7.98	8.32	7.97
Household	2.34	1.57	2.91	1.47	Household	2.11	1.44	2.81	1.29
Housework	1.02	0.36	1.49	0.43	Housework	0.92	0.26	1.20	0.20
Food	0.99	0.41	1.38	0.37	Food	0.78	0.34	1.03	0.29
Lawn	0.11	0.23	0.14	0.28	Lawn	0.10	0.21	0.12	0.22
Purchasing	0.79	0.51	0.86	0.46	Purchasing	0.74	0.48	0.82	0.41
Grocery	0.18	0.10	0.19	0.11	Grocery	0.16	0.09	0.16	0.11
Consumer	0.47	0.31	0.53	0.28	Consumer	0.44	0.29	0.49	0.26

Hierarchical Column Header | Hierarchical Row Header | Nested Sub-Tables | Implicit Multi-Table Join

Figure 1: A complex table example with typical complex structures and a question-answer example.

further enhances models' capability of table comprehension. Given these advancements, the rapid development of LLMs and the continuous emergence of diverse tabular data in real-world scenarios have highlighted the need for a comprehensive evaluation of LLMs' table understanding capabilities, putting forward new requirements on the relevant benchmarks.

Unfortunately, we find that the current tabular data benchmarks, such as TAT-QA (Zhu et al., 2021), TableBench (Wu et al., 2024), and InfiAgent-DABench (Hu et al., 2024), largely consist of flat tables. More concretely, in such tables, each column represents an attribute, each row represents a record, and all data are stored in a simple one-dimensional format; see Figure 13 for visual demonstration. However, in practical applications, humans often organize relatively complex tables to represent multifaceted relationships between variables. Such hierarchically structured tables are popular in various domains, including economy, science, and employment, on public data platforms.

[*] Equal contribution.

[†] Corresponding author.

Findings of the Association for Computational Linguistics: ACL 2025, pages 7105–7137

To address this issue, some benchmarks have considered hierarchical tables, such as HiTab (Cheng et al., 2022) and SciTab (Lu et al., 2023). However, these benchmarks fail to comprehensively present the LLMs' understanding capabilities of complex table structures. For example, SciTab (Lu et al., 2023) and AIT-QA (Katsis et al., 2022) only consider tables of the scientific and airline domains, respectively. MM-Tab (Zheng et al., 2024) provides only image-based input, while hierarchical tables in realistic applications are also presented in a textual way. SpreadSheetBench (Ma et al., 2024) focuses on the operation of tables. HiTab (Cheng et al., 2022) organizes its tables with lossy JSON format, only contains fundamental QA tasks, and provides incomplete supervision. More importantly, most of these benchmarks focus primarily on relatively simple hierarchical tables, particularly those with a basic column hierarchy, where the hierarchy typically does not exceed two levels. To date, there is still a lack of dedicated benchmarks for comprehensively evaluating LLMs' ability to understand complex tabular hierarchies.

To this end, we propose **RealHiTBench**, a challenging **Real**istic **Hi**erarchical **T**able **Bench**mark based on complex tables and tasks. **(i) Complex Table Structures:** Our benchmark includes complex tables with intricate features (partially depicted in Figure 1), which are commonly found in real-world scenarios but often overlooked in existing benchmarks. **(ii) Modal and Format Diversity:** We explore the performance of both text and image approaches while others focusing solely on one of them. Our benchmark provides LLMs and MLLMs to be tested with various input formats, such as LaTeX, HTML, and PNG. **(iii) Question Diversity:** RealHiTBench covers a wide range of question types, each designed to test different aspects of a model's ability. Especially a type called *Structure Comprehending* is designed on the basis of complex parts. **(iv) Accurate and Efficient Annotation:** Our benchmark employs a rigorous annotation process. GPT-based automated annotation and human checks ensure the accuracy and reliability of the questions and answers.

We conducted a comprehensive evaluation of abundant types of models consisting of table-oriented LLMs, and open-source/closed-source generic LLM/MLLMs. In addition, we tested a couple of table reasoning tasks with different proper metrics. The performance of different models differs greatly, with average scores of all tasks ranging from 7.27 to 56.95. Importantly, overall low scores (mainly below 70) highlight that the ability of LLMs to comprehend and process intricate table structures remains an area in need of significant improvement. We also develop a tree-based pipeline (dubbed TreeThinker) that automatically injects table hierarchies into instructions. Empirically, we show that, *with self-emphasized table hierarchies, LLMs' structural understanding ability can indeed be enhanced.*

2 Related Work

Table Analysis. Table analysis is pivotal across numerous domains (He et al., 2024), while table question answering (TableQA) can effectively assess analytical capabilities (Müller et al., 2019). Recently, there has been a growing number of studies focusing on the ability of LLMs to comprehend tabular data (Singha et al., 2023; Nguyen et al., 2024; Deng et al., 2024). However, the development of LLMs leads to the emergence of different branches. Common tabular data are in text form to be processed by LLMs (Anil et al., 2023; Yang et al., 2024a; Dubey et al., 2024), while other image-based tables are suited to be processed by MLLMs (Wang et al., 2024; Liu et al., 2024). Additionally, some table-oriented LLMs for both modalities have also been proposed (Su et al., 2024; Zheng et al., 2024). Therefore, the flourishing development of LLMs makes challenging benchmarks for tabular data increasingly important.

Table Analysis Benchmarks. An increasing number of benchmarks for table analysis are discussed (Chen et al., 2020; Nan et al., 2022). However, a proportion of benchmarks focus more on other aspects, such as reasoning methods and supervision fine-tuning, while overlooking the structural complexity of the data, especially the table dimensions (Wu et al., 2024; Zheng et al., 2024). Although some recent benchmarks have introduced the concepts of hierarchical tables (Cheng et al., 2022; Katsis et al., 2022; Ma et al., 2024), and a complex question answering benchmark has been proposed to bridge the gap between theoretical and real-world tabular data (Wu et al., 2024), the tables in these benchmarks are not complex enough and there has been no detailed discussion of the tables.

Therefore, in order to introduce a benchmark that can effectively evaluate current LLMs, we propose RealHiTBench consisting of complex tables and TableQA, which evaluates LLMs and MLLMs, as

well as textual and visual input data. Furthermore, we look forward to the possible future development directions of LLMs in table reasoning.

3 RealHiTBench

In practical applications, tabular data holds significant value. Numerous previous studies have proposed a series of benchmarks (Zhu et al., 2021; Parikh et al., 2020), which have effectively propelled research in this field. However, as application scenarios become increasingly complex and large language models (LLMs) gradually emerge as the mainstream method for table reasoning (Yang et al., 2024b), the previous benchmarks struggle to provide a comprehensive and accurate assessment of model capabilities. Hence, there is an urgent need for a new, realistic, and challenging benchmark to evaluate the progress of research methods in this field. We propose RealHiTBench focusing on complex structures and tasks. Complex tables, including hierarchical tables, are widely used in various domains (Lu et al., 2023; Katsis et al., 2022). We propose such a challenging benchmark full of complex tables and corresponding difficult tasks to assess the upper bound of the comprehension capability of LLMs, which may further inspire future research on processing tables with LLM.

3.1 Table Collection and Process

To construct a realistic and comprehensive dataset, all tables in our benchmark are raw from 13 different open platforms and cover 24 different domains like economy, society, science, and so on (shown in the Appendix A.1 and A.3). In real applications, table-related benchmarks are presented in either textual (Cheng et al., 2022; Parikh et al., 2020) or visual (Kim et al., 2024) manner. However, there is still no fixed format for complex table understanding. Therefore, we explore the influence of textual and visual ways. Specifically, we also explore the influence of different input formats, including LaTeX, HTML, CSV, Markdown, and PNG, whose performance is shown in Figure A.2.

3.2 Complex Structures

In actual scenarios, we often encounter tables that show structural and even semantic complexity. However, current benchmarks (Cheng et al., 2022; Lu et al., 2023) have not adequately discussed the complexity of table structure. Concentrating on complexity in tabular data, we define 5 complex

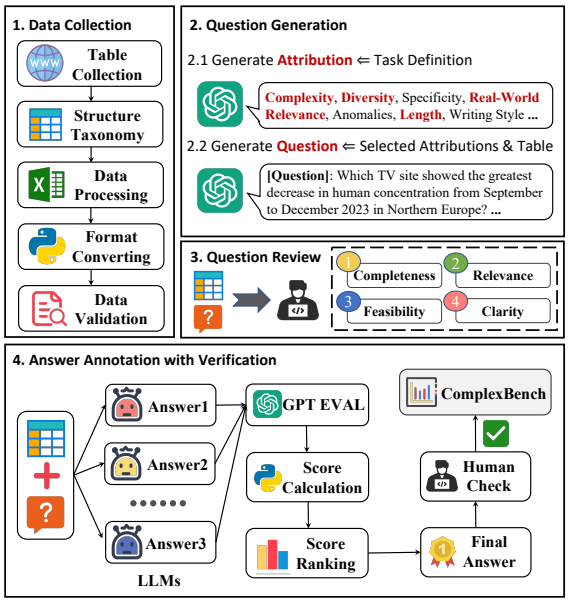

Figure 2: The construction process of **RealHiTBench**.

structure types as following definitions. Most types of complex structure are presented in Figure 1.

(1) Hierarchical Column Header. A column header, typically located at the top of a table, serves as the title for each column. It identifies the category, attribute, or subject of the data in that column and provides a structured organization of the table data. In most cases, the feature of the complex column header is cell merging, which leads to the complex header hierarchy in the table (the cells with green background in Figure 1).

(2) Hierarchical Row Header. A row header is a label or identifier for each row in a table, typically located at the beginning of the row. In realistic tables. Row headers sometimes use indentation or clustering to present classified semantics. The common hierarchical row header is presented by indentation within a single column (the cell groups in blue within Figure 1). Moreover, the hierarchical row header is also presented by multiple columns, with a large merged cell corresponding to several subcategory cells in horizontal direction.

(3) Nested Sub-Tables. Sometimes, due to semantic requirements, a whole table consists of several areas with vertical segmentations. Typical presentation is that there are horizontal cells spanning the full width of the table (the cells in orange within Figure 1). These full-width cells divide the root table into nested sub-tables.

(4) Multi-Table Join. Speaking of the dimensions of a table, single-table scenarios cannot nec-

Table 1: Comparison with existing datasets in Dataset Information, Task Types, and Input Formats. Here are abbreviations inside and their meanings: FC stands for Fact Checking, NR for Numerical Reasoning, DA for Data Analysis, CG for Chart Generation, and SC for Structure Comprehending.

Benchmark	Year	Dataset Information					Task Type					Data Format	
		Hierarchy	#Table	#QA pairs	Source	Domain	FC	NR	DA	CG	SC	Text	Image
HiTab	2022	High	3,597	10,672	2	28	✓	✓	✘	✘	✘	✓	✘
AIT-QA	2022	High	116	515	1	1	✘	✓	✘	✘	✘	✓	✘
MultiHierTT	2022	High	1,800	10,440	1	1	✘	✓	✘	✘	✘	✓	✘
TableBench	2024	Low	586	3,544	6	18	✓	✓	✓	✓	✘	✓	✘
InfiAgent-QABench	2024	Low	52	257	1	9	✘	✓	✓	✘	✘	✓	✘
TableVQA-Bench	2024	Low	894	1,500	3	4	✓	✓	✘	✘	✘	✓	✓
Text2Analysis	2024	Low	347	2,249	1	1	✘	✘	✓	✓	✘	✓	✘
RealHiTBench	**2025**	**High**	**708**	**3,752**	**13**	**24**	✓	✓	✓	✓	✓	✓	✓

essarily reflect the complexity of real-world applications, and multi-table tasks have been proposed in recent work (Wu et al., 2025). We extend the representation of multiple tables to two refined categories: *(i) Explicit Multi-Table Join:* We define the majority of the multi-tables that have been discussed as Explicit Multi-Table (shown in Figure 14). *(ii) Implicit Multi-Table Join:* However, we notice other special multi-tables in our dataset. As depicted with red box in Figure 1, there are subtables with the same structures, especially column headers. The implicit Multi-Table type looks not different from a normal single table, but in fact, it is in the form of multiple tables. Interestingly, this type sometimes contains additional semantics, such as comparisons, but is not easy to detect when comprehending tables.

(5) Miscellanies. We also find that some other special contents obtain a part of the tabular information, including *additional explanatory texts* outside the table and *cell background colors*. These nonstructural elements also play a certain role in complex tables.

Remark. It is worth noting that the hierarchical information we consider also appears in some existing benchmarks. However, we include tables with higher complexity (see Table 8 for more details). Notably, it is necessary to clarify some differences between our work and the previous HiTab (Cheng et al., 2022) benchmark. First, HiTab pre-extracts the tree structure of hierarchical tables in JSON format, which prevents effective evaluation of whether LLMs can directly understand structural information from table inputs. Additionally, it suffers from a restricted focus on three domains, a simplistic QA task, and incomplete supervision.

3.3 Complex Tasks

In order to evaluate the model's abilities comprehensively, we define 5 primary types following TableBench (Wu et al., 2024): *Fact Checking (FC)*, *Numerical Reasoning (NR)*, *Data Analysis (DA)*, *Chart Generation (CG)*, and *Structure Comprehending (SC)* (in the Table 2). More detailed subtypes (shown in Appendix B.1) are derived from the above types. Notably, we extend one type named ***Structure Comprehending***, which is tailored for complex tables. This kind of task provides the new table by exchanging some complex parts of the source table. Then we ask LLMs the same question with two similar tabular input to evaluate the ability of LLMs to comprehend structures.

Based on the question types above, we meticulously design pretty difficult questions for each complex structure. We take questions generated from the table in Figure 1 as an example: **(i) Header Hierarchy:** For the characteristics of header hierarchy, a possible question is: *For Children under 18, which exact activity of Household cost the most time of Mothers when both full time?* It is difficult to identify the scale of columns and rows with classification and hierarchy. **(ii) Nested Sub-Tables:** several segments of a whole table lead to interesting semantic tricks. One challenging question is: *How much total time costs Mothers in Both full time for all the children recorded in the table?* The correct conclusion is to just sum up total time for children under 18, instead of the whole table. Such summing questions with content

partition tests if models thoroughly understand the inclusion relationships among sub-tables.

3.4 Annotation Generation

Question Generation. Following the previous work (Yu et al., 2023a), we select some attributes that we also agree are important as the core for question construction from what GPT-4o considers. After feeding different prompts of each question type and acquiring generated questions, we conduct manual question review according to four factors: *Relevance*, *Completeness*, *Feasibility*, and *Clarity*. After the first turn of annotation, we swap questions of each other to check the rationality.

Answer Generation. To eliminate the instability of GPT generation, we generate answers with GPT-4o based on well-designed answer prompt and innovate the pipeline: We feed a table with prompts to several models, acquiring multiple results and the frequency of occurrence of each result. Then the results are input into the *G-Eval* (Liu et al., 2023) module and scored for each result. Thus, final scores of each result can be figured out with initial score and frequency. The result with the highest final score becomes the candidate answer. We also conduct careful human check again that the table expert is dedicated to check answers one by one according to the question and exact table content. Furthermore, we employ a tree-based method to enhance our annotation process, finding that the automatic annotation accuracy is further improved.

3.5 Dataset Statistics

We propose RealHiTBench containing 708 tables and 3,752 questions. RealHiTBench completely focuses on discussing complex tables with such characteristics: **(1)** Tables in our benchmark are totally raw from available public platforms, and our benchmark totally consists of complex tables in structure or semantics. Thus, we organize 6 university students as annotators spending 150 hours each person on collecting these tables. By the way, annotators spend 480 hours each person on question answering annotations. **(2)** Compared to work involving complex tables such as HiTab (Cheng et al., 2022) and MultiHierTT (Zhao et al., 2022) before, the tables in our dataset are more complex (quantized in Table 8), even the median sizes of the tables are larger than tables in other works (Wu et al., 2024). **(3)** We systematically define different types of complex tabular structure and classify them for further exploration. **(4)** We introduce a couple of challenging task types. The tasks of *structure comprehending* prove to be difficult for LLMs (shown in section 5.3). The above characteristics are shown in Figure 1 and Table 1.

4 TreeThinker

Beyond benchmark, we propose **TreeThinker**, a understanding-augmented pipeline that enhances the model's ability in complex hierarchical tables question answering. As shown in Figure 3, we first prompt the model to **automatically** organizing hierarchical headers into **Tree Structure**, then aligning keywords from questions with the tree headers, and efficiently locating sub-table relevant to the question.

4.1 Tree Generation

Complex tables often contain multi-level row and column headers, making it difficult for models to accurately analyze their intricate hierarchical relationships. Therefore, to enhance the model's ability in understanding complex table structures, we prompt model to explain the table's headers structure and organize them into tree structure.

Explanation. We first enable the model to self-explain the structural information of the table headers, including their meanings, scopes of influence, hierarchical structure, and relationships between headers. Referring to the previous approach (Zhao et al., 2023), we prompt the model by encoding each node of the table header as a tuples $T = (t_1, t_2, t_3, t_4)$. Specifically, the first element represents row header (R) or column header (C), along with its corresponding level. And the second and third elements represent its start and end positions, while the fourth element contains the content of the cell.

For example, a tuple (R0, 1, 2, City) indicates that it is a row header (R) at level 0, spanning from row 1 to row 2, with the value City. This approach compresses header information into a tuple list $L = [T_1, T_2, \cdots, T_n]$, enables the model to more clearly identify the hierarchical structure and generating a "Structural Blueprint" of the table that effectively guides subsequent reasoning.

Generation. The tree structure, with its clear parent-child connections, can accurately represents the hierarchical relationships of table headers, while making table data organization and retrieval

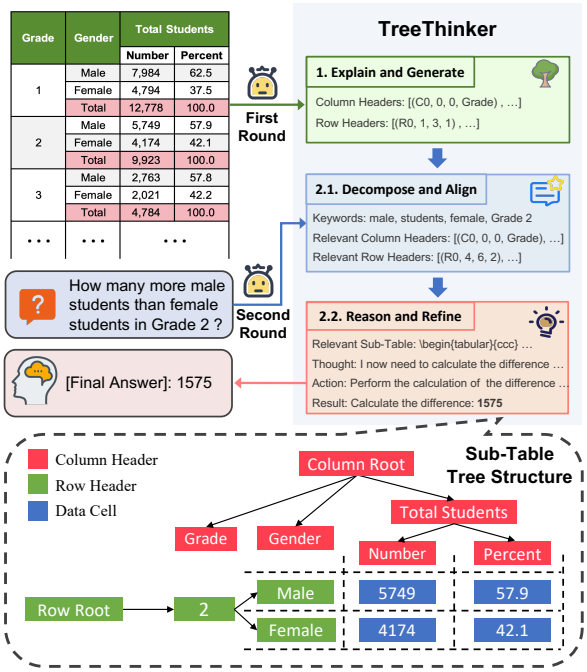

Figure 3: The overall framework of **TreeThinker**.

more intuitive and efficient. Therefore, we ask the model to organize the scattered header list L based on their hierarchical relationships into the Table-Header Tree H with following steps: **(1)** Divide the tuple list L into groups according to their levels, such that all tuples with the same level are grouped together. Add a special ROOT node Leveled "-1" for rows and columns. **(2)** For each tuple $A \in L$. If the start and end positions of A are equal ($A_{t_2} = A_{t_3}$), mark A as a leaf node. **(3)** Otherwise, compare its t_2 and t_3 values with every closest higher level and same flag tuple B. If tuple A is within the range of tuple B ($A_{t_2} \geq B_{t_2}$ and $A_{t_3} \leq B_{t_3}$), then B is the parent-header of A. **(4)** Repeat steps 2 and 3 iteratively until all tuples in L are linked to their respective parent nodes (Tuples without parent node are linked to the ROOT node), forming a hierarchical Table-Header Tree H.

4.2 Tree-based Reasoning

Previous studies (Shi et al., 2023) have shown that LLMs are often distracted by irrelevant information. To address this issue, we prompt the model to decompose question into keywords and align them with the table headers, thereby helping it accurately identify the sub-table relevant to the question.

Decomposition. Decomposing questions into more fine-grained components can simplify the reasoning process required and enhance the model's performance. Therefore, we first prompt the model to decompose the question Q into several keywords $K = [k_1, k_2, ..., k_m]$, helping it focus on the most critical aspects of the question.

Aligning and Reasoning. Once the question decomposition is finished, we instruct the model to align keywords with header tuples, enabling it to accurately pinpoint the headers relevant to the question. Specifically, given the keywords K and the header tuples H, the goal of Aligning is to build a Keyword-Header Tree $H' = select(T, Align_{LM}(T, k) > \theta), T \in H, k \in K$. The function $Align_{LM}(T, k)$ calculate the degree of matching between header tuple T and keyword k. The $select()$ function then chooses the header T whose matching degree with keyword k exceeds the threshold θ and add them to the Keyword-Header Tree H'. After that, we incorporate H' in prompts, guiding the LLM to retrieve essential information, e.g. relevant sub-tables, from tables with improved reasoning abilities.

Lastly, we also supplement a React-Style multi-round refinement strategy — through multiple rounds of *Thought*, *Action* and *Result*, ultimately outputting the final answer. Full prompts are shown in Table 17 and 18.

5 Experiment

5.1 Experimental Setup

Baselines. We evaluated 25 models, including different modalities, with parameters ranging from 7B to 90B across four categories: (1) Close-source models, including GPT-o1 (OpenAI, 2024), GPT-4o (OpenAI, 2023), Deepseek-R1-API(Guo et al., 2025), Gemini-1.5-pro (Anil et al., 2023), QwQ (Team, 2024) and Doubao-1.5-pro (Team, 2025) . (2) Open-source language models, including Llama3s (Dubey et al., 2024), Qwen2.5s (Yang et al., 2024a), Mistral (Jiang et al., 2023) and Deepseek-R1-Distalls (Guo et al., 2025). (3) Open-source multimodal models, including LLaVA-1.5 (Liu et al., 2024), mPLUG-owl3 (Ye et al., 2024), mPLUG-owl2 (Ye et al., 2023), and Llama3-Visions. (4) Table-oriented models, including TableGPT (Su et al., 2024), TableLLMs (Wu et al., 2024), TableLlama (Zhang et al., 2024), and Table-LLaVA (Zheng et al., 2024).

Metrics. For Fact Checking, Numerical Reasoning and Structure Comprehending, we refer to previous work (Zhao et al., 2022) using F1 and EM as score metric for these tasks. For chart generation,

Table 2: The evaluation results of advanced models with Text and Image Inputs on RealHiTBench.

Model	Input	Fact Checking		Numerical Reasoning		Structure Comprehending		Data Analysis		Chart Generation	
		EM	F1	EM	F1	EM	F1	GPT-EVAL	ROUGE	PASS@1	ECR
Table-oriented Models											
TableGPT2-7B	Text	**46.10**	**53.80**	**29.31**	**39.81**	48.23	56.68	**62.76**	**33.25**	**32.47**	**67.53**
TableLlama	Text	14.30	19.35	7.26	12.63	36.31	42.91	24.73	7.89	0	0
TableLLM-Llama3.1-8B	Text	33.44	39.49	13.36	21.02	**53.28**	**58.72**	47.86	27.26	6.49	22.73
TableLLM-Qwen2-7B	Text	33.53	40.41	22.05	29.04	36.09	44.75	47.30	18.86	4.55	10.39
Table-LLava-7B	Image	4.19	7.05	2.98	5.59	7.38	11.83	22.46	8.72	0	1.30
Open-source Large Language Models											
Llama3.1-8B-Instruct	Text	30.32	44.93	14.53	27.21	35.90	50.80	60.12	32.25	4.55	13.64
Qwen2.5-7B-Instruct	Text	18.65	38.39	5.32	19.75	23.48	44.81	40.17	24.17	15.58	48.70
Mistral-7B-Instruct-v0.3	Text	15.61	31.88	3.37	16.10	21.21	49.62	57.74	19.84	7.14	18.18
Llama3.3-70B-Instruct	Text	**53.08**	**64.53**	**36.58**	**48.99**	**55.81**	**68.93**	52.26	27.98	**24.03**	**50.65**
Qwen2.5-72B-Instruct	Text	51.93	62.15	26.98	39.23	54.55	68.34	**68.45**	**35.90**	14.29	27.27
Open-source Multimodal Large Language Models											
LLaVa-v1.5-7B	Image	5.51	9.37	1.17	6.54	13.23	30.51	26.60	17.74	0	11.04
mPLUG-Owl3-7B	Image	8.71	14.34	4.02	12.72	41.24	48.86	30.71	18.53	3.25	6.49
Llama3.2-11B-Vision-Instruct	Text	18.41	38.44	9.86	22.97	23.39	43.22	52.75	30.59	4.55	17.53
Llama3.2-11B-Vision-Instruct	Image	15.84	26.55	9.59	18.49	20.87	32.57	41.32	22.75	1.95	20.78
Llama3.2-11B-Vision-Instruct	Image+Text	19.39	32.54	10.89	20.94	27.99	43.85	39.98	22.43	6.49	35.71
Llama3.2-90B-Vision-Instruct	Text	52.10	61.84	28.15	41.21	58.52	69.57	**57.28**	31.87	7.79	22.73
Llama3.2-90B-Vision-Instruct	Image	23.75	36.08	25.46	28.05	33.33	46.40	41.18	25.46	5.19	16.23
Llama3.2-90B-Vision-Instruct	Image+Text	**55.60**	**65.19**	**38.65**	**49.71**	**59.80**	**70.23**	53.06	**32.36**	**13.64**	**39.61**
Close-source Models											
GPT4o	Text	60.31	68.97	38.65	50.12	63.04	71.14	73.37	36.36	**20.13**	**40.26**
GPT4o	Image	43.39	51.87	27.63	36.68	42.68	52.89	65.24	33.10	10.39	25.32
GPT4o	Image+Text	62.45	69.01	41.37	49.59	65.91	74.49	72.05	35.25	14.29	30.52
Gemini1.5-pro	Text	59.08	66.14	35.54	43.74	63.64	69.71	70.72	36.17	9.74	25.32
Gemini1.5-pro	Image	50.37	57.62	25.42	33.76	41.52	50.23	67.22	36.26	7.79	38.96
Gemini1.5-pro	Image+Text	62.04	68.08	37.61	45.62	56.74	65.80	74.80	36.26	9.09	24.03
DeepSeek-R1	Text	**70.91**	**79.45**	**70.31**	**72.54**	**82.71**	**84.62**	**79.55**	**42.59**	7.14	32.14
Models with TreeThinker											
GPT4o(TreeThinker)	Text	64.50	72.41	53.34	**65.08**	64.40	75.67	77.26	**37.63**	**39.47**	**67.76**
GPT4o(TreeThinker)	Image	44.13	52.41	40.57	49.35	49.21	58.32	70.83	34.44	19.61	67.32
GPT4o(TreeThinker)	Image+Text	**65.82**	**73.32**	**55.60**	64.28	**66.31**	**77.42**	**79.45**	37.08	33.55	65.13

we calculate the ECR to assess code executability, extract y-axis values to compare with reference data, and compute PASS@1 to evaluate the pass rate. For Data Analysis, we calculate ROUGE-L (Lin, 2004) to evaluate the objective similarity between generated and reference answers. Inspired by G-EVAL(Liu et al., 2023), we also design an evaluation template using GPT-4o to evaluate the scores of model answers. A detailed evaluation prompt template can be found in Appendix C.3.

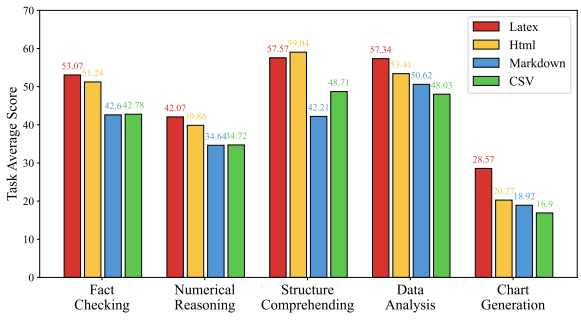

Figure 4: The comparison of average scores for different text formats across various tasks.

5.2 Implementation Details

We deploy open-source models on 8 H800 GPUs using the *Transformer* library, while closed-source models are accessed through official APIs in accordance with their documentation. For table input formats, we tested GPT-4's performance on LaTeX, HTML, Markdown, and CSV formats as text modality input, with LaTeX outperforming the others, as shown in the figure 4. Therefore we select Latex for text modality input. Meanwhile, we selected PNG format tables for visual modality input. Additionally, we set the model's temperature to 0 to ensure deterministic outputs and configured the maximum output length to 4,096 tokens, balancing detail and efficiency. In order to focus on the hierarchical characteristics, we separate long tables from normal-size tables in experiment, but we still keep long tables in our dataset. For more details, please refer to the appendix C.

5.3 Main Results

① **Overall Performance.** The results in Table 2 highlight *the limitations of current LLMs in handling realistic hierarchical table analysis.* For our tasks, the EM metric of almost all models remains

is relatively low, with the highest not exceeding 70. Plus, all models demonstrate remarkably low outcomes in Chart Generation, with code execution accuracy below 30. Interestingly, many Table-oriented models exhibit pronounced overfitting. However, TableGPT2 delivers the most impressive results among 7B-level models. For models with over 10B parameters, GPT-4o and LLama show comparable performance. It's worth mentioning that DeepSeek-R1 achieves the most outstanding results among all models. Although this might be attributed to its 671B MoE architecture, it also, to some extent, indicates the potential of reasoning for large models in addressing hierarchical structures.

② **The text modality outperforms the vision modality as an input.** On average, GPT-4o with text input outperforms image input by 15, while Gemini-Pro with text input exceeds image input by 10. Similarly, open-source MLLMs generally lag behind their backbone LLMs. Some table-oriented MLLMs like Table-LLaVa also perform worse than other textual ones. The performance of image inputs is not as good as that of text inputs, but they may serve as a complementary to text inputs.

③ **Automatic tree reasoning significantly enhances structure understanding ability.** Compared to the baseline, GPT-4o with our TreeThinker method demonstrates consistent improvements across different input modalities, achieving the best performance in RealHiTBench. For example, GPT-4o in the Chart Generation, the PASS@1 for text+image input increased from 14.29 to **33.55**, marking a **134.7%** improvement. Even for text-only input, TreeThinker enhances performance in NR, raising F1 from 36.68 to **49.35**.

Table 3: The evaluation results of advanced models with Text and Image Inputs on RealHiTBench.

Model	Input	Table Tokens		
		<10K	10K-20K	>20K
GPT-4o	Text	56.45	40	30.77
GPT-4o	Image	42.54	21.83	13.48
Llama3	Text	50.9	39.14	27.78
Llama3V	Image	30.03	18	9.01

④ **Too long table size still matters.** While handling tables, we identify 127 tables with complex structures that are significantly larger, making it impossible for LLMs to take in the complete table within a single conversation during evaluation. We extracted the impact of different table sizes on the performance of various modal models, which indicates that long table sizes do affect models' table comprehension abilities, with a more pronounced impact on visual modality models. Given that long tables can exhibit unique complexities, we consider breaking down the large-scale content and inputting them into LLMs with multi-turn dialogues, which are likely to disrupt the tabular information of the tables, which is unacceptable. Therefore, we hope that future works can develop reasonable methods to fully utilize the value of these long tables, or that future LLMs can support the input requirements for enormous tables due to real usage requirements.

Table 4: The ablation results of TreeThinker when using GPT-4o and Llama3.3-70b-Instruct as base models.

Components of TreeThinker	GPT-4o		Llama3-70b	
	Avg Score	Delta	Avg Score	Delta
All	63.29	0	62.23	0
w/o Tree Generation	55.27	**-8.02**	54.04	**-8.19**
w/o Tree-based Reasoning	60.75	**-2.54**	54.58	**-7.65**

⑤ **Each TreeThinker component enhances the model's effectiveness.** As indicated in Table 4, we conduct ablation studies on GPT-4o and Llama3-70b to assess the impact of various components on the performance of TreeThinker. We find that each component contributes to the model's effectiveness, with Tree Generation playing a particularly crucial role in enhancing the model's understanding of realistic complex hierarchical table.

Note: We place more experimental results in the Appendix D.

6 Conclusions and Future Directions

In this paper, we establish a new benchmark for evaluating modern LLMs' ability to handle complex hierarchical tables. Our results with 25 LLMs suggest that it is still challenging for LLMs to understand realistic complex hierarchical table analysis. Based on our full empirical findings, we anticipate further research in table analysis would consider some promising directions such as **(i)**-improving the visual structure understanding ability of MLLMs; **(ii)**-developing better tree-based structural reasoning techniques; **(iii)**-scaling LLMs to handle very long tables.

Limitations

We discover some limitations during benchmark construction. **(i)** We recruit and train 6 college students, who spent 540 hours each on data collection, processing, and annotation. We focus on the complexity of the tables and the accuracy of the annotations, so there is still room for improvement in terms of data volume. **(ii)** In the TreeThinker pipeline, we implement a multi-turn prompting approach, which significantly enhances the model's ability to understand table structures. However, this slows down the process of applying the TreeThinker pipeline, presenting a trade-off.

Ethical Considerations

RealHiTBench, an open English benchmark that supports research in the field of table analysis. Our table data sources are all from well-known public data platforms on the Internet, and the data are fully usable. To ensure that our dataset does not contain any sensitive information, we establish strict review rules for annotators and set up 3 rounds of human checks throughout the annotation process, corresponding to the table, question, and answer, respectively. Therefore, we propose a benchmark that is clean and free from any ethical issues. More details about our dataset and annotation are in Section 3 and Appendix A and B.

Acknowledgments

This work is supported by NSFC under Grants (No. U24A201401) and partially supported by Pioneer R&D Program of Zhejiang (No. 2024C01035).

References

Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, Anja Hauth, Katie Millican, David Silver, Slav Petrov, Melvin Johnson, Ioannis Antonoglou, Julian Schrittwieser, Amelia Glaese, Jilin Chen, Emily Pitler, Timothy P. Lillicrap, Angeliki Lazaridou, Orhan Firat, James Molloy, Michael Isard, Paul Ronald Barham, Tom Hennigan, Benjamin Lee, Fabio Viola, Malcolm Reynolds, Yuanzhong Xu, Ryan Doherty, Eli Collins, Clemens Meyer, Eliza Rutherford, Erica Moreira, Kareem Ayoub, Megha Goel, George Tucker, Enrique Piqueras, Maxim Krikun, Iain Barr, Nikolay Savinov, Ivo Danihelka, Becca Roelofs, Anaïs White, Anders Andreassen, Tamara von Glehn, Lakshman Yagati, Mehran Kazemi, Lucas Gonzalez, Misha Khalman, Jakub Sygnowski, and et al. 2023. Gemini: A family of highly capable multimodal models. *CoRR*, abs/2312.11805.

Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. 2020. 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.

Zhoujun Cheng, Haoyu Dong, Zhiruo Wang, Ran Jia, Jiaqi Guo, Yan Gao, Shi Han, Jian-Guang Lou, and Dongmei Zhang. 2022. Hitab: A hierarchical table dataset for question answering and natural language generation. 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 1094–1110. Association for Computational Linguistics.

Irwin Deng, Kushagra Dixit, Vivek Gupta, and Dan Roth. 2024. Enhancing temporal understanding in llms for semi-structured tables. *CoRR*, abs/2407.16030.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Virginia K. Felkner, Jennifer A. Thompson, and Jonathan May. 2024. GPT is not an annotator: The necessity of human annotation in fairness benchmark construction. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 14104–14115. Association for Computational Linguistics.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Xinyi He, Mengyu Zhou, Xinrun Xu, Xiaojun Ma, Rui Ding, Lun Du, Yan Gao, Ran Jia, Xu Chen, Shi Han, Zejian Yuan, and Dongmei Zhang. 2024. Text2analysis: A benchmark of table question answering with advanced data analysis and unclear queries. In *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*, pages 18206–18215. AAAI Press.

Xueyu Hu, Ziyu Zhao, Shuang Wei, Ziwei Chai, Qianli Ma, Guoyin Wang, Xuwu Wang, Jing Su, Jingjing Xu, Ming Zhu, Yao Cheng, Jianbo Yuan, Jiwei Li, Kun Kuang, Yang Yang, Hongxia Yang, and Fei Wu.

2024. Infiagent-dabench: Evaluating agents on data analysis tasks. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *CoRR*, abs/2310.06825.

Yannis Katsis, Saneem A. Chemmengath, Vishwajeet Kumar, Samarth Bharadwaj, Mustafa Canim, Michael R. Glass, Alfio Gliozzo, Feifei Pan, Jaydeep Sen, Karthik Sankaranarayanan, and Soumen Chakrabarti. 2022. AIT-QA: question answering dataset over complex tables in the airline industry. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Track, NAACL 2022, Hybrid: Seattle, Washington, USA + Online, July 10-15, 2022*, pages 305–314. Association for Computational Linguistics.

Yoonsik Kim, Moonbin Yim, and Ka Yeon Song. 2024. Tablevqa-bench: A visual question answering benchmark on multiple table domains. *CoRR*, abs/2404.19205.

Qianlong Li, Chen Huang, Shuai Li, Yuanxin Xiang, Deng Xiong, and Wenqiang Lei. 2024a. Graphotter: Evolving llm-based graph reasoning for complex table question answering. *CoRR*, abs/2412.01230.

Zhenyu Li, Xiuxing Li, Sunqi Fan, and Jianyong Wang. 2024b. Optimization techniques for unsupervised complex table reasoning via self-training framework. *IEEE Trans. Knowl. Data Eng.*, 36(12):8996–9010.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. 2024. Llava-next: Improved reasoning, ocr, and world knowledge.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-eval: NLG evaluation using gpt-4 with better human alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 2511–2522. Association for Computational Linguistics.

Xinyuan Lu, Liangming Pan, Qian Liu, Preslav Nakov, and Min-Yen Kan. 2023. SCITAB: A challenging benchmark for compositional reasoning and claim verification on scientific tables. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 7787–7813. Association for Computational Linguistics.

Zeyao Ma, Bohan Zhang, Jing Zhang, Jifan Yu, Xiaokang Zhang, Xiaohan Zhang, Sijia Luo, Xi Wang, and Jie Tang. 2024. Spreadsheetbench: Towards challenging real world spreadsheet manipulation. *CoRR*, abs/2406.14991.

Thomas Müller, Francesco Piccinno, Peter Shaw, Massimo Nicosia, and Yasemin Altun. 2019. Answering conversational questions on structured data without logical forms. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 5901–5909. Association for Computational Linguistics.

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.

Giang Nguyen, Ivan Brugere, Shubham Sharma, Sanjay Kariyappa, Anh Totti Nguyen, and Freddy Lécué. 2024. Interpretable llm-based table question answering. *CoRR*, abs/2412.12386.

OpenAI. 2023. GPT-4 technical report. *CoRR*, abs/2303.08774.

OpenAI. 2024. Introducing openai o1-preview: A new series of reasoning models for solving hard problems. available now.

Ankur P. Parikh, Xuezhi Wang, Sebastian Gehrmann, Manaal Faruqui, Bhuwan Dhingra, Diyi Yang, and Dipanjan Das. 2020. Totto: A controlled table-to-text generation dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 1173–1186. Association for Computational Linguistics.

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.

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H. Chi, Nathanael Schärli, and Denny Zhou. 2023. Large language models can be easily distracted by irrelevant context. In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 31210–31227. PMLR.

Ananya Singha, José Cambronero, Sumit Gulwani, Vu Le, and Chris Parnin. 2023. Tabular representation, noisy operators, and impacts on table structure understanding tasks in llms. *CoRR*, abs/2310.10358.

Aofeng Su, Aowen Wang, Chao Ye, Chen Zhou, Ga Zhang, Gang Chen, Guangcheng Zhu, Haobo Wang, Haokai Xu, Hao Chen, Haoze Li, Haoxuan Lan, Jiaming Tian, Jing Yuan, Junbo Zhao, Junlin Zhou, Kaizhe Shou, Liangyu Zha, Lin Long, Liyao Li, Pengzuo Wu, Qi Zhang, Qingyi Huang, Saisai Yang, Tao Zhang, Wentao Ye, Wufang Zhu, Xiaomeng Hu, Xijun Gu, Xinjie Sun, Xiang Li, Yuhang Yang, and Zhiqing Xiao. 2024. Tablegpt2: A large multimodal model with tabular data integration. *CoRR*, abs/2411.02059.

Doubao Team. 2025. Doubao-1.5-pro.

Qwen Team. 2024. Qwq: Reflect deeply on the boundaries of the unknown.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. 2024. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *CoRR*, abs/2409.12191.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large 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*.

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*.

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, et al. 2024a. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

Bohao Yang, Chen Tang, Kun Zhao, Chenghao Xiao, and Chenghua Lin. 2024b. Effective distillation of table-based reasoning ability from llms. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation, LREC/COLING 2024, 20-25 May, 2024, Torino, Italy*, pages 5538–5550. ELRA and ICCL.

Jiabo Ye, Haiyang Xu, Haowei Liu, Anwen Hu, Ming Yan, Qi Qian, Ji Zhang, Fei Huang, and Jingren Zhou. 2024. mplug-owl3: Towards long image-sequence understanding in multi-modal large language models. *CoRR*, abs/2408.04840.

Qinghao Ye, Haiyang Xu, Jiabo Ye, Ming Yan, Anwen Hu, Haowei Liu, Qi Qian, Ji Zhang, Fei Huang, and Jingren Zhou. 2023. mplug-owl2: Revolutionizing multi-modal large language model with modality collaboration. *CoRR*, abs/2311.04257.

Yue Yu, Yuchen Zhuang, Jieyu Zhang, Yu Meng, Alexander J. Ratner, Ranjay Krishna, Jiaming Shen, and Chao Zhang. 2023a. Large language model as attributed training data generator: A tale of diversity and bias. 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*.

Yue Yu, Yuchen Zhuang, Jieyu Zhang, Yu Meng, Alexander J. Ratner, Ranjay Krishna, Jiaming Shen, and Chao Zhang. 2023b. Large language model as attributed training data generator: A tale of diversity and bias. 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*.

Tianshu Zhang, Xiang Yue, Yifei Li, and Huan Sun. 2024. Tablellama: Towards open large generalist models for tables. 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 6024–6044. Association for Computational Linguistics.

Bowen Zhao, Changkai Ji, Yuejie Zhang, Wen He, Yingwen Wang, Qing Wang, Rui Feng, and Xiaobo Zhang. 2023. Large language models are complex table parsers. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 14786–14802. Association for Computational Linguistics.

Yilun Zhao, Lyuhao Chen, Arman Cohan, and Chen Zhao. 2024. Tapera: Enhancing faithfulness and interpretability in long-form table QA by content planning and execution-based reasoning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 12824–12840. Association for Computational Linguistics.

Yilun Zhao, Yunxiang Li, Chenying Li, and Rui Zhang. 2022. Multihiertt: Numerical reasoning over multi hierarchical tabular and textual data. 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 6588–6600. Association for Computational Linguistics.

Mingyu Zheng, Xinwei Feng, Qingyi Si, Qiaoqiao She, Zheng Lin, Wenbin Jiang, and Weiping Wang. 2024. Multimodal table understanding. In *Proceedings of*

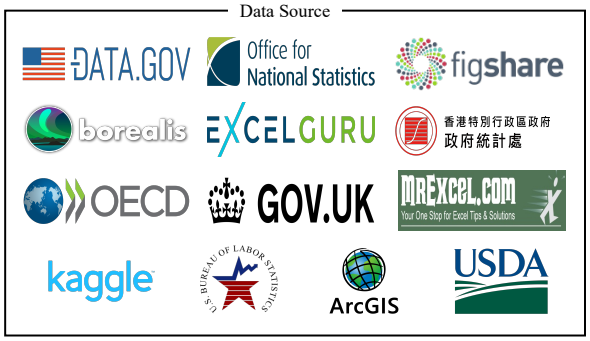

Figure 5: All data sources of tables in RealHiTBench.

the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024, pages 9102–9124. Association for Computational Linguistics.

Yitong Zhou, Mingyue Cheng, Qingyang Mao, Qi Liu, Feiyang Xu, Xin Li, and Enhong Chen. 2024. Enhancing table recognition with vision llms: A benchmark and neighbor-guided toolchain reasoner. *CoRR*, abs/2412.20662.

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, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 3277–3287. Association for Computational Linguistics.

A More Dataset Details

In this section, we explain some details of our dataset. First, we investigate existing open platforms containing tables before collecting the data. Then, to build a unified dataset, we process the collected tables. Finally, we tally up the domains involved in the table content and topics to cover as many real-world application scenarios as possible.

A.1 Data Sources

The tables in our dataset are raw from 13 open platforms. We investigate these platforms in advance to ensure the authority and rationality of the data. Introduction of the sources is as following.

Kaggle. Kaggle is a valuable platform in the field of data science and machine learning. It hosts a variety of competitions that involve solving real-world problems. It also provides a vast repository of datasets across various domains. The data on Kaggle is generally considered to be of high quality and reliable, as it sourced from various places, including public datasets, partner datasets, and user-generated datasets.

USDA.gov. The U.S. Department of Agriculture website provides a wealth of data related to the food, agriculture, and rural sectors. The data are of high quality and reliability, as they follow strict quality standards and are used to inform and enhance public and private decision-making on economic and policy issues. It is an authoritative source for agricultural data.

Catalog.data.gov. This is a catalog of datasets from the US government. It offers a wide range of data resources that are reliable and valuable for research and analysis in various fields. Data are collected and maintained by government agencies, ensuring their authority.

Borealisdata. Borealis, the Canadian Dataverse Repository, is a bilingual, multidisciplinary, secure, Canadian research data repository. It is supported by academic libraries and research institutions across Canada, providing reliable and valuable data for researchers. Data are managed according to the FAIR principles for scientific data management.

Arcgis. Arcgis provides geospatial data and tools for mapping and analysis. Data are reliable and widely used in various industries, such as urban planning, environmental management, and transportation. It offers valuable insights through its geospatial capabilities.

ExcelGuru. This platform focuses on Excel-related data and resources. Although not as comprehensive as some other platforms, it can provide valuable data and tips for Excel users, especially in the context of data analysis and management.

Bureau of Labor Statistics. It is a principal statistical agency of the US Federal Statistical System. The data provided by the Bureau of Labor Statistics are highly reliable and authoritative, covering various aspects of labor economics and statistics,

https://www.kaggle.com/

https://www.usda.gov/
https://catalog.data.gov/
https://borealisdata.ca/
https://hub.arcgis.com/
https://www.excelguru.ca/
https://www.bls.gov/

such as employment, unemployment, inflation, and productivity.

OECD. The Organization for Economic Co-operation and Development (OECD) provides a wide range of data on economic, social, and environmental issues. Data are collected from member countries and are of high quality and reliability, making it a valuable resource for policy-making and research.

Gov.UK. The UK government's website provides data on various topics, such as government reference data, statistics, and reports. The data is reliable and authoritative, as it is collected and maintained by government departments.

Censtatd.gov.hk. The Census and Statistics Department of Hong Kong provides data on Hong Kong's economy, population, and social affairs. The data is reliable and valuable for understanding the local situation and making informed decisions.

MrExcel. Similar to ExcelGuru, MrExcel focuses on Excel - related content. It can offer valuable data and insights for Excel users, especially in terms of advanced Excel techniques and applications.

Office for National Statistics (UK). It is the national statistics office of the UK. The data provided by the Office for National Statistics is highly reliable and authoritative, covering a wide range of topics, such as population, economy, and society.

Figshare. Figshare is a repository for research data and outputs. It allows researchers to store, share, and publish their data, making it a valuable resource for the research community. The data on figshare is diverse and can be used for various research purposes.

A.2 Data Formats

With the current development of large language models (LLMs) and Multimodal Large Language Models (MLLMs), it is necessary to explore the performance of textual and visual models and input formats.

https://www.oecd.org/
https://www.gov.uk/
https://www.censtatd.gov.hk/
https://www.mrexcel.com/
https://www.ons.gov.uk/
https://figshare.com/

Textual Formats. In order to explore the performance of textual input formats, we transform our xlsx tables into various textual formats, including LaTeX, HTML, CSV, Markdown. We conduct a preliminary rationality analysis of these formats. Formats like CSV and Markdown use simple delimiters to represent structural information, which are unlikely to obtain complex tabular and supplementary information. And LaTeX and HTML seem capable to comprehensively exhibit abundant information in complex tables. Thus, we conduct experiment (shown in Figure 4) to compare the performance of these textual formats, including CSV and Markdown to be rigorous. The result shows the LaTeX outperforms among these formats. Therefore, we choose LaTeX as our textual input format.

Visual Format. Image-based representations have effectiveness on table reasoning, so we also explore the performance of visual format on the same benchmark as textual formats. We convert Excel tables into PNG image format and provide them to MLLMs along with corresponding prompts. However, we discover a shortcoming of image format of table representations. For some enormous tables in our collections, image format either cannot cover complete data in the table or makes the picture fuzzy in the vision of MLLMs, which makes it much difficult to understand long tables with image formats. Nevertheless, we conduct experiments both on text and image, finding that although the image modality performs worse than the text modality, the image can bring auxiliary improvements to the text in understanding complex tables.

A.3 Data Domains

We are devoted to constructing a realistic benchmark to reflect the complexity of tables in real scenarios. However, open-domain tables typically exhibit much simpler structures than the complex structures found in specific domains. Tables in specific domains commonly obtain characteristics that they contain domain-specific terms and typically not accompanied by domain-specific labeled data. Thus, it is not promising to conduct supervised fine-tuning on models for TableQA tasks (Katsis et al., 2022). Therefore, we cover 24 domains (shown in Figure 6) to align with real-world application scenarios. And we maintain a balance in the number of tables under each domain as much as possible.

