SemEval-2025 Task 8: Question Answering over Tabular Data

Jorge Osés Grijalba[1,3], Luis Alfonso Ureña-López[1], Eugenio Martínez Cámara[1], Jose Camacho-Collados[2]
[1]SINAI Research Group, Advanced Studies Center in ICT (CEATIC)
University of Jaén, Spain
[2]Cardiff NLP, Cardiff University, United Kingdom
[3]Graphext, Spain
jorge@graphext.com, laurena@ujaen.es, emcamara@ujaen.es,
camachocolladosj@cardiff.ac.uk

Abstract

We introduce the findings and results of SemEval-2025 Task 8: Question Answering over Tabular Data. This shared task featured two subtasks, DataBench and DataBench Lite. DataBench consists on question answering over tabular data, and DataBench Lite comprises small datasets that might be easier to manage by current models by for example fitting them into a prompt.

In this paper, we present the task, analyze a number of system submissions and discuss the results. The results show how approaches leveraging LLMs dominated the task, with larger models exhibiting a considerably superior performance compared to small models. Open models proved competitive with respect to proprietary LLMs, but further work would be required to improve the performance of smaller models.

1 Introduction

Large Language Models (LLMs) have demonstrated emerging capabilities (Wei et al., 2022), with one of the latest recognized tasks being Question Answering (QA) on Tabular Data (Chen, 2023). **QA on Tabular Data**, as illustrated in Figure 1, involves responding to natural language queries using structured information stored in tables. Different approaches exist for retrieving answers, including translating natural language questions into formal programming languages like SQL, which can then be used to interact with databases (Nan et al., 2022a; Aly et al., 2021; Nan et al., 2022b). Since these models are widely applied across various domains, ensuring their accurate evaluation is essential. However, the research community currently lacks a comprehensive evaluation benchmark to assess and compare different LLMs and prompting strategies for this task.

Benchmarking in Tabular QA has traditionally relied on a limited set of collections, such as (Zhong et al., 2017; Pasupat and Liang, 2015; Kweon et al., 2023), which are predominantly based on tables extracted from Wikipedia.

Name	Subject	Score
Alice	Math	92
Bob	Science	85
Charlie	History	78
Diana	Math	88
Ethan	Science	95

Q: Who scored highest in Math?
A: Alice

Q: What is the mean score for Science?
A: 80

Figure 1: Examples of correct and wrong answers to simple and factual questions made on Tabular Data

While these datasets have been widely used, they exhibit common characteristics—such as low data variety and a small number of columns—that make them less representative of the complex tabular data encountered in real-world applications. Additionally, Wikipedia tables often pose challenges related to scale, data cleanliness, and structural limitations.

To address these shortcomings, we introduced DataBench at LREC-COLING 2024 (Osés-Grijalba et al., 2024), a novel benchmark designed to provide a more diverse and realistic evaluation framework for question answering on tabular data. DataBench consists of real-world datasets from various domains, featuring large and heterogeneous tables, along with a rich collection of annotated question-answer pairs.

Domain	Datasets	Rows	Columns
Business	26	1,156,538	534
Health	7	98,032	123
Social	16	1,189,476	508
Sports	6	398,778	177
Travel	10	427,151	273
Total	65	3,269,975	1615

Table 1: Domains and statistical data of DataBench.

The availability of DataBench encourages us to challenge the research community to design QA on tabular data models with the ability of processing and answering questions about data stored in tables. Accordingly, we propose the Tabular QA task with the enough freedom to encourage the creativity of researchers to provide solutions to this challenge. As we will describe hereinafter, we provide two versions of DataBench, and one of them is oriented to facilitate the use of models that are not able to process large contexts by presenting datasets small enough to fit whole into a single prompt by whatever representation the users desire.

The task has arisen the participation of more than 100 teams, with 35 research teams having submitted a system description paper. The top-ranked models evidence the superiority of models leveraging LLMs, and among them those with large size of parameters. However, the approaches also followed evidence that the task needs smart prompting strategies, hence the size of the models is not the only important feature at play. The top ranked systems for each kind are described further in the results section.

Codabench. The competition has been hosted on Codabench[1] where you can find more details, examples and links for the relevant test set data.

2 Datasets

DataBench was introduced in Osés-Grijalba et al. (2024) to provide a dataset for Question Answering over Tabular Data, addressing challenges commonly found in real-world data that are often absent from existing datasets, which primarily consist of Wikipedia tables. By incorporating these complexities, DataBench offers a more reliable benchmark for developing models capable of handling such data.

The dataset includes 65 datasets spanning various domains, as detailed in Table 1. These domains include **Business**, covering topics such as churn prediction and market basket analysis; **Health**, featuring datasets on diseases and treatments; **Social**, containing data from surveys and social networks; and **Travel**, which focuses on the travel industry.

DataBench also includes 1,300 tagged QA pairs, providing insights into the types of columns being used. The questions, as illustrated in Table 2, are concise and objective, each targeting specific pieces of information expected in a particular format.

DataBench Lite DataBench Lite was also introduced along DataBench and it contains sampled versions of all datasets and answers for each sampled version. The size of this is essentially the same as Table 1 but with only 20 rows per dataset. Everything else stays the same, except for the answers to the questions which have been adapted to fit the sampled data. We created DataBench Lite because of the limitation of most language models to a given context window. This smaller version of the data can help explore fitting the whole data within the prompt following an In-Context Learning approach, or to test models which have not been able to fully scale up to large sizes yet.

DataBench and DataBench Lite are hosted publicly on HuggingFace.[2]

Train and Development sets The full set of DataBench was divided in two sets: the **Train Set**, containing the first 40 datasets, and the **Dev set**, containing the last 15. This artificial distinction was made in order to facilitate evaluating on the development set during the first phase of the competition, but otherwise participants were encouraged to use the two sets as they found best fit.

Test set The test set released in the competition phase comprises 522 QA pairs over 15 datasets. In Table 3 we can see that the number of total rows of data is 438,909 and the number of columns is 391. The original five domains from DataBench have been included here as well. Table 4 shows the data types of the columns of the datasets. This full test set was used for Subtask A, while a reduced version (*lite*) with up to twenty rows per dataset was used for Subtask B.

The language of the datasets is primarily English, and only understanding English is required in order to retrieve the appropriate answers.

[1]https://www.codabench.org/competitions/3360/

[2]https://huggingface.co/datasets/cardiffnlp/databench/

Question	Answer	Type	Columns Used	Column Types
Is Lil Llama the oldