ProTrix: Building Models for Planning and Reasoning over Tables with Sentence Context

Anonymous ACL submission

Abstract

Tables play a crucial role in conveying information in various domains, serving as indispensable tools for organizing and presenting data in a structured manner. We propose a *Plan-then-Reason* framework to answer different types of user queries over tables with sentence context. The framework first plans the reasoning paths over the context, then assigns each step to program-based or textual reasoning to reach the final answer. We construct an instruction tuning set `TrixInstruct` following the framework. Our dataset cover queries that are program-unsolvable or need combining information from tables and sentences to obtain planning and reasoning abilities. We present ProTrix by finetuning Llama-2-7B on `TrixInstruct`. Our experiments show that ProTrix generalizes to diverse tabular tasks and achieves comparable performance to GPT-3.5-turbo. We further demonstrate that ProTrix can generate accurate and faithful explanations to answer complex free-form questions. Our work underscores the importance of the planning and reasoning abilities towards a model over tabular tasks with generalizability and interpretability. We will release our dataset and model to the research community.

Athletics at the 2001 Goodwill Games 3000 meters

Rank	Name	Nationality	Time
1	Brahim Boulami	Morocco	8:17.73
2	Reuben Kosgei	Kenya	8:18.63
3	Stephen Cherono	Kenya	8:19.98
4	Bouabdellah Tahri	France	8:20.25
5	Tim Broe	United States	8:20.75
6	Luis Miguel Martín	Spain	8:24.03
7	Raymond Yator	Kenya	8:27.19
8	Thomas Chorny	United States	9:24.26

Sentence Context

Brahim Boulami (born April 20, 1972 in Safi) is a Moroccan athlete who set two world records in the 3,000 meter steeplechase.

Reuben Seroney Kosgei (born 2 August 1979 in Kapcherop, Kenya), is a middle and long distance athlete mostly famous for 3000 m steeplechase in which he became the youngest ever winner of an Olympic gold medal.

Saif Saaeed Shaheen formerly Stephen Cherono (born 15 October 1982), is a steeplechase runner.

How many medals were won by Kenya?

Is this claim true or false?
The silver medalist of the 3,000 meters steeplechase at 2001 Goodwill Games has never won a Olympic gold medal.

How successful is Kenya in the 3000 meters steeplechase at 2001 Goodwill Games?

Figure 1: Demonstrations of user queries to a table in Wikipedia. The table is extracted from the Wikipedia page Athletics at the 2001 Goodwill Games. Some of the sentences with hyperlinks to the table are presented as sentence context.

1 Introduction

Tables, serve as a fundamental tool for organizing and presenting information across various domains. Whether in business reports, or scientific publications, tables are commonly employed to convey complex data effectively. Despite their widespread utility, the process of human beings answering questions involving tables appears to be time-consuming, given the often substantial amount of content involved. Recognizing this challenge, there arises a need to leverage the capabilities of Large Language Models (LLMs) to understand and respond to user query automatically.

Figure 1 demonstrates three kinds of user queries for a table from Wikipedia. In the first example, the user query is *How many medals were won by Kenya?*. This question is annotated as a program-unsolvable question by SQL experts (Shi et al., 2020) attributed to the absence of an explicit column for medals in the table. To resolve this, the model must fill the gap between the query and the table by recalling the common knowledge that *only the top three players can win medals*. The

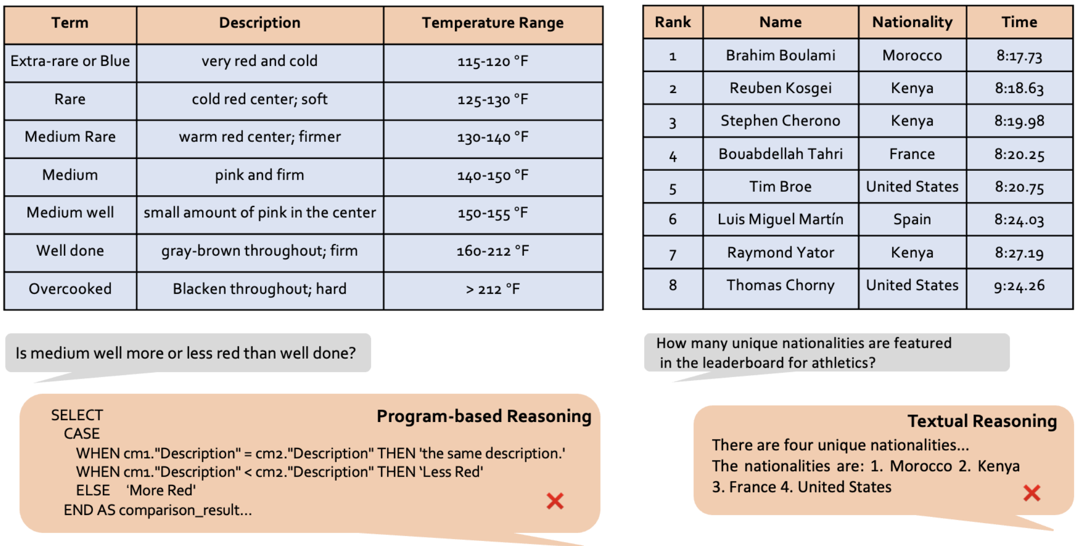

Figure 2: Demonstration of disadvantages of program-based and textual reasoning on tabular tasks. program-based reasoning fails to answer the query since it tries to compare general concepts with a math operator. The textual reasoning fails on a program-solvable query that needs to count distinct countries in the table.

second query delves into a multi-hop scenario asking whether the silver medalist at the 2001 Goodwill Games has ever won an Olympic gold medal. Addressing such queries raises two challenges (1) Decompose the query into sub-tasks. Such as the model plans to derive the silver medalists first and then verify their record of Olympic medals. (2) Combining structured and unstructured context. The model must extract the athletic name from the table and derive the information from the sentence context that Kosgei has won an Olympic gold medal since he is *the youngest ever winner of an Olympic gold medal*. The last query also requires the model to recall common knowledge to decide which contextual information can be used as evidence to judge if *Kenya is successful at the 2001 Goodwill Games*. Subsequently, the model must generate explanations to arrive at certain conclusions. The first two queries mainly require the model to fill the information gap in the query with a short-form answer while the third query seeks for information on a more general concept *how successful is Kenya*. It underscores the importance of the planning and reasoning abilities to connect the general concept with the actual information in the context and generate faithful and accurate explanations for the conclusions.

Various pre-trained models are proposed for tabular tasks (Yin et al., 2020; Wang et al., 2021; Iida et al., 2021; Deng et al., 2022; Yang et al., 2022; Jiang et al., 2022; Liu et al., 2021). But they are often limited to specific query types and could not generalize well to unseen tasks. Regarding models fine-tuned with respect to general tabular querying tasks (Xie et al., 2022; Liu et al., 2023a; Zhang et al., 2023a), they are expected to generate the answers directly, which inevitably lacks interpretability. Previous methods are not specifically designed for enhancing the planning and reasoning abilities of models while these abilities are crucial for building a tabular model with generalizability and interpretability.

In this paper, we propose a *Plan-then-Reason* that *plans* upon various types of user queries and then *reasons* to reach the final answer. We leverage this framework to fine-tune models to enhance planning and reasoning abilities. Recent base models are pre-trained with a large amount of corpora thus obtaining intrinsic common knowledge (Touvron et al., 2023; Roziere et al., 2023). These models suit as the backbone for our models that can fill the gap between queries and tables, understand general concepts, and plan the reasoning paths.

There are generally two ways to enhance reasoning ability. One is textual reasoning which prompts the model to answer questions step-by-step (Wei et al., 2022). The other one is program-based reasoning, prompting the model to write code to answer the questions. Each of the reasoning methods has its disadvantages as shown in Figure 2. The textual reasoning method such as Chain-of-Thought (Wei et al., 2022) can be used to enhance the tabular reasoning ability but often lacks precision in tabular operations such as sorting, counting and filtering, and may not generalize well to large tables (Chen et al., 2019). The program-based reasoning method writes SQL or Python code to answer users' query (Chen et al., 2022). The left example in Figure 2 queries the color comparison between steaks with different cooking methods. Therefore, it raises a need to integrate the advantages of program-based and textual reasoning. The model could write code to extract necessary information from the table or perform specific operations with high precision, which would help the model generalize to unseen or larger tables. And the model could leverage textual reasoning to maintain understanding of general concepts and combine information from table and sentence context to reach final answers or conclusions.

To enhance the planning and reasoning abilities mentioned above, we construct an instruction tuning dataset `TrixInstruct` based on benchmarks with queries that are program-unsolvable or need combining information from table and sentence context. We finetune Llama-2-7B (Touvron et al., 2023) with `TrixInstruct`. The resulting model ProTrix[1] is designed to **P**lan and **R**eason **O**n **T**abula**R** tasks with integration of code execut**I**on and te**X**tual reasoning. Our experiments show that models trained with *Plan-then-Reason* framework can generalize to unseen tabular tasks in different domains with only a handful of training examples and give accurate and faithful explanations even for complex *how* and *why* queries.

In summary, our contributions are listed as:

- We propose a *Plan-then-Reason* framework to answer user queries on tabular tasks with sentence context. The framework first plans the reasoning pathways by ingesting the query and the context, and assigns each step to textual and program-based reasoning to arrive at the final answer.
- We construct `TrixInstruct`, an instruction-tuning set to build models with generalizability and interpretability over tables with sentence context. To obtain the required planning and reasoning abilities, we include queries that are program-unsolvable or need combining tables and sentences in our instruction-tuning dataset.
- We will open-source our model ProTrix, capable of planning and reasoning on tabular tasks with sentence context. ProTrix can generalize to unseen tasks and achieves comparable performance with GPT-3.5-turbo.

[1]Protrix originally means a chemical reactor for small-scale production with compatibility and process control.

2 Our Method

2.1 Problem Formulation

This study centers on tabular tasks with sentence context. Each instance is structured as (Q, T, S, A), where Q represents users' query. T denotes a singular table, while S denotes the sentence context. The sentence context usually is passages linked to the table or retrieved from a knowledge base. Finally A stands for the predicted answer. The answer could be short-form for answering questions like *hom many...* or *is this true or false...*. For *how* and *why* questions, the answer is generally one or more sentences which is defined as free-form answers.

2.2 Plan-then-Reason Framework

We propose a *Plan-then-Reason* framework to build a generalist model that can answer different types of queries over tables and texts. The framework first ingests the query and the context by recalling common knowledge and general concepts. Then it begins to design the model's reasoning pathway, planning the utilization of program-based and textual reasoning to arrive at conclusions.

Planning The model first analyzes the query and fills the potential gap between the query and the context. Consider the first query in Figure 3, there is no explicit column of *color* in the table. The model recalls commonsense that *pink*, *gray-brown* and other colors in the description column can be used to answer the question. Similarly, in the second query, the model recalls that *only top 3 athletes can win medals.*

Then the model adaptively plans the reasoning path with program-based and textual reasoning to address the limitations of each method. For the first query, the model plans to use SQL to extract relevant information from the table and make comparisons of concepts through textual reasoning. For the second query, the model decomposes the task into a multi-hop reasoning chain. It uses SQL to extract the silver medalist from the table and uses sentence context to verify his Olympic record.

Reasoning The reasoning phase initiates with program-based reasoning, writing SQL queries to extract relevant cells or perform operations such as counting and ordering. After running SQL on

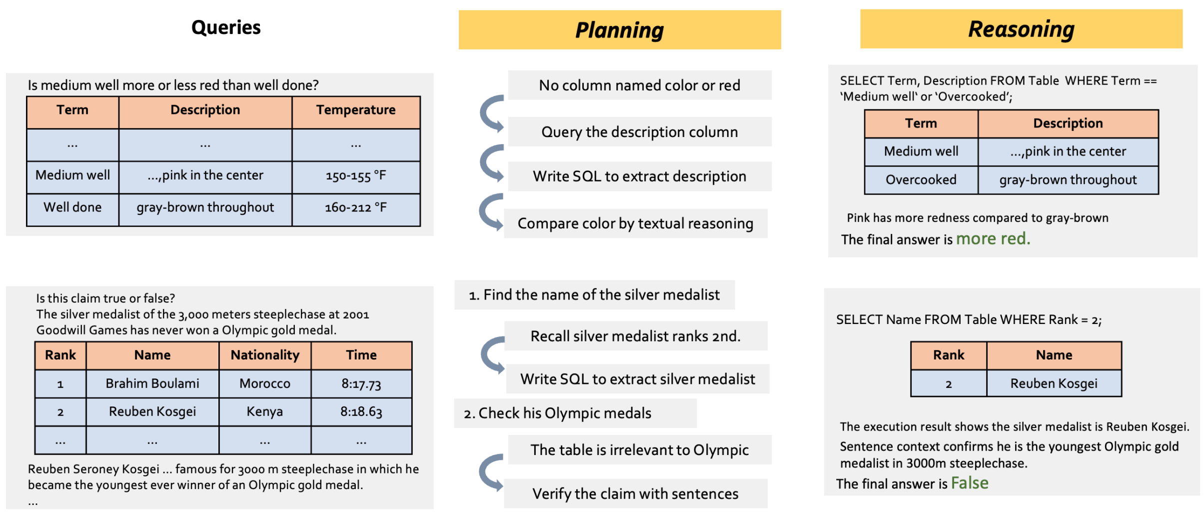

Figure 3: Illustration of our framework. The figure shows the process of our framework to answer a question. The framework first performs strategical planning, decomposing questions into reasoning chains to be solved by either table or sentence context, then perform reasoning based on symbolic programs or natural language to reach answer.

a code interpreter, the results are fed back into the model's ongoing reasoning process. Subsequently, during textual reasoning, the model selects relevant sentences from a pool of retrieved information to complement the table context. *Reuben Kosgei... youngest ever winner of an Olympic gold medal* suggests that he has won a gold medal in his career. At last, the model summarizes insights from the program-based and textual reasoning to reach the answer.

2.3 Instruction Tuning

Based on the analysis in §1, we highlight the abilities *Plan-then-Reason* framework possesses towards tabular tasks with sentence context. (1) Understanding user's query: use parametric knowledge of commonsense insights and general concepts to analyze the relationship between the query and the context; (2) Adaptive planning: decompose a query into sub-questions and plan to answer these sub-questions with different types of context or design multi-hop reasoning paths for the query, and (3) Blending program-based and textual reasoning: synthesize the strength of each method to maximize performance.

To train our model with such abilities, we construct an instruction tuning set `TrixInstruct` based on two tasks i.e., WikiTQ (WikiTableQuestions) (Pasupat and Liang, 2015) and FEVEROUS (Aly et al., 2021). WikiTQ involves a table question-answering task based on a single provided table, requiring multi-step reasoning and performing diverse data operations such as comparison, aggregation, and arithmetic computation. WikiTQ also contains cases that can not be solved by SQL programming (Shi et al., 2020) solely which need to be solved with textual reasoning as the case shown in Figure 2.

On the other hand, FEVEROUS presents an open-domain fact verification challenge spanning both sentences and tables. We select samples containing precisely one table in their gold evidence set. Each case is presented with the corresponding table along with 5 sentences as contextual information. To introduce variability to the sentence context, we ensure the inclusion of gold sentence evidence and augment the context with noisy sentences retrieved from Wikipedia by a dense retriever (Hu et al., 2023). Training examples on claim verification from FEVEROUS impart the ability to decompose claims and reason on each sub-claim with a specific table or sentence context.

For each task, we sampled 4,000 instances from the training datasets. We employ GPT-4 (Achiam et al., 2023) to generate responses according to the *Plan-then-Reason* framework following the prompts detailed in Table 8 and Table 9 in Appendix. We filter out instances that GPT-4 predicts answers inconsistent with the original dataset annotations. This results in a training set comprising 3,517 cases from FEVEROUS and 2,866 cases from WikiTQ. We train our model based on Llama-2-7B (Touvron et al., 2023) and CodeLlama-7B (Roziere et al., 2023).

	WikiTQ	WikiSQL	TabFact	SciTAB	FEVEROUS	HybridQA	TATQA
Closed-Source Model							
GPT-3.5-turbo	51.8	55.0	68.8	45.3	61.0	45.7	59.1
Finetuned SOTA	63.3[†]	89.2[†]	90.8 [†]	73.1[†]	75.9[†]	61.0[†]	74.5[†]
7B Parameter Model							
Llama-2	21.4	17.4	48.6	27.2	47.1	27.6	28.7
CodeLlama	13.1	17.3	49.5	37.1	43.0	28.5	28.4
TableLlama	31.6	41.7	82.6[†]	29.2	56.8	30.7	36.1
ProTrix	56.2[†]	54.0	71.6	45.0	75.6[†]	37.1	39.6
ProTrix-Coder	57.8[†]	60.0	70.6	41.2	71.4[†]	39.3	41.3

Table 1: Experimental results on short question answering and fact verification tasks. Most results are zero-shot performance. [†] The model is trained on the benchmark.

3 Experiments

3.1 Benchmarks for Evaluation

We use existing tabular benchmarks with different input and output configurations to evaluate the performance of our model on queries with short-form or free-form answers. We further divide existing benchmarks on short-form answer tasks into short-form question answering and fact verification following the category in Figure 1.

Short-form Question Answering WikiSQL and WikiTQ are question answering benchmarks on tables from Wikipedia without sentence context (Zhong et al., 2017; Pasupat and Liang, 2015). HybridQA (Chen et al., 2020) requires models to answer questions based on both tables and sentences. We use retrieved sentences, admittedly noisy, from Chen et al. (2020) as the sentence context. TATQA (Zhu et al., 2021) is focused on tables with sentence context from financial reports.

Fact Verification We use fact verification benchmarks to evaluate the performance of answering questions: *is this true or false*. We follow our method in §2.3 to construct the evaluation dataset for FEVEROUS (Aly et al., 2021). TabFact (Chen et al., 2019) verifies claims based on tables from the Wikipedia. SciTAB (Lu et al., 2023) focuses on tables from scientific papers. This benchmark requires compositional reasoning and commonsense knowledge.

Free-form Question Answering FetaQA contains *what* questions with multiple answers and *how* and *why* questions that requires model to generate explanations (Nan et al., 2022). The original FetaQA dataset has annotated highlighted cells, we turn to a more challenging and realistic scenario where the highlighted cells are not provided as input and the model will answer the question directly based on the complete table context. Since our model is only finetuned on short-form answer tasks, FetaQA can be utilized to further evaluate the interpretability and generalizability of our model.

3.2 Short-form Answer Tasks

Baselines We choose the following baselines: (1) Closed-source model: We use the zero-shot end-to-end QA performance on GPT-3.5-turbo as baseline. (2) Finetuned SOTA: We select the fine-tuned SOTA for each task as baselines. We use OmniTab (Jiang et al., 2022) for WikiTQ, TAPEX (Liu et al., 2021) for WikiSQL, PASTA(Gu et al., 2022) for TabFact, finetuned BERT for SciTAB (Lu et al., 2023), S^3HQA (Lei et al., 2023) for HybridQA and APOLLO (Sun et al., 2022) for TATQA . For FEVEROUS, we run DCUF (Hu et al., 2022) on our training and development set of FEVEROUS and obtain an accuracy of 75.9%. Notice that S^3HQA uses a more precise sentence retriever compared to ours. And DCUF leverages an additional retriever to select 25 table cells as input. (3) 7B parameter model: We first compare our model with the zero-shot performance of base models, Llama-2-7B (Touvron et al., 2023) and CodeLlama 7B (Roziere et al., 2023). Then we compare with TableLlama, which is the most similar baseline to our model. TableLlama is a generalist model trained on TableInstruct (Zhang et al., 2023a), a large-scale training set with approximately 260k training examples.

Evaluation Metrics Our model is trained to reach a final answer after a phrase *the answer is*. During the evaluation of question answering and fact verification tasks, we match the gold answer after *the answer is* phrase. If *the answer is* phrase is not found in the response, we consider the answer as wrong. We report three-class F1 score for SciTAB and accuracy for the other datasets.

Main Results The experimental result in Table 1 shows that *our model generalizes to diverse tabular tasks with only 6k training instances.* Our model ProTrix has comparable performance to the closed-source model GPT-3.5-turbo. Compared with the backbone model Llama-2-7B, the performance gain on in-domain benchmarks is 34.8% on WikiTQ and 28.7% on FEVEROUS. And the performance gain on out-of-domain benchmarks is 19.0% on average. Comparing the out-of-domain performance with TableLlama, ProTrix surpasses Tablellama by 14.2% on WikiSQL, 6.4% on HybridQA, 13.3% on SciTAB and 3.5% on TATQA. The overall performance gain on out-of-domain benchmarks demonstrates the planning and reasoning abilities obtained from `TrixInstruct` is not restricted to in-domain queries. The model adaptatively generalizes to queries with different input and output configurations and can even be applied to specific domains such as science and finance.

Comparison Between Base Models We experiment with two different backbone models Llama-2-7B (Touvron et al., 2023) and CodeLlama-7B (Roziere et al., 2023) with two resulting models ProTrix and ProTrix-Coder. From Table 1, we can observe that ProTrix-Coder achieves higher accuracy on question answering tasks while ProTrix has better performance on fact verification tasks. ProTrix-Coder benefits on the reasoning ability from the base model CodeLlama which is continually trained on code but falls short on the planning ability for verification.

Training Cost TableLlama (Zhang et al., 2023a) is the most similar to our model as a generalist model for tabular tasks. Notably, TableLlama takes 9 days to train on a 48 80*A100 cluster while our model is trained on 4 Nvidia A40 GPUs(48GB) for only 5 hours.

3.3 Free-form Answer Tasks

Baselines We run GPT-3.5-turbo and TableLlama (Zhang et al., 2023a) as our baselines. The prompt for each model is shown in Table 10 in Appendix. We also use the result of fine-tuning method using T5-large, and human performance from Nan et al. (2022) as baselines. Notably, the results from Nan et al. (2022) are evaluated with the original setting where the highlighted cells are provided.

Models	Fluency	Correct	Adequate	Faithful
T5-large*	94.6	54.8	50.4	50.4
GPT-3.5-turbo	99.0	83.0	85.0	96.0
TableLlama	63.0	67.0	55.0	82.0
ProTrix	96.0	77.0	71.0	91.0
Human performance*	95.0	92.4	95.6	95.6

Table 2: Human evaluation results on FetaQA. *: results reported by Nan et al. (2022)

Evaluation Metrics Since the response of our model contains step-by-step reasoning over symbolic code and natural language, BLEU (Papineni et al., 2002) would underestimate the performance of our model. BLEU also can not be used to evaluate the correctness and faithfulness of the responses. We sample 100 cases from the dataset to perform human evaluation following Nan et al. (2022). The evaluation is based on four criteria: (1) *fluency* if an answer is natural and grammatical; (2) *correctness* if an answer is correct; (3) *adequacy* if an answer contains all the information that is asked by the question; (4) *faithfulness* if an answer is faithful and grounded to the contents of the table.

Results From Table 2, we can observe that *our model exclusively trained on short-form answer tasks can adaptively generalize to give accurate and faithful explanations for complex free-form questions.* Our model achieves a fluency score of 96.0, closely following the human performance at 95.0, indicating its natural and coherent responses.

ProTrix surpasses TableLlama by 33% on fluency. TableLlama is observed to lose fluency in some cases where it generates a float number like *2008.0* to answer *what year* or a list of structured *<entity_name>* which is used to answer entity linking questions from its training set.

Our model achieves *correct* score of 77.0 and *faithful* score of 91.0 which are comparable to GPT-3.5-turbo. Although our model is only trained on short-form answer tasks, the learned planning and reasoning abilities can be utilized to answer complex *how* and *why* questions. ProTrix can analyze a general concept with actual information in the context to reach final conclusions. We present an example of the responses in Table 11 in Appendix.

Models	WikiTQ	WikiSQL	Tabfact	SciTAB	FEVEROUS	HybridQA	TATQA
ProTrix	**53.7**	**55.9**	**73.4**	**45.0**	**75.6**	**37.1**	**39.6**
w/o Planning	43.7	45.3	66.4	31.8	66.8	33.0	38.5
w/o Reasoning	41.4	44.0	65.4	33.4	70.4	31.9	27.6
w/o Planning and Reasoning	36.3	39.0	59.0	29.4	64.8	27.7	25.5

Table 3: Ablation study

4 Ablation study

We perform the following ablation studies to evaluate the effectiveness of our *Plan-the-Reason* framework (1) w/o Planning: we split each instance in `TrixInstruct` into planning and reasoning parts. We train our model with only the reasoning part of the training instances. (2) w/o Reasoning: Similar to (1), we finetune the model with only the planning part of the training instances. (3) w/o Planning and Reasoning: We finetune the model to perform end-to-end QA that generates answers directly.

The result of ablation study is presented in Table 3. *Both planning and reasoning contribute significantly to the overall generalizability and interpreterbility of our model.* Excluding planning or reasoning would cause the average performance to decrease by 7.8% or 9.6%, respectively. In w/o planning setting, the performance on SciTAB and FEVEROUS drops significantly by 13.2% and 8.8%, respectively. It suggests the importance of planning ability in utilizing commonsense knowledge and decomposing the query into reasoning chains over tables and sentences. The w/o planning and reasoning setting is similar as previous methods that train the model to answer queries directly (Xie et al., 2022; Zhang et al., 2023a). The performance of in-domain and out-of-domain benchmarks drops by 14.1% and 13.7% on average, emphasizing the effectiveness of the *Plan-then-Reason* framework in promoting generalizability across diverse tabular tasks.

5 Analysis

5.1 Program-Unsolvable Queries

To analyze the performance on queries that need commonsense knowledge or textual reasoning. We decompose the original development set of WikiTQ into program-solvable and program-unsolvable parts following Shi et al. (2020). We compare the performance of ProTrix and ProTrix-Coder with Binder (Cheng et al., 2022b),UnifiedSKG (Xie et al., 2022), TAPEX (Liu et al., 2021), TaCube (Zhou et al., 2022). Notably, our models are only trained with less than 3k examples from WikiTQ while TAPEX and TaCube are trained on the original training set which contains over 11k examples. UnifiedSKG is trained on 21 tasks involving WikiTQ. Binder prompts Codex to write code with LLMs as APIs. We do not compare with TableLlama since it is not trained on WikiTQ.

From Table 4, we can observe that ProTrix-Coder achieves the highest accuracy on program-unsolvable queries compared with fine-tuned methods. It suggests `TrixInstruct` can teach a model to understand commonsense and general concepts in the query and adaptatively plan to reason with programs or languages. ProTrix-Coder still falls behind TAPEX and TaCube on the program-solvable subset. But these models require table pretraining which is computationally expensive. ProTrix-Coder surpasses the previous generalist model by 1.5% and 4.7% on program unsolvable and solvable subsets, indicating the effectiveness of the proposed *Plan-then-Reason* framework.

Models	P-Unsolvable	P-Solvable	Overall
Closed-source Models			
Codex	40.3	53.4	50.5
Binder	41.3	71.8	65.0
Finetuning Methods			
UnifiedSKG	37.6	56.0	51.9
TAPEX*	33.6	68.0	60.4
TaCube*	34.9	68.5	61.1
ProTrix	35.0	59.1	53.8
ProTrix-Coder	38.9	60.7	55.7

Table 4: Breakdown performance on the development set of WikiTQ. P-(un)solvable: program-(un)solvable subset. *: with table pretraining.

5.2 Combining Tables and Sentences

We break down the performance on FEVEROUS into subsets following Aly et al. (2021). We choose subsets that are related to the planning and reasoning abilities to analyze our model as shown in Table 5. We use GPT-3.5-turbo and DCUF (Hu et al., 2022) as baselines. Notably, our reproduction of DCUF leverages an additional module (Wu et al., 2023) to select top 25 cells from the table to control input context length. GPT-3.5-turbo and our models use the whole table as input.

From Table 5, we can observe that ProTrix has comparable performance with GPT-3.5-turbo and DCUF on combining tables and texts and multi-hop reasoning. It suggests that our model can learn to plan the reasoning steps and assign them to programs or languages by training on `TrixInstruct`. ProTrix surpasses GPT-3.5-turbo and DCUF by 25.5% and 5.4%, respectively, on the numerical reasoning subset. It underscores that symbolic programming can achieve high-precision performance.

Models	Table+Text	Multi-hop	Numerical
Closed-source Models			
GPT-3.5-turbo	81.3	79.2	48.6
Finetuning Methods			
DCUF†	83.4	77.8	68.7
ProTrix	81.8	73.9	74.1
ProTrix-Coder	78.1	68.8	73.1

Table 5: Breakdown performance on our development set of FEVEROUS. Table+Text: combining tables and texts. Multi-hop: multi-hop reasoning. Numerical: Numerical reasoning. †: select top 25 cells from the table as input following Wu et al. (2023).

6 Related Work

Prompting Methods for LLMs Large language models can be guided to solve tasks in a step-by-step manner (Wei et al., 2022; Hao et al., 2023). Chen (2023) first utilized Chain-of-Thought (Wei et al., 2022) to enhance the reasoning of LLMs on tabular tasks and points out that textual reasoning can not generalize to large tables directly. Researchers prompt the model to select relevant rows and columns as one step in the chain of reasoning to enable LLM to focus on the following reasoning step (Jiang et al., 2023; Ye et al., 2023; Wang et al., 2024). Chen et al. (2022) proposes Program-of-Thought (PoT) that answers a question in pure programming language. Compared with textual reasoning, program-based reasoning is executed by a code interpreter, achieving high-precision reasoning in complex tabular or mathematical questions. LEVER (Ni et al., 2023) writes code to solve tabular tasks with the additional verification step. ReAcTable(Zhang et al., 2023b) prompts the model to choose to use SQL or Python tools to answer the questions in multiple turns. Binder (Cheng et al., 2022b) binds LLMs as API calls within a Python or SQL program to address the program-unsolvable aspect of the queries. Liu et al. (2023b) proposes mix self-consistency that combines the potential of both textual and program-based reasoning.

Finetuned Models Various pre-trained models are proposed for tabular tasks (Yin et al., 2020; Wang et al., 2021; Iida et al., 2021; Deng et al., 2022; Yang et al., 2022; Jiang et al., 2022; Liu et al., 2021). But they often are limited to one specific downstream fine-tuning task. As for models with generalizability, Liu et al. (2023a) mix symbolic SQL execution task with FLAN task to further finetune FLAN-T5 to improve zero-shot tabular question answering performance. Li et al. (2023) finetunes models with a large synthesized dataset of table manipulation and data augmentation to serve as a table-foundation model that understands table structures. Zhang et al. (2023a) collects an instructing tuning set that covers diverse tables and tasks and finetune Llama to obtain a generalist model without table pretrainning. Compared with existing generalist models that are expected to generate answers directly, ProTrix is interpretable by generating the process of planning and reasoning.

TaCo (Zheng et al., 2023) is finetuned with step-by-step solutions of math problems over tabular data. However, it is only limited to mathematical table reasoning and can not generalize to other types of tabular tasks. Our instruction set constructed following *Plan-then-Reason* framework can be leveraged to train generalist models over tables with sentence context while maintaining interpretability.

7 Conclusions

In this paper, we propose *Plan-then-Reason* framework to answer different types of user queries over tables with sentence context. It understands the commonsense and concepts in the query and plans the reasoning steps over programs and languages. We construct an instruction tuning set `TrixInstruct` to finetune models to obtain such planning and reasoning abilities with only 6k examples. The experiments show that our resulting models ProTrix and ProTrix-Coder generalize to unseen tabular tasks with sentence context and produce accurate and faithful explanations. Our work highlights the required abilities for generalist models over tabular tasks with sentence context, and paves the way for future research directions.

Limitations

`TrixInstruct` only contain relational tables. It currently does not contain complex tables with hierarchical headers (Cheng et al., 2022a). And `TrixInstruct` is restricted to queries over one table. It can not be directly applied to tabular tasks over multiple tables or retrieved top k tables that are noisy in context. But as to our knowledge, we are the first to study the generalist model over tables with noisy sentence context while maintaining interpretability. We plan to extend `TrixInstruct` to cover more realistic scenarios in future work.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Rami Aly, Zhijiang Guo, Michael Sejr Schlichtkrull, James Thorne, Andreas Vlachos, Christos Christodoulopoulos, Oana Cocarascu, and Arpit Mittal. 2021. Feverous: Fact extraction and verification over unstructured and structured information. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*.

Wenhu Chen. 2023. Large language models are few (1)-shot table reasoners. In *Findings of the Association for Computational Linguistics: EACL 2023*, pages 1090–1100.

Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W Cohen. 2022. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *arXiv preprint arXiv:2211.12588*.

Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. 2019. Tabfact: A large-scale dataset for table-based fact verification. *arXiv preprint arXiv:1909.02164*.

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

Zhoujun Cheng, Haoyu Dong, Zhiruo Wang, Ran Jia, Jiaqi Guo, Yan Gao, Shi Han, Jian-Guang Lou, and Dongmei Zhang. 2022a. 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)*, pages 1094–1110.

Zhoujun Cheng, Tianbao Xie, Peng Shi, Chengzu Li, Rahul Nadkarni, Yushi Hu, Caiming Xiong, Dragomir Radev, Mari Ostendorf, Luke Zettlemoyer, et al. 2022b. Binding language models in symbolic languages. In *The Eleventh International Conference on Learning Representations*.

Xiang Deng, Huan Sun, Alyssa Lees, You Wu, and Cong Yu. 2022. Turl: Table understanding through representation learning. *ACM SIGMOD Record*, 51(1):33–40.

Zihui Gu, Ju Fan, Nan Tang, Preslav Nakov, Xiaoman Zhao, and Xiaoyong Du. 2022. Pasta: table-operations aware fact verification via sentence-table cloze pre-training. *arXiv preprint arXiv:2211.02816*.

Shibo Hao, Yi Gu, Haodi Ma, Joshua Hong, Zhen Wang, Daisy Wang, and Zhiting Hu. 2023. Reasoning with language model is planning with world model. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 8154–8173, Singapore. Association for Computational Linguistics.

Nan Hu, Zirui Wu, Yuxuan Lai, Xiao Liu, and Yansong Feng. 2022. Dual-channel evidence fusion for fact verification over texts and tables. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5232–5242.

Nan Hu, Zirui Wu, Yuxuan Lai, Chen Zhang, and Yansong Feng. 2023. Unifee: Unified evidence extraction for fact verification. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1142–1152.

Hiroshi Iida, Dung Thai, Varun Manjunatha, and Mohit Iyyer. 2021. Tabbie: Pretrained representations of tabular data. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Jinhao Jiang, Kun Zhou, Zican Dong, Keming Ye, Xin Zhao, and Ji-Rong Wen. 2023. StructGPT: A general framework for large language model to reason over structured data. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9237–9251, Singapore. Association for Computational Linguistics.

Zhengbao Jiang, Yi Mao, Pengcheng He, Graham Neubig, and Weizhu Chen. 2022. Omnitab: Pretraining with natural and synthetic data for few-shot table-based question answering. *arXiv preprint arXiv:2207.03637*.

Fangyu Lei, Xiang Li, Yifan Wei, Shizhu He, Yiming Huang, Jun Zhao, and Kang Liu. 2023. S^3hqa: A three-stage approach for multi-hop text-table hybrid question answering. *arXiv preprint arXiv:2305.11725*.

Peng Li, Yeye He, Dror Yashar, Weiwei Cui, Song Ge, Haidong Zhang, Danielle Rifinski Fainman, Dongmei Zhang, and Surajit Chaudhuri. 2023. Table-gpt: Table-tuned gpt for diverse table tasks. *arXiv preprint arXiv:2310.09263*.

Qian Liu, Bei Chen, Jiaqi Guo, Morteza Ziyadi, Zeqi Lin, Weizhu Chen, and Jian-Guang Lou. 2021. Tapex: Table pre-training via learning a neural sql executor. In *International Conference on Learning Representations*.

Qian Liu, Fan Zhou, Zhengbao Jiang, Longxu Dou, and Min Lin. 2023a. From zero to hero: Examining the power of symbolic tasks in instruction tuning. *arXiv preprint arXiv:2304.07995*.

Tianyang Liu, Fei Wang, and Muhao Chen. 2023b. Rethinking tabular data understanding with large language models. *arXiv preprint arXiv:2312.16702*.

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*, pages 7787–7813, Singapore. Association for Computational Linguistics.

Linyong Nan, Chiachun Hsieh, Ziming Mao, Xi Victoria Lin, Neha Verma, Rui Zhang, Wojciech Kryściński, Hailey Schoelkopf, Riley Kong, Xiangru Tang, et al. 2022. Fetaqa: Free-form table question answering. *Transactions of the Association for Computational Linguistics*, 10:35–49.

Ansong Ni, Srini Iyer, Dragomir Radev, Veselin Stoyanov, Wen-tau Yih, Sida Wang, and Xi Victoria Lin. 2023. Lever: Learning to verify language-to-code generation with execution. In *International Conference on Machine Learning*, pages 26106–26128. PMLR.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Panupong Pasupat and Percy Liang. 2015. Compositional semantic parsing on semi-structured tables. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1470–1480, Beijing, China. Association for Computational Linguistics.

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 3505–3506.

Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, et al. 2023. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.

Tianze Shi, Chen Zhao, Jordan Boyd-Graber, Hal Daumé III, and Lillian Lee. 2020. On the potential of lexico-logical alignments for semantic parsing to SQL queries. In *Findings of EMNLP*.

Jiashuo Sun, Hang Zhang, Chen Lin, Yeyun Gong, Jian Guo, and Nan Duan. 2022. Apollo: An optimized training approach for long-form numerical reasoning. *arXiv preprint arXiv:2212.07249*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Zhiruo Wang, Haoyu Dong, Ran Jia, Jia Li, Zhiyi Fu, Shi Han, and Dongmei Zhang. 2021. Tuta: Tree-based transformers for generally structured table pre-training. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 1780–1790.

Zilong Wang, Hao Zhang, Chun-Liang Li, Julian Martin Eisenschlos, Vincent Perot, Zifeng Wang, Lesly Miculicich, Yasuhisa Fujii, Jingbo Shang, Chen-Yu Lee, et al. 2024. Chain-of-table: Evolving tables in the reasoning chain for table understanding. *arXiv preprint arXiv:2401.04398*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.

Zirui Wu, Nan Hu, and Yansong Feng. 2023. Enhancing structured evidence extraction for fact verification. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6631–6641.

Tianbao Xie, Chen Henry Wu, Peng Shi, Ruiqi Zhong, Torsten Scholak, Michihiro Yasunaga, Chien-Sheng Wu, Ming Zhong, Pengcheng Yin, Sida I Wang, et al. 2022. Unifiedskg: Unifying and multi-tasking structured knowledge grounding with text-to-text language models. *arXiv preprint arXiv:2201.05966*.

Jingfeng Yang, Aditya Gupta, Shyam Upadhyay, Luheng He, Rahul Goel, and Shachi Paul. 2022. Tableformer: Robust transformer modeling for table-text encoding. *arXiv preprint arXiv:2203.00274*.

Yunhu Ye, Binyuan Hui, Min Yang, Binhua Li, Fei Huang, and Yongbin Li. 2023. Large language models are versatile decomposers: Decomposing evidence and questions for table-based reasoning. In *Proceedings of the 46th International ACM SIGIR*

Conference on Research and Development in Information Retrieval, pages 174–184.

Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. 2020. Tabert: Pretraining for joint understanding of textual and tabular data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8413–8426.

Tianshu Zhang, Xiang Yue, Yifei Li, and Huan Sun. 2023a. Tablellama: Towards open large generalist models for tables.

Yunjia Zhang, Jordan Henkel, Avrilia Floratou, Joyce Cahoon, Shaleen Deep, and Jignesh M Patel. 2023b. Reactable: Enhancing react for table question answering. *arXiv preprint arXiv:2310.00815*.

Mingyu Zheng, Hao Yang, Wenbin Jiang, Zheng Lin, Yajuan Lyu, Qiaoqiao She, and Weiping Wang. 2023. Chain-of-thought reasoning in tabular language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 11006–11019, Singapore. Association for Computational Linguistics.

Victor Zhong, Caiming Xiong, and Richard Socher. 2017. Seq2sql: Generating structured queries from natural language using reinforcement learning. *arXiv preprint arXiv:1709.00103*.

Fan Zhou, Mengkang Hu, Haoyu Dong, Zhoujun Cheng, Fan Cheng, Shi Han, and Dongmei Zhang. 2022. TaCube: Pre-computing data cubes for answering numerical-reasoning questions over tabular data. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2278–2291, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Fengbin Zhu, Wenqiang Lei, Youcheng Huang, Chao Wang, Shuo Zhang, Jiancheng Lv, Fuli Feng, and Tat-Seng Chua. 2021. Tat-qa: A question answering benchmark on a hybrid of tabular and textual content in finance. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3277–3287.

A Implementation Details

We fully fine-tune Llama-2 7B (Touvron et al., 2023) and CodeLlama-7B (Roziere et al., 2023) with our instruction tuning set with the context of length 4096. We set the learning rate as 5e-6 and the batch size as 32. The training process uses a cosine scheduler with a 3% period for 3 epochs. We utilize DeepSpeed training with ZeRO-3 stage (Rasley et al., 2020). Our model is trained with 4 NVIDIA A40 GPUs (48GB) and the whole training process takes about 5 hours. During inference, we set the output length as 1024 and truncate large tables to fit in context length. Then we prompt the model to generate a response for the query, if there is a SQL in the response, we replace the execution result with an output of the actual SQL execution tool and ask model to generate the rest of the response. We stop when there is no SQL in the generated response. If the SQL can not be executed, we fall back to the execution result the model generates.

B Training Dataset Analysis

Our instruction tuning dataset is extracted from GPT-4 responses. We filter out the responses that have inconsistent final answers with the original dataset annotations, but the reasoning process of the responses in the training set has not been fully validated. We perform a quality analysis of our training dataset. We categorize the cases into the following types: (1) Planning error: the response fails to generate a correct plan to answer the question, (2) SQL error: The response generates SQL containing columns that do not exist at all or the SQL simply can not be executed. (3) Execution error: the execution result given by GPT-4 is wrong and (4) Reasoning error: the model achieves a correct answer with wrong reasoning.We demonstrate examples for each error type in Table 7. We sample 50 cases with sentence context and 50 cases without sentence context from `TrixInstruct` to perform manual evaluation.

	w/o Sentence	w/ Sentence	Overall
Planning Error	4%	10%	7%
SQL Error	10%	8%	9%
Execution Error	2%	2%	2%
Reasoning Error	2%	6%	4%
Accurate	82%	74%	78%

Table 6: Quality analysis of `TrixInstruct`. w/(w/o) Sentence: subset of queries over tables with(without) sentence context.

The analysis of the instruction tuning set reveals distinct error patterns in responses generated for queries over tables with or without sentence context. In the subset without sentence context, the most prevalent error type is SQL errors, indicating issues with SQL query generation for some complex questions, including referencing non-existent columns or generating unexecutable queries. As for the subset with sentence context, the most prevalent error type is planning errors. The response fails to decompose the claim into sub-claims or generates wrong plans to query the tables. Notably, both subsets showcase minimal execution errors, suggesting the proficiency of GPT-4 in providing accurate execution results. Reasoning errors are more common in the subset with sentence error with an error rate of 6%. We observe that the reasoning process generated by GPT-4 can not always follow the reasoning chains designed during planning. Overall, the manual analysis of `TrixInstruct` indicates a combined 7% planning error rate, a 9% SQL error rate, a 2% execution error rate and a 4% reasoning error rate. 78% of instances reach correct answers with accurate planning and reasoning process.

We have checked all the instances in `TrixInstruct` and make sure they do not contain any private information or offensive content.

Error Type	Claim	Response
Planning Error	McCord Museum is a history museum containing 1,440,000 objects, images and manuscripts.	- To verify the claim, we need to check the collection size of the McCord Museum. This information can be found in the table context under the "Collection size" column. - We do not need to use the sentence context for this claim as the table provides the exact information needed.
SQL Error	Georgia Southern did not make Athlon's list of preseason top 25 of 2011 NCAA Division I FCS football rankings. (column 0 is a column for rankings)	SELECT column0 FROM w WHERE column0 = 'Georgia Southern'
Execution Error	In 1949, the Chicago Cardinals season played twelve matches, resulting to ten losses and two tied matches.	SELECT COUNT(*) as Losses FROM w WHERE Result LIKE 'L%' Result: 4
Reasoning Error	In Southern Peninsular Malaysian Hokkien phonology, there are three front vowels, two central vowels and two back vowels.	- The claim states that in Southern Peninsular Malaysian Hokkien phonology, there are three front vowels two central vowels, and four back vowels. ... Therefore, the answer is REFUTES.

Table 7: Types of errors in *Plan-then-Reason* framework. For each response, we only demonstrate the part of the response where the error first occurs. The error is highlighted in red. In planning error, the model response fails to list all the information that needs to be checked apart from collection size. In SQL error, the SQL can not be used to extract the ranking of Georgia Southern. As for reasoning error, the model fails to summarize the whole planning and reasoning because of the hallucination of the original claim.

This is a table question answering task. You will answer the question based on the given context.

Question
How many medals were won by Kenya?

Table Context
Rank	Name	Nationality	Time
1	Brahim Boulami	Morocco	8:17.73
2	Reuben Kosgei	Kenya	8:18.63
3	Stephen Cherono	Kenya	8:19.98
4	Bouabdellah Tahri	France	8:20.25
5	Tim Broe	United States	8:20.75
6	Luis Miguel Martín	Spain	8:24.03
7	Raymond Yator	Kenya	8:27.19
8	Thomas Chorny	United States	9:24.26

Task
You will answer the question based on the given context.You should reach a short-form answer after reasoning. You are asked to answer the question in three steps.
1. Analyze the question and the given context. Make up a plan to answer the question.
2. Write one or more SQL to query the table for necessary information and output expected execution result.
3. Reason step-by-step to reach the final answer.

Answer

Let's answer in three steps. **1. Plan for answering the question**
- The question focus on the medals won by Kenya. According to common sense, only top 3 players can win medals.
- So we write a SQL to select the top three players. Then we count the numbers of medals won by Kenya based on the output of SQL.

2. **Write SQL and execute SQL**
SQL:
SELECT Name, Nationality FROM Table WHERE Rank <= 3

Executed result:
Name	Nationality
Reuben Kosgei	Kenya
Stephen Cherono	Kenya

3. **Step-by-Step Reasoning**:
- The question is asking about the number of medals won by Kenya.
- Based on the execution result of SQL query, only two of the players that win medals are from Kenya.
- Therefore, the total number of medals won by Kenya is 2.

The final answer is **2**.

Table 8: Prompt for generating responses for queries from WikiTQ.

This is a fact verification task. You are asked to check the veracity of the claim. Both table and sentence context are provided but you are not required to use both of them. You can use either of them or both of them.

Claim
Sony Dwi Kuncoro, born on July 7, 1984, was the champion of the Men's singles event of the 2009 Indonesia National Badminton Championship with a score of 21-17, 22-20.