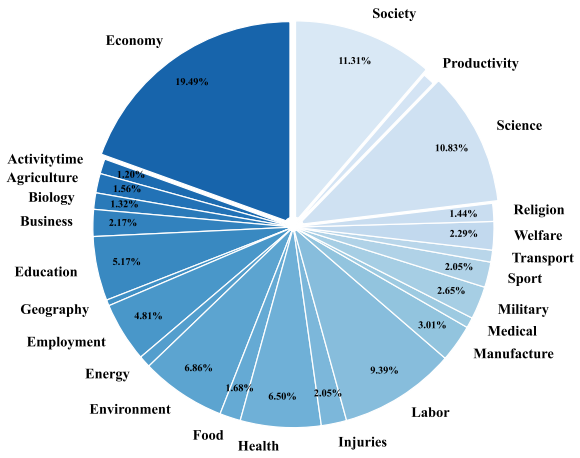

Figure 6: All domains of tables in RealHiTBench.

B More Annotation Details

B.1 Task types

We are devoted to evaluate the performance of LLMs in table understanding and reasoning comprehensively. And table question answering (TableQA) is one of the most common applications about tables. Thus, referring to the previous work (Wu et al., 2024), we introduce 5 primary task types and a couple of more specific subtypes. Here we explain each type to avoid ambiguity.

Fact Checking. Fact Checking in TableQA tasks is the process of verifying the accuracy and consistency of information within a table. It involves assessing whether the data presented in the table aligns with real-world facts or external sources of information. This category focuses on ensuring that the table's content is reliable and free from errors or inconsistencies.

- **Multi-hop Fact Checking** This involves verifying the accuracy of a statement or answer that requires information from multiple cells or rows in a table. It is more complex than single-hop fact checking because it involves cross-referencing different parts of the table to ensure consistency and correctness. For example, verifying the relationship between two different data points in a table would require multi-hop fact checking.

- **Value-Matching** This involves checking whether a specific value or set of values in a table matches the requirements or conditions specified in a question or statement. It is straightforward and focuses on direct comparisons between the values in the table and the values mentioned in the question. For example, verifying the number of employees in a specific department would involve value-matching.

- **Inference-based Fact Checking** This involves verifying the accuracy of an answer that requires logical reasoning or inference based on the information provided in the table. It is more complex than value-matching because it requires drawing conclusions from the data in the table rather than simply matching values. For example, verifying the implications of a certain trend in the data would involve inference-based fact checking.

Numerical Reasoning. Numerical Reasoning in TableQA tasks involves solving problems that require numerical calculations or analyses based on the data in a table. It encompasses tasks that involve arithmetic operations, comparisons, and other numerical computations to derive meaningful insights or answers from the tabular data.

- **Multi-hop Numerical Reasoning** This involves solving numerical problems that require information from multiple cells or rows in a table. It is more complex than single-hop numerical reasoning because it involves cross-referencing different parts of the table to derive the final answer. For example, calculating the total sales of a product across multiple regions would require multi-hop numerical reasoning.

- **Counting** This involves determining the number of occurrences of a specific value or condition in a table. It is straightforward and focuses on counting the instances that meet the specified criteria. For example, counting the number of employees with a specific job title would involve counting.

- **Sorting** This involves arranging the data in a table in a specific order based on a particular attribute or condition. It is used to organize the data in a way that makes it easier to analyze or answer questions. For example, arranging sales data in descending order to identify the top three highest sales figures would involve sorting.

- **Comparison** This involves comparing the values of different cells or rows in a table to determine their relative magnitudes or relationships. It is used to answer questions that require identifying the highest, lowest, or other comparative relationships between data points. For example, determining which product has the highest sales would involve comparison.

- **Calculation** This involves performing mathematical operations on the data in a table to derive a final answer. It is used to answer questions that require arithmetic operations such as addition, subtraction, multiplication, or division. For example, calculating the average sales per region would involve calculation.

Data Analysis. Data Analysis in TableQA tasks is the process of extracting meaningful insights and patterns from the data in a table. It involves tasks such as identifying trends, relationships, and anomalies within the data. This category focuses on understanding the data's underlying structure and using it to answer more complex questions or make predictions.

- **Rudimentary Analysis** This involves performing basic analytical tasks on the data in a table, such as identifying patterns, trends, or simple relationships between data points. It is straightforward and focuses on providing a basic understanding of the data. For example, identifying the most common job title in a department would involve rudimentary analysis.

- **Summary Analysis** This involves providing a concise summary of the data in a table, highlighting key findings or insights. It is used to answer questions that require a brief overview of the data rather than detailed calculations or in-depth analysis. For example, providing a summary of sales performance across different regions would involve summary analysis.

- **Predictive Analysis** This involves using the data in a table to make predictions about future trends or outcomes. It is more complex and often involves statistical or machine learning techniques to model the data and make predictions. For example, making a prediction of future sales based on historical data would involve predictive analysis.

- **Exploratory Analysis** This involves examining the data in a table to identify patterns, relationships, or anomalies that may not be immediately obvious. It is used to answer questions that require a deeper understanding of the data and often involves visualizations or statistical techniques to explore the data. For example, analyzing the relationship between employee performance and job satisfaction would involve exploratory analysis.

- **Anomaly Analysis** This involves identifying and analyzing data points in a table that deviate significantly from the norm or expected values. It is used to answer questions that require identifying outliers or unusual data points that may indicate errors or special cases. For example, identifying any unusual sales figures in a table would involve anomaly analysis.

Chart Generation. Chart Generation in TableQA tasks involves creating visual representations of the data in a table to make it more intuitive and easier to understand. It includes generating various types of charts and graphs, such as line charts, bar charts, scatter charts, and pie charts. This category helps users quickly grasp the key information and trends in the data.

- **LineChart Generation** This involves creating a line chart to visually represent the data in a table, typically to show trends or changes over time. It is used to answer questions that require a clear and intuitive display of how data points change over a period. For example, visualizing sales trends over the past year would involve line chart generation.

- **BarChart Generation** This involves creating a bar chart to visually represent the data in a table, typically to compare the values of different categories or groups. It is used to answer questions that require a clear and intuitive display of the relative magnitudes of different data points. For example, visualizing sales figures across different regions would involve bar chart generation.

- **ScatterChart Generation** This involves creating a scatter chart to visually represent the data in a table, typically to show the relationship between two variables. It is used to an-

swer questions that require a clear and intuitive display of the correlation or pattern between two sets of data points. For example, visualizing the relationship between employee performance and job satisfaction would involve scatter chart generation.

- **PieChart Generation** This involves creating a pie chart to visually represent the data in a table, typically to show the proportion of each category or group relative to the whole. It is used to answer questions that require a clear and intuitive display of the distribution of data points across different categories. For example, visualizing the distribution of employees across different departments would involve pie chart generation.

Structure Comprehending. Structure comprehending involves providing a new table that is created by exchanging some complex parts of the source table and then asking the model the same question with two similar tabular inputs. The goal is to evaluate whether the model can accurately comprehend the structural changes and provide the correct answer. This type of task is particularly challenging because it requires the model to not only understand the data but also the intricate relationships and hierarchies within the table structure.

B.2 QA Annotation

Complex tables alone cannot fully present the challenges in RealHiTBench. Therefore, we meticulously design an annotation process that is suited for question answering tasks on complex tables. Specifically, we apply LLMs to initial question answering (QA) generation to increase annotation efficiency, and we keep the accuracy of annotations by careful multi-turn human check. The complete process diagram is depicted as Figure 7, and introduced as below:

Data Collection and Process. Initially, we collect tabular data from various sources. Then we divide the tables into different categories according to their structures. To tables with complex structures, we process tables before accepting them. Specifically, first we check whether table content contains no improper information. When content is done, we adjust the unrestricted structures to emphasize the complex parts. Because we collect tables as xlsx format, which allows a file to contain multiple sheets, we conduct sheet division to keep the needed table only. Finally, we convert the source table into chosen textual format LaTeX and visual format PNG (The reasons to choose these two formats have been clarified in Appendix 4). Then a qualified table is added into our dataset.

Question Generation. In order to display the complexity in our dataset comprehensively, we introduce LLM to assist in generating annotations. Before generating questions, we interact with GPT to acquire some inspirations about the attribute dimensions of question that are vital for constructing tasks in tabular reasoning (Yu et al., 2023b). We select some points from the GPT answers, including *Complexity*, *Diversity*, *Real-World Relevance*, and *Length*. Then, we design our prompt templates from these dimensions (shown in Table 9). Moreover, we also design different prompt templates for each task type to ensure highlighting the characteristics of each type. Eventually, we combine the prompt template with the exact question type and table content (xlsx format to simulate human intuitive vision) as input for GPT, acquiring the generated answer.

Question Review. Referring to the previous work, the LLM-assisted annotation is likely to lead to the bias or instability of benchmark (Felkner et al., 2024). Therefore, we set question review step to ensure the rationality of questions generated by GPT. We set four attributes of a reasonable question for annotators. Then we provide the tables and questions along with these attributes as review standard to annotators to confirm rationality or adjust the questions.

Answer Annotation with Verification. Similarly, we also design a prompt template for answer generation (shown in Table 13). We combine the prompt template, table content (LaTeX format to simulate careful observation), and question as initial input for models. Then we feed these input to various models and acquire some answers. At the same time, the occurrence times of each answer can be automatically counted as their frequency. Then we input all the generated answers along with initial answers into GPT-Eval module, and gain the score of each answer. We multiply each score by the occurrence frequency of the answer to obtain a final score. The answer with the highest final score becomes the candidate answer. Finally, the candidate answer is provided to annotators for human check and the corrected final answer can be filled

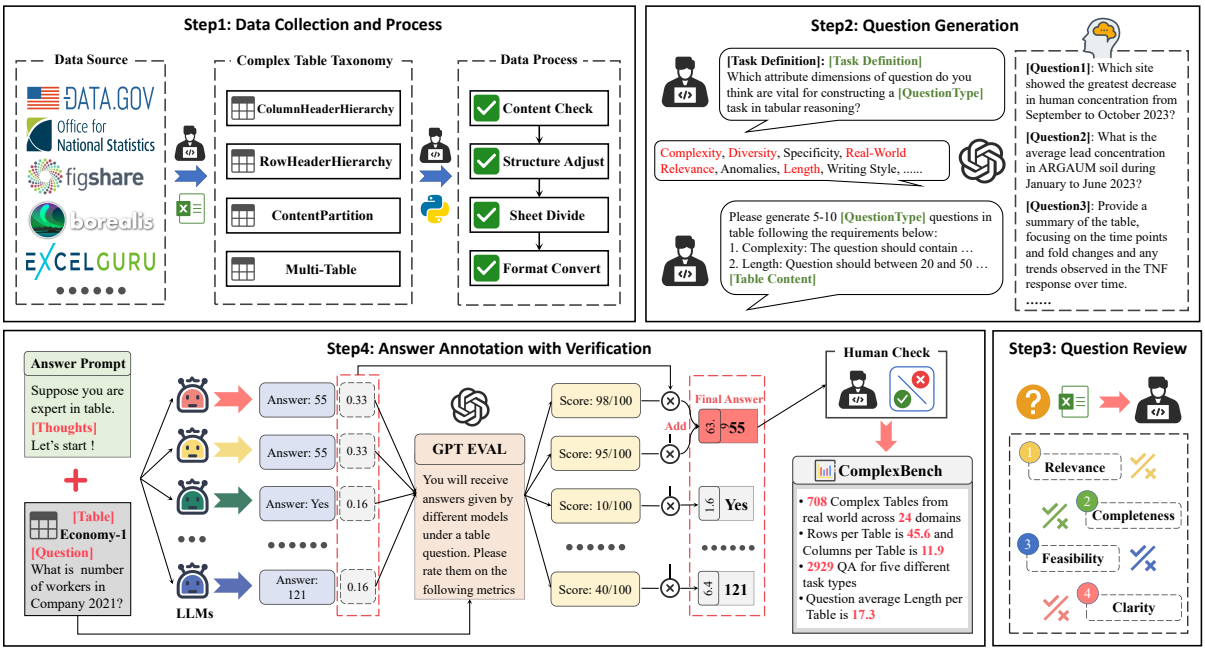

Figure 7: Process diagram of our dataset construction process.

into the annotation field.

B.3 QA prompt

Question Prompt. We introduce question prompts to generate diverse questions with GPT. Considering the uniqueness of different task types, we meticulously design different prompts for each type according to corresponding attribute dimensions (have been explained above). we provide our complete prompts from Table 9 to Table 12.

Answer Prompt. We also introduce answer prompts to generate various questions according to different task type with GPT. Apart from the fundamental prompt parts, such as *Role Play* and *Output Control*, we apply **COT** (Wei et al., 2022) to the answer prompt to improve the capability of understanding. All answer prompts are presented from Table 13 to Table 16. Notably, because the answer content and formats obviously vary among different subtypes of Data Analysis, we design detailed answer prompt for each subtype (Here we just display the prompt of rudimentary analysis.).