Table Context
Page Title: 2009 Indonesia National Badminton Championship
Caption:
Category | Winners | Runners-up | Score
Men's singles | Sony Dwi Kuncoro | Andre Kurniawan Tedjono | 21-17, 22-20
Women's singles | Maria Kristin Yulianti | Maria Febe Kusumastuti | 21-17, 17-21, 23-21
Men's doubles | Hendra Setiawan | Rian Sukmawan , Yonathan Suryatama Dasuki | 21-9, 21-13
Women's doubles | Meiliana Jauhari , Greysia Polii | Liliyana Natsir , Shendy Puspa Irawati | 21-19, 18-21, 21-17
Mixed doubles | Tontowi Ahmad , Greysia Polii | Muhammad Rijal , Debby Susanto | 21-11, 21-19".

Sentence Context
Sony Dwi Kuncoro (1): Sony Dwi Kuncoro (born 7 July 1984) is a badminton singles player from Indonesia.
Sony Dwi Kuncore (2): He reached a career high as world number 3 on 7 October 2004.
Sony Dwi Kuncore (3): In 2009, his best performance in Superseries was semifinalist in Indonesia Open, beaten by Taufik Hidayat, 17–21, 14–21 and in Denmark Open beaten by Marc Zwiebler of Germany in three tough games.

Task
You will predict the claim's label based on the given claim and context. There are 3 possible labels: *SUPPORTS*, *REFUTES*, and *NOT ENOUGH INFO*. *SUPPORTS* means the claim is supported by the context. *REFUTES* means the claim is refuted by the context. *NOT ENOUGH INFO* means the context does not provide enough information to make the support or refute the claim. You are asked to check the claim in three steps.
1. Analyze the question and the given context. Make up a plan to verify the claim.
2. Write one or more SQL to query the table for necessary information and output expected execution result.
3. Reason step-by-step to reach the final answer.

Answer:
Let's check the claim in 3 steps.

1. Plan for verify the claim
- Based on the claim and the given context. The information for winner of Men's singles can be obtained from the table. We could write a SQL to query the name of the winner and the score of the final for Men's singles.
- The claim also states that Sony Dwi Kuncoro is born on July 7, 1984. We will use the sentences to check this part of claim.

2. Write SQL and execute SQL
SQL:
SELECT Winners FROM Table WHERE Category = Men's singles

Executed result:
Category | Winners | Score
Men's singles | Sony Dwi Kuncoro | 21-17, 22-20

3. **Step-by-step Reasoning**
- The claim states Sony Dwi Kuncoro, born on July 7, 1984, was the champion of the Men's singles event of the 2009 Indonesia National Badminton Championship with a score of 21-17, 22-20.
- The SQL query result supports the claim. Sony Dwi Kuncoro was the champion of the Men's singles event of the 2009 Indonesia National Badminton Championship with a score of 21-17, 22-20.
- The sentence evidence Sony Dwi Kuncoro (1) supports that he was indeed born on July 7, 1984.

Therefore, the answer is **SUPPORTS**

Table 9: Prompt for generating responses for queries from FEVEROUS

TableLlama & **GPT-3.5-turbo** Below is an instruction that describes a task, paired with an input that provides further context. Write a response that appropriately completes the request. ### Instruction: This is a free-form table question answering task. The goal for this task is to answer the given question based on the given table. ### Input: [TLE] The Wikipedia page title of this table is *page title*. The Wikipedia section title of this table is *section title*. [TAB] *table* ### Question: *question* ### Response:
Our model ## Question *question* ## Table Context Page title: *page title* Section title:*section title* *table* ## Task You will answer the question based on the given context. You should reach a free-form answer after reasoning. You are asked to answer the question in three steps. 1. Analyze the question and the given context. Make up a plan to answer the question. 2. Write one or more SQL to query the table for necessary information and output expected execution result. 3. Reason step-by-step to reach the final answer. ##Answer:

Table 10: Prompt for evaluation on FetaQA

Question
How well did Peralta play at the 2015 CONCACAF Gold Cup?

Table Context
Page Title: Oribe Peralta
Caption: International goals
| Goal | Date | Venue | Opponent | Score | Result | Competition
...17 rows with goals in 2014 FIFA World Cup qualification
|18. | 9 July 2015 | Soldier Field, Chicago, United States | Cuba | 1-0 | 6-0 | 2015 CONCACAF Gold Cup
|19. | 9 July 2015 | Soldier Field, Chicago, United States | Cuba | 3-0 | 6-0 | 2015 CONCACAF Gold Cup
|20. | 9 July 2015 | Soldier Field, Chicago, United States | Cuba | 5-0 | 6-0 | 2015 CONCACAF Gold Cup
|21. | 26 July 2015 | Lincoln Financial Field, Philadelphia, United States | Jamaica | 3-0 | 3-1 | 2015 CONCACAF Gold Cup
...5 rows with goals in 2015 CONCACAF Cup, 2017 FIFA Confederations Cup

Gold answer
Peralta played at the 2015 CONCACAF Gold Cup and scored a 6-0 defeat of Cuba in Chicago, as well as a 3-1 win against Jamaica in Philadelphia.

GPT-3.5-turbo response
Peralta played very well at the 2015 CONCACAF Gold Cup, scoring a total of 8 goals in the tournament. ✗

TableLlama
Peralta scored twice in a 6-0 win over Cuba. ✗

Our model
1. Plan for answering the question
   - The question asks for an assessment of Peralta's performance at the 2015 CONCACAF Gold Cup.
   - To answer this, we need to filter the table for entries that are related to the 2015 CONCACAF Gold Cup and then analyze the results to determine if Peralta played well or not.

2. Write SQL and execute SQL
SQL
SELECT Goal, Date, Venue, Opponent, Score, Result, Competition FROM International_Goals WHERE Competition = '2015 CONCACAF Gold Cup'

Executed result:
Goal | Date | Venue | Opponent | Score | Result | Competition
18. | 9 July 2015 | Soldier Field, Chicago, United States | Cuba | 1-0 | 6-0 | 2015 CONCACAF Gold Cup
19. | 9 July 2015 | Soldier Field, Chicago, United States | Cuba | 3-0 | 6-0 | 2015 CONCACAF Gold Cup
20. | 9 July 2015 | Soldier Field, Chicago, United States | Cuba | 5-0 | 6-0 | 2015 CONCACAF Gold Cup
21. | 26 July 2015 | Lincoln Financial Field, Philadelphia, United States | Jamaica | 3-0 | 3-1 | 2015 CONCACAF Gold Cup

3. Step-by-Step Reasoning:
   - The SQL query filters the table to show only the goals scored by Peralta during the 2015 CONCACAF Gold Cup.
   - From the execution result, we can see that Peralta scored three goals on 9 July 2015 against Cuba at Soldier Field, Chicago, United States and one goal on 26 July 2015 against Jamaica on 26 July 2015 at Lincoln Financial Field, Philadelphia, United States ✔
   - The results were 6-0 and 3-1, respectively, which indicates a dominant performance by Peralta and his team.
   - Therefore, Peralta played very well at the 2015 CONCACAF Gold Cup.

The final answer is that Peralta played very well at the 2015 CONCACAF Gold Cup. ✔

Table 11: An example of the responses of models to FetaQA