Tree-Based Answer Prompt. We introduce TreeThinker 4, a two-round prompt pipeline that enhances the model's ability in table understanding, whose performance is shown the experimental results above. Here, we provide the detailed prompts in Table 17 and Table 18, respectively.

C More Implementation Details

C.1 Omit Long Tables

We collect as much difficult tables as we can so that some enormous tables are in our dataset. However, considering the structural complexity, we are also devoted to multimodal exploration. Unfortunately, it seems impossible to input some enormous tables in realistic applications into MLLMs because of the oversize. Therefore, we omit long tables in our experiment, but we still keep them for future use.

C.2 Chart Generation

As shown in Table 2, Chart Generation is too difficult, leading to the PASS@1 results for some models being 0. We investigate executable python codes generated by models, finding that *DataFrame*, a common python module, exists in most of these codes. However, we find it difficult for python module *DataFrame* to represent hierarchical table structures in detail, obscuring models from meaningful hierarchical information. Finally, non-functional codes are generated.

C.3 GPT-Eval Prompts

As demonstrated above, we utilize GPT-Eval to assess the reliability of answers to some open questions (in the Data Analysis task). Thus, we also design a series of prompts to evaluate different task types. The example prompt is presented in Table 19, which is a representative of the GPT-Eval

Table 5: The more results of advanced models with Text and Image Inputs on RealHiTBench.

Model	Method	Fact Checking		Numerical Reasoning		Structure Comprehending		Data Analysis		Chart Generation	
		F1	EM	F1	EM	F1	EM	ROUGE	GPT-EVAL	ECR	PASS
QwQ-32B	Text	28.95	43.13	17.94	33.57	12.99	37.96	68.66	20.86	31.25	12.5
Doubao-1.5-pro-32k	Text	59.65	66.34	45.4	53.77	65.9	72.45	75.21	36.01	25.97	14.94
DeepSeek-R1-Distill-Qwen-1.5B	Text	11.18	17.75	7.13	15.74	9.92	19.93	38.02	16.24	14.94	0
DeepSeek-R1-Distill-Qwen-7B	Text	28.1	34.53	17.64	24.96	19.08	28.1	54.39	25.19	**39.61**	6.49
DeepSeek-R1-Distill-Llama-8B	Text	22.43	33.29	15.18	24.64	14.76	26.66	54.55	26.98	12.99	5.19
mPLUG-Owl2-7B	Image	3.46	7.23	1.15	4.72	9.14	14.21	25.46	12.13	1.53	0
Qwen2-VL-7B-Instruct	Image	28.5	38.34	9.34	22.88	45.29	55.99	37.41	24.3	16.23	5.84
Qwen2-VL-7B-Instruct	Image+Text	32.79	45.19	11.54	24.97	48.6	60.42	39.69	25.39	25.32	9.09

Table 6: The evaluation results of reasoning models on RealHiTBench.

Models	Method	Avg Score				
		FC	NR	SC	DA	CG
GPT-o1	CoT	70.61	63.08	78.72	54.02	28.73
	TT	75.83	65.63	81.92	56.50	66.67
DeepSeek-R1	CoT	75.18	71.43	83.67	61.07	19.64
	TT	79.18	71.23	84.61	68.94	61.91
QwQ-32B	CoT	41.79	23.12	28.83	47.39	13.27
	TT	47.77	41.235	47.39	49.69	35.72

prompts due to the similarity of these subtypes.

D More Experiments

D.1 Experiments on Reasoning Models.

We evaluated the performance of the most advanced inference models, GPT-o1, DeepSeek-R1, and QWQ on RealHitBench. Considering the high computational cost of GPT-o1 and the recent instability of DeepSeek's server due to a surge in traffic, we randomly selected 500 samples from benchmark to assess the reasoning models' ability to understand realistic hierarchical complex tables, with the results shown in the figure 6. Compared to general-purpose models, inference-based models perform better in handling complex tables. Moreover, our proposed TreeThinker further enhances their ability to process tabular data.

D.2 Different size and complexity table.

We evaluated the overall performance of the model under varying table token counts and different levels of table complexity, as shown in the figure 8. The results indicate that as the number of tokens increases and the table complexity rises, the model's performance gradually declines. Moreover, when these two factors are combined, the challenge for the model becomes even greater.

D.3 TreeThinker on open-source models.

As shown in the table 7, we evaluated TreeThinker on open-source models, specifically Llama3.1-8B-Instruct and Llama3.3-70B-Instruct, across multiple reasoning and analysis tasks. The results demonstrate consistent improvements over the CoT baseline in all tasks. Notably, TreeThinker significantly enhances model's ability in Chart Generation, with ECR rising from 50.65 to 76.62 and PASS from 24.03 to 35.71 in Llama3.3-70B-Instruct. These results highlight TreeThinker's effectiveness in enhancing open-source models across a diverse range of tasks. Regardless of model size, TreeThinker consistently improves performance in various reasoning and analytical tasks, demonstrating its effectiveness in boosting the comprehension and processing of complex hierarchical data.

E Case Study

E.1 Complex Tables Examples

To better illustrate the complexity of each table structure, we provide examples from our dataset, including tables and corresponding question answering (QA) pairs. By the way, to make the table structure more intuitive, we appropriately process the table in the dataset to highlight its complex structures.

Hierarchical Column Header. The specific example is presented in Figure 9. Merged cells, such as 'Landings into' and 'Scotland', present the Hierarchical Column Header structure.

Hierarchical Row Header. Hierarchy structure in row headers is depicted in Figure 10, which shows the type that the classification is not in the same column. The other type, classification shown in the same column, is displayed in Figure 1.

Table 7: The results of TreeThinker when using Llama3.1-8B-Instruct and Llama3.3-70b-Instruct as the base models.

Model	Method	Fact Checking		Numerical Reasoning		Structure Comprehending		Data Analysis		Chart Generation	
		F1	EM	F1	EM	F1	EM	ROUGE	GPT-EVAL	ECR	PASS
Llama3.1-8B-Instruct	CoT	44.93	30.32	27.21	14.53	50.8	35.9	32.25	60.12	13.64	4.55
	TT	47.4	39.85	37.69	29.18	52.17	44.27	34.62	61.69	35.06	27.27
Llama3.3-70B-Instruct	CoT	64.53	53.08	48.99	36.58	68.93	55.81	27.98	52.26	50.65	24.03
	TT	69.94	63.93	61.69	54.6	73.22	64.89	37.38	70.57	76.62	35.71

Nested Sub-Tables. Just as Figure 11, the whole table content is divided into several sub-tables by merged cells spanning the full width of the table. Notably, the content in sub-table 'TOTAL' is the superset of the other parts, while the data in area 'Men, 16 years and over' are the superset of the segments below. This trick sometimes causes ambiguities.

Multi-Table Join. We prepare Figure 12 as the example of Multi-Table Join. Actually, columns headers of the table consist of 5 cells, which look like 10 cells though. The table-maker present the tabular data like this is to create intuitive comparison between 'C1-C10' and 'C11-C20'.

Table 8: Comparison between RealHiTBench and other existing related benchmarks across two aspects: table size and complexity score.

Benchmark	TableBench	MULTIHIERTT	HiTab	RealHiTBench
Table Size				
70th Pctl Row	19	10	23	**69**
70th Pctl Column	7	5	13	**16**
70th Pctl Cell Length	7.52	11.92	7.30	**9.59**
Complexity Score				
Header Hierarchy	27.30	38.62	57.91	**60.55**
Content Partition	19.45	32.16	47.10	**57.54**
Implicit Multi-Table	12.34	21.52	33.98	**43.48**
Supplement Information	32.59	37.26	55.64	**69.86**

[1] 70th Pctl: 70th Percentile

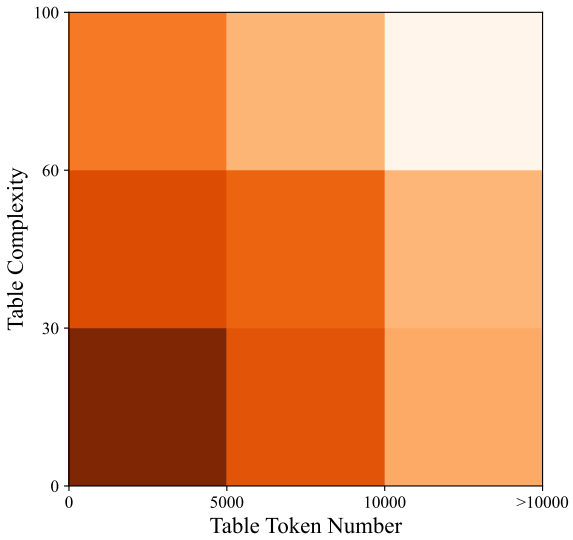

Figure 8: Different complexity and tokens of tables comparsion in RealHiTBench.

Table 9: The complete question prompt of **Fact Checking**. Fill the corresponding content into the '...' positions after '[]' field.

Question Generation Prompt for Fact Checking
Role play Suppose you are an expert in question annotation and your task is to generate high-quality and diverse questions based on a given task description and tabular data.
Task Descriptions Fact checking task in tabular data involves locating relevant data from a table based on a question posed by a user and performing the necessary lookups, calculations, or comparisons to verify or generate an accurate answer. The task typically involves extracting keywords from the question, locating specific rows and columns of the table, and performing calculations, aggregations, or conditional filtering of the data, if necessary, to produce a result that meets the requirements of the question.
Generation Restrictions To generate 5 questions based on the given task description and tabular data, give due consideration to the following aspects: 1.Complexity: Include a range of problem complexities to suit different levels of fact-checking skill. Simple problems should involve straightforward calculations or direct comparisons, like checking totals or averages, while complex questions might involve multi-step calculations or inferential reasoning that requires synthesizing data across several rows or columns. This diversity in complexity ensures that users encounter a mix of questions, from basic checks to challenging analytical tasks, enhancing skill development. 2.Length: Questions should be between 20 and 50 words in length. Ensure that the questions are as concise as possible while still containing the necessary details. 3.Diversity: The subtypes covered by the question should contain Multi-hop Fact Checking, Value-Matching, Inference-based Fact-Checking. 4.Real-World Relevance: The questions asked should match real-world scenarios, making them more practical and relatable. 5.Writing-Style: Use straightforward and accessible language. Keep terminology consistent, especially for statistical or financial terms, to avoid confusion. Ensure questions are free from ambiguity and clearly specify what is expected in the answer. 6.Answer Control: Ensure that the final answer format of the question can be expressed as "Final Answer: AnswerName1, AnswerName2..." form, no other form. Ensure the "AnswerName" is a number or entity name, as short as possible, without any explanation.
Output Control Please generate it in the following format: [Question1]: ... , [subtype1]: ... . [Question2]: ... , [subtype2]: ... . The subtype of each question should be one of Multi-hop Fact Checking, Value-Matching, Inference-based Fact-Checking.

Table 10: The complete question prompt of **Numerical Reasoning**. Fill the corresponding content into the '...' positions after '[]' field.

Question Generation Prompt for Numerical Reasoning
Role play Suppose you are an expert in question annotation and your task is to generate high-quality and diverse questions based on a given task description and tabular data.
Task Descriptions Numerical reasoning tasks in tabular data are tasks that involve analyzing, calculating, or making logical inferences based on numerical information from tabular-structured data. The task typically involves recognizing patterns, making numerical comparisons, identifying trends, predicting outcomes, or solving specific problems from large amounts of numerical data.
Generation Restrictions To generate 5 questions based on the given task description and tabular data, give due consideration to the following aspects: 1.Complexity: Include a range of problem complexities. Simple problems may require direct calculations or basic comparisons, while complex ones could involve multi-step calculations or inferential reasoning across multiple rows/columns. 2.Length: Questions should be between 20 and 50 words in length. Ensure that the questions are as concise as possible while still containing the necessary details. 3.Diversity: The subtypes covered by the question should contain Multi-hop Numerical Reasoing, Counting, Ranking, Comparison, Calculation (Numerical Calculation and Time-based Calculation). 4.Real-World Relevance: The questions asked should match real-world scenarios, making them more practical and relatable. 5.Writing-Style: Use straightforward and accessible language. Keep terminology consistent, especially for statistical or financial terms, to avoid confusion. Ensure questions are free from ambiguity and clearly specify what is expected in the answer. 6.Answer Control: Ensure that the final answer format of the question can be expressed as "Final Answer: AnswerName1, AnswerName2..." form, no other form. Ensure the "AnswerName" is a number or entity name, as short as possible, without any explanation.
Output Control Please generate it in the following format: [Question1]: ... , [subtype1]: ... . [Question2]: ... , [subtype2]: ... . The subtype of each question should be one of Multi-hop Numerical Reasoning, Counting, Ranking, Comparison, Calculation.

Table 11: The complete question prompt of **Data Analysis**. Fill the corresponding content into the '...' positions after '[]' field.

Question Generation Prompt for Data Analysis

Role play
Suppose you are an expert in question annotation and your task is to generate high-quality and diverse questions based on a given task description and tabular data.

Task Descriptions
The task of Data Analysis in tabular data is the process of systematically examining and transforming structured data to extract meaningful information, discover patterns in the data, support decision making, or test hypotheses. It aims to mine key features or trends from data, reveal hidden relationships or anomalies, predict future changes, and provide solutions to problems through quantitative or qualitative methods. This process typically outputs results in a clear and actionable form, providing a solid foundation for deeper understanding of the data.

Generation Restrictions
To generate 5 questions based on the given task description and tabular data, give due consideration to the following aspects:
1.Complexity: Include a range of problem complexities to suit different levels of data analysis skill. Simple tasks should involve straightforward operations, such as calculating averages, identifying basic trends (e.g., increasing or decreasing patterns), or performing direct comparisons between values in a single column. Intermediate tasks might require analyzing relationships between variables using correlation or group-level aggregations, identifying potential impacts, or summarizing data patterns across multiple rows or columns. Complex tasks could involve multi-step processes such as performing causal analysis, anomaly detection requiring cross-referencing multiple datasets, or building predictive models that integrate trends and relationships.
2.Length: Questions should be between 20 and 50 words in length. Ensure that the questions are as concise as possible while still containing the necessary details.
3.Diversity: The subtypes covered by the question should contain Rudimentary Analysis, Summary Analysis, Predictive Analysis, Exploratory Analysis, Anomaly Analysis.
4.Real-World Relevance: The questions asked should match real-world scenarios, making them more practical and relatable.
5.Writing-Style: Use straightforward and accessible language. Keep terminology consistent, especially for statistical or financial terms, to avoid confusion. Ensure questions are free from ambiguity and clearly specify what is expected in the answer.
6.Question example: Rudimentary Analysis: "What is the mean and standard deviation of the Year built column?", "Which state or region has the highest proportion of Military MPs to total MPs, and what is the percentage?". Summary analysis: "Can you provide a descriptive explanation of the table, including the main columns and some basic insights?", "Can you provide a detailed description of the table, including explanations for each main column and highlight any notable trends or insights from the data?". Predictive Analysis: "Based on the historical population growth from 1956 to 2006, what could be the projected population of Tabriz in 2026?". Exploratory Analysis: "How does the number of examinees affect the pass percentage over the years?", "Does a higher crude birth rate causally influence the natural change in population?". Anomaly Analysis: "What are the anomalies in the viewership data for the TV episodes?", "Can you identify which surname data points deviate significantly from the norm?".

Output Control
Please generate it in the following format:
[Question1]: ... , [subtype1]: ... .
[Question2]: ... , [subtype2]: ... .
The subtype of each question should be one of Rudimentary Analysis, Summary Analysis, Predictive Analysis, Exploratory Analysis, Anomaly Analysis.

Table 12: The complete question prompt of **Chart Generation**. Fill the corresponding content into the '...' positions after '[]' field.

Question Generation Prompt for Chart Generation
Role play Suppose you are an expert in question annotation and your task is to generate high-quality and diverse questions based on a given task description and tabular data.
Task Descriptions The task of chart generation in tables is to help users quickly understand and analyze data by converting tabular data into charts or other graphical representations that visualize relationships, trends, or characteristics of the data. The goal is to simplify complex data patterns so that distributions, trends, comparative differences, or correlations between variables can be more easily perceived. Typically, this type of task requires clarifying the analysis objectives, choosing the appropriate chart type (e.g., bar charts, line charts, pie charts, or scatter plots, etc.), and cleaning and processing the data to ensure that the final visualization results are clear, accurate, and able to convey information effectively. This approach not only enhances the intuition and efficiency of data presentation, but also facilitates further analysis and decision-making.
Generation Restrictions To generate 5 questions based on the given task description and tabular data, give due consideration to the following aspects: 1.Complexity: Include a range of problem complexities. Simple problems may only require extracting data and drawing a picture, while complex problems require understanding the data items that need to be used in the problem and processing them before drawing the picture. 2.Length: Questions should be between 20 and 50 words in length. Ensure that the questions are as concise as possible while still containing the necessary details. 3.Diversity: The subtypes covered by the question should contain LineChart Generation, BarChart Generation, ScatterChart Generation, PieChart Generation. 4.Real-World Relevance: The questions asked should match real-world scenarios, making them more practical and relatable. 5.Writing-Style: Use straightforward and accessible language. Keep terminology consistent, especially for statistical or financial terms, to avoid confusion. Ensure questions are free from ambiguity and clearly specify what is expected in the answer. It needs to be made clear in the question what type of chart is being plotted, such as "According to the table, draw a bar chart to illustrate ..." and "Please help me draw a line chart showing ...". 6.Answer Control: Ensure the final answer format is the python code block that can generate the chart correctly.
Output Control Please generate it in the following format: [Question1]: ... , [subtype1]: ... . [Question2]: ... , [subtype2]: ... . The subtype of each question should be one of LineChart Generation, BarChart Generation, ScatterChart Generation, PieChart Generation.

Table 13: The complete answer prompt of **Fact Checking**. Fill the corresponding content into the '...' positions after '[]' field.

Answer Generation Prompt for Fact Checking
Role play Suppose you are an expert in table analysis and your task is to provide answers to questions based on the content of the table.
Chain-of-Thought Let's think step by step as follows and make the most of your strengths as a table analysis expert: 1.Fully understand the question and extract the necessary information from it. 2.Clearly and comprehensively understanding the content of the table, including the structure of the table, the meaning and formatting of each row and column header(Note: There is usually summative cell in the table, such as all, combine, total, sum, average, mean, etc. Please pay careful attention to the flag information in the row header and column header, this information can help you to skip many operations.) 3.Based on the question, select the row and column headers in the table that are most relevant to it and find the corresponding cells based on them. 4.According to the requirements of the question, perform statistical, calculation, ranking, or other operations on the cells you selected, and output of the answer in the format specified by the definition.
Output Control 1.First, you need to output your reasoning steps according to the question and table itself. The reasoning steps should follow the format below: [Reasoning steps for this question are as following: 1.First, we need to... 2.We need to...]. output steps until final answers get solved. 2.Then, you need to output the final answer. The final answer should follow the format below: [Answer Format] Final Answer: AnswerName1, AnswerName2... Ensure the final answer format is the last output line and can only be in the "Final Answer: AnswerName1, AnswerName2..." form, no other form. 3.Ensure the "AnswerName" is a number or entity name, as short as possible, without any explanation. Give the final answer to the question directly without any explanation. If the question is judgmental, please answer 'Yes' or 'No'. Let's get start! [Question]: ...

Stock	Landings into					
	Scotland			England, Wales & N.I.		
	2021	2022	% change	2021	2022	% change
Deep Sea						
Tusk IV	79.5	72.8	-8.4	0.2	0.3	60.5
Ling IV	3139.0	2954.5	-5.9	328.0	315.0	-4.0
Tusk V,VI,VII	138.5	123.9	-10.6	1.4	1.1	-22.5
Ling VI-X,XII,XIV	2171.7	1701.8	-21.6	366.9	226.4	-38.3
Black Scabbardfish V-VII,XII	33.9	4.9	-85.6	0.0	0.0	-100.0
Greater Silver Smelt V-VII	0.0	0.0	-	0.0	0.0	-
Roundnose/Roughead Grenadie	11.1	16.0	43.9	0.0	0.0	-
Blue Ling Vb,VI,VII	1761.9	2123.1	20.5	0.0	0.0	-
Deep sea Shark V-IX	0.0	0.0	-	0.0	0.0	-
Forkbeard V,VI,VII	93.6	67.0	-28.5	0.0	0.0	-

Figure 9: The **Hierarchical Column Header** sample in RealHiTBench.

Table 14: The complete answer prompt of **Numerical Reasoning**. Fill the corresponding content into the '...' positions after '[]' field.

Answer Generation Prompt for Numerical Reasoning
Role play Suppose you are an expert in table analysis and your task is to provide answers to questions based on the content of the table.
Chain-of-Thought Let's think step by step as follows and make the most of your strengths as a table analysis expert: 1.Fully understand the question and extract the necessary information from it. 2.Clearly and comprehensively understanding the content of the table, including the structure of the table, the meaning and formatting of each row and column header(Note: There is usually summative cell in the table, such as all, combine, total, sum, average, mean, etc. Please pay careful attention to the flag information in the row header and column header, this information can help you to skip many operations.) 3.Based on the question, select the row and column headers in the table that are most relevant to it and find the corresponding cells based on them. 4.According to the requirements of the question, perform statistical, calculation, ranking, or other operations on the cells you selected, and output of the answer in the format specified by the definition.
Output Control 1.First, you need to output your reasoning steps according to the question and table itself. The reasoning steps should follow the format below: [Reasoning steps for this question are as following: 1.First, we need to... 2.We need to...]. output steps until final answers get solved. 2.Then, you need to output the final answer. The final answer should follow the format below: [Answer Format] Final Answer: AnswerName1, AnswerName2... Ensure the final answer format is the last output line and can only be in the "Final Answer: AnswerName1, AnswerName2..." form, no other form. 3.Ensure the "AnswerName" is a number or entity name, as short as possible, without any explanation. Give the final answer to the question directly without any explanation. Note: If the final answer has multiple decimals, retain two decimals. Let's get start! [Question]: ...

Table 15: The complete answer prompt of **Data Analysis**. Fill the corresponding content into the '...' positions after '[]' field.

Answer Generation Prompt for Data Analysis
Role play Suppose you are an expert in table analysis and your task is to provide answers to questions based on the content of the table.
Chain-of-Thought Let's think step by step as follows and make the most of your strengths as a table analysis expert: 1.Fully understand the question and extract the necessary information from it. 2.Clearly and comprehensively understanding the content of the table, including the structure of the table, the meaning and formatting of each row and column header(Note: There is usually summative cell in the table, such as all, combine, total, sum, average, mean, etc. Please pay careful attention to the flag information in the row header and column header, this information can help you to skip many operations.) 3.Based on the question, select the row and column headers in the table that are most relevant to it and find the corresponding cells based on them. 4.According to the requirements of the question, perform statistical, calculation, ranking, or other operations on the cells you selected, and output of the answer in the format specified by the definition.
Output Control 1.First, you need to output your reasoning steps according to the question and table itself. The reasoning steps should follow the format below: [Reasoning steps for this question are as following: 1.First, we need to... 2.We need to...]. output steps until final answers get solved. 2.Then, you need to output the final answer. The final answer should follow the format below: [Answer Format] Final Answer: AnswerName1, AnswerName2... Ensure the final answer format is the last output line and can only be in the "Final Answer: AnswerName1, AnswerName2..." form, no other form. 3.The "AnswerName" should represent the primary result of the rudimentary analysis, such as a number or an entity name, expressed as concisely as possible. Provide the final answer directly without additional explanation or extra output. Let's get start! [Question]: ...

Table 16: The complete answer prompt of **Chart Generation**. Fill the corresponding content into the '...' positions after '[]' field.

Answer Generation Prompt for Chart Generation
Role play Suppose you are an expert in table analysis and your task is to provide answers to questions based on the content of the table.
Chain-of-Thought Let's think step by step as follows and make the most of your strengths as a table analysis expert: 1.Fully understand the question and extract the necessary information from it. 2.Clearly and comprehensively understanding the content of the table, including the structure of the table, the meaning and formatting of each row and column header(Note: There is usually summative cell in the table, such as all, combine, total, sum, average, mean, etc. Please pay careful attention to the flag information in the row header and column header, this information can help you to skip many operations.) 3.Based on the question, select the row and column headers in the table that are most relevant to it and find the corresponding cells based on them. 4.According to the requirements of the question, perform statistical, calculation, ranking, or other operations on the cells you selected, and output of the answer in the format specified by the definition.
Output Control 1.First, you need to output your [reasoning steps] according to the question and table itself. The reasoning steps should follow the format below: [Reasoning steps for this question are as following: 1.First, we need to... 2.We need to...]. output steps until final answers get solved. Then, you need to output the final answer. 2.The final answer should follow the format below and ensure the first three code lines is exactly the same with the following code block: [Answer Format] python import pandas as pd import matplotlib.pyplot as plt df = pd.read_excel('table.xlsx') ... plt.show(). Ensure the code can generate the chart correctly and output this code block completely. 3.You should take the values needed to draw the chart directly from the table and write them into the code block, e.g. name1: value1, name2: value2... 4.Then transform it into a string with newlines represented as '\n', indents represented as '\t' and no comments. Ensure code block and corresponding string are right. Do NOT output [answer format]. 5.Ensure that the X-axis used for drawing in the code is arranged in ascending alphabetical or numerical order. Ensure the last line in python code can only be "plt.show()", no other from. Give the final answer to the question directly without any explanation. Let's get start! [Question]: ...

Table 17: The **First-Round Tree-Based** answer prompt. Fill the corresponding content into the '...' positions after '[]' field.

First-Round Tree-Based Answer Prompt

 # Role play
You are tasked with performing detailed table analysis. Your task is to generate a hierarchical tree structure for the top-row and left-column headers based on a LaTeX syntax complex table.

 # Task Description
[Reasoning Steps]
Your thought process is as follows:
1.Understand the Table Structure: Provide a comprehensive description of the table, including the various levels of row and column headers and their corresponding meanings. Construct two distinct hierarchical trees: one for the row headers and one for the column headers. Each tree should accurately represent the levels and relationships of the headers.
2.Traverse the Table: Analyze each row and column header to extract its content, indentation, and positions in the table. Identify merged cells and indentation, as they often indicate hierarchical relationships. Determine the parent-child relationships based on these visual cues and arrange the data under the correct parent node in both row and column header trees.
3.Validate the Hierarchical Relationships: Iterate through both the row header tree and column header tree. Verify that the parent-child relationships are accurate and that the nodes are correctly placed within their respective hierarchies.

 # Node Definition
You will be provided with a table in LaTeX format. The table may contain complex structures, such as merged or nested cells. Your task is to encode each node of table header as a tuple T(t1, t2, t3, t4).
The first element t1 indicates it represents row header (R) or column header (C), along with its corresponding level.
The second element t2 and third element t3 represent its start and end positions, while the fourth element t4 contains the value from the table. For example, a tuple (R0, 1, 2, City) indicates that it is a row header (R) at level 0, spanning from row 1 to row 2, with the value City.
Please Convert the table headers to list L=[T1, T2, ...].

 # Tree Generate
1. Divide the tuples list L into groups based on their levels, such that all tuples with the same level are grouped together. Add a special ROOT node for rows and columns, each with a level of "-1".
2. For each tuple A in L. If the start and end positions of A are equal, mark A as a leaf node.
3. Otherwise, compare its T2 and T3 values with every closest higher-level and same flag tuple B. If tuple A is within the range of tuple B, then B is the parent-header of A.
4. Repeat steps 2 and 3 iteratively until all tuples in L are linked to their respective parent nodes (Tuples without parent node are linked to the ROOT node), forming a hierarchical Table-Header Tree H.

 # Output Control
Next, we will provide a table for you to analyze the hierarchical structure for the table and please organize the table header tuples as a tree, which can help you better understand the table structure. You should clearly and comprehensively understand the content of the table, including the structure of the table, the meaning and formatting of each row and column header (Note: There is usually summative cell in the table, such as all, combine, total, sum, average, mean, etc. Please pay careful attention to the flag information in the row header and column header, this information can help you to skip many operations.)
Please check the constructed tree structure carefully and make sure that you have not missed any information in the contents of the table.
Let's get started!
[TABLE]: ...

Table 18: The **Second-Round Tree-Based** answer prompt. Fill the corresponding content into the '...' positions after '[]' field.

Second-Round Tree-Based Answer Prompt
Role play You are a table analyst. Your task is to first extract relevant keywords based on the questions posed, identify related content from the previous tables, and match it with the corresponding headers in the structure tree. Then you need to answer questions based on the provided table content.
Thinking Guidelines and Output Format Control 1.Understand the Question: Begin by carefully reading the question to extract the essential information needed for answering. This helps ensure that you focus on the right aspects of the table in the next steps. 2.Analyze the Table Content: Thoroughly examine the structure tree and original content of the table, paying close attention to both row and column headers, which may include special indicators such as "Total," "Sum," "Average," or other summary metrics. Be mindful of any rows or columns dedicated to aggregates, as these can provide quick answers without the need for detailed calculations. It's also crucial to recognize that the table might have complex structures, such as merged cells or semantic nesting, which could influence the interpretation of the data. 3.Identify Relevant Data: With a clear understanding of the question, identify the rows and columns in the table that are most relevant to the inquiry. This involves locating the cells that correspond to the relevant headers, ensuring the selected data is directly related to the question at hand. 4.Perform Necessary Analysis or Calculations: Once the relevant data is identified, perform any required operations, such as statistical analysis, mathematical calculations, ranking, or other necessary procedures. This will help you derive the needed insights and provide a comprehensive answer.
Output Action Pattern Your output should follow a React-like pattern of thinking, which includes one or more cycles of "Thought/Action/Result", ultimately leading to a "Final Answer" on the last line. [Action Patterns] 1.Thought: Consider the next action based on the result of the previous one. 2.Action: The action should always be a single processing action. 3.Result: Simulate the action result, analyze the result, and decide whether to continue or stop. (This "Thought/Action/Result" cycle can repeat multiple times.) Verify the table, observations, and question thoroughly before providing the final answer.
Output Format When answering, if the final answer comes from the original format in the table, please use the original format from the table without modifying it. Below is an example of an output format. You need to first output the relevant keywords, headers, and content related with the question, then go through a multi-round interaction of thought/action/result, and finally provide the final answer.
Output Example Relevant Keywords: Keywords related to the table in question. Relavant Table Headers: column/row headers related with the question. Relavant Content: related table content. Thought: Your first round of thinking. Action: The action of your first round. Result: The observation and result of your first round of simulation. Thought: Your second round of thinking. The 'Thought/Action/Result' cycle can repeat 1 or more times until the final answer is reached. Action: The action of your second round. Result: The observation and result of your second round of simulation. Final Answer: Your output result, following the format "Final Answer: AnswerName1, AnswerName2...". The "AnswerName" should be a number or entity name, as short as possible. Let's get start! [Question]: ...

Table 19: The representative GPT-Eval prompt of **Data Analysis**. Fill the corresponding content into the '...' positions after '[]' field.

Representative GPT-Eval Prompt for Data Analysis
Suppose you are an expert in table analysis and your task is to rate the user answer on one metric based on the table content, question and corresponding reference answer. You will be given table content and a question about rudimentary analysis of the table. And the corresponding reference answer to a question and the answer from the user. Your task is to rate the answer on one metric. Please make sure you read and understand these instructions carefully. Please keep this document open while reviewing, and refer to it as needed. Evaluation Criteria: Correctness (1-100) - the answer should be as close as possible to the reference answer, with perfectly equal answers receiving full marks, smaller differences receiving higher marks, and larger differences receiving only lower marks. Evaluation Steps: 1. Read the table carefully and fully understand the contents of the table. 2. Read the result and compare it to the reference answer and the table. Determine if the answer is correct and if not score it based on how different it is from the correct answer. 3. Assign a score for correctness on a scale of 0 to 100, where 0 is the lowest and 100 is the highest based on the Evaluation Criteria. [Question]: ... , [Reference Answer]: ... , [User Answer]: ... . Emphasize: you need to make sure your final answer is formatted in this way: [Score]: xx/100

City-wise Sales and Call Metrics by Representative				
City	Name	Sum of Calls	Sum of Units	Sum of Value
Cairo	Drew	9	58	$1,461.60
	Morgan	9	21	$967.47
	Taylor	11	15	$645.45
Lima	Drew	4	7	$327.60
	Morgan	11	18	$571.14
	Taylor	5	3	$120.60
London	Drew	7	18	$502.02
	Morgan	1	0	$0.00
	Taylor	13	27	$1,291.95
Montreal	Drew	8	19	$621.87
	Morgan	2	5	$232.45
	Taylor	10	16	$530.88
New York	Drew	10	23	$901.60
	Morgan	10	11	$381.26
	Taylor	12	10	$458.80
Rome	Drew	6	12	$568.32
	Morgan	16	23	$607.43
	Taylor	16	103	$4,900.74
Shanghai	Drew	5	4	$141.64
	Morgan	15	20	$787.20
	Taylor	20	28	$1,072.40
Grand Total		200	441	$17,092.42

Figure 10: The **Hierarchical Row Header** sample in RealHiTBench.

Employment status, sex, and age	Total		White		Black or AfricanAmerican		Asian	
	2022	2023	2022	2023	2022	2023	2022	2023
TOTAL								
Civilian noninstitutional population	263973	266942	203214	204515	34131	34667	16933	17591
Civilian labor force	164287	167116	125957	127327	21236	21886	10921	11439
Participation rate	62.2	62.6	62	62.3	62.2	63.1	64.5	65
Employed	158291	161037	121908	123165	19937	20674	10615	11096
Employment-population ratio	60	60.3	60	60.2	58.4	59.6	62.7	63.1
Unemployed	5996	6080	4049	4162	1300	1212	306	344
Unemployment rate	3.6	3.6	3.2	3.3	6.1	5.5	2.8	3
Not in labor force	99686	99826	77257	77188	12895	12781	6012	6152
Men, 16 years and over								
Civilian noninstitutional population	128617	130476	100122	101059	15740	16067	8013	8325
Civilian labor force	87421	88877	68162	68892	10259	10543	5775	6057
Participation rate	68	68.1	68.1	68.2	65.2	65.6	72.1	72.8
Employed	84203	85500	65924	66524	9617	9943	5610	5852
Employment-population ratio	65.5	65.5	65.8	65.8	61.1	61.9	70	70.3
Unemployed	3218	3377	2238	2368	642	599	165	204
Unemployment rate	3.7	3.8	3.3	3.4	6.3	5.7	2.9	3.4
Not in labor force	41197	41599	31959	32168	5480	5524	2238	2268
Men, 20 years and over								
Civilian noninstitutional population	119965	121714	93759	94665	14508	14801	7566	7858
Civilian labor force	84276	85683	65738	66407	9866	10159	5671	5955
Participation rate	70.3	70.4	70.1	70.1	68	68.6	74.9	75.8
Employed	81409	82698	63743	64316	9294	9617	5517	5764
Employment-population ratio	67.9	67.9	68	67.9	64.1	65	72.9	73.4
Unemployed	2867	2985	1995	2091	572	542	154	191
Unemployment rate	3.4	3.5	3	3.1	5.8	5.3	2.7	3.2
Not in labor force	35689	36031	28021	28258	4642	4642	1896	1903

Figure 11: The **Nested Sub Tables** sample in RealHiTBench.

Animal	Treatment	Time Open Arms (s)	(Open Distance/Total Distance)*100	Total Distance (m)	Animal	Treatment	Time Open Arms (s)	(Open Distance/Total Distance)*100	Total Distance (m)
C1	Vehicle	118.2	33.76551	5.399	C11	Fluoxetine	139.2	41.23347	8.772
C2	Vehicle	3.4	0.47638	2.519	C12	Fluoxetine	137.8	37.87928	9.261
C3	Vehicle	99.7	26.40384	8.548	C13	Fluoxetine	151.3	32.08281	12.848
C4	Vehicle	161.6	37.16497	12.275	C14	Fluoxetine	123.4	33.05901	9.235
C5	Vehicle	122.6	21.82437	7.345	C15	Fluoxetine	123	33.83007	7.65
C6	Vehicle	106.8	31.84214	7.399	C16	Fluoxetine	120.1	34.86208	8.918
C7	Vehicle	68	24.61115	6.172	C17	Fluoxetine	125.3	27.49749	7.968
C8	Vehicle	162	42.56557	12.2	C18	Fluoxetine	125.7	26.99287	7.991
C9	Vehicle	152.9	46.48288	8.089	C19	Fluoxetine	131	29.02075	8.966
C10	Vehicle	103.1	24.73884	4.212	C20	Fluoxetine	156	44.55694	10.766
average		109.83	28.987565	7.4158	average		133.28	34.101477	9.2375
stdv		47.98631865	12.89433517	3.128689282	stdv		12.47101528	5.775681618	1.543826865
sterror		15.17460635	4.077546806	0.989378422	sterror		3.943681303	1.826430895	0.488200921

Figure 12: The **Multi-Table Join** sample in RealHiTBench.

Year	Gallons Consumed (in millions)	Fuel Expense (in millions)	Average Price Per Gallon	Percentage of Total Operating Expense	Available Seat Miles per Fuel Gallon
2018	4,137	$9,307	$2.25	24%	67
2017	3,978	$6,913	$1.74	20%	66
2016	3,904	$5,813	$1.49	18%	65

Figure 13: The flat table sample in TableBench (Wu et al., 2024).

Head				Management	
Head ID	Name	Born State		Department ID	Head ID
1	Tiger Woods	Alabama		2	5
...				...	
4	Dudley Hart	California		7	1
Department					
Department ID	Name	Creation	Ranking	Num_Employees	
1	State	1789	1	30266	
...					
7	Commerce	1903	7	36000	

Figure 14: The multiple table sample in MMQA (Wu et al., 2025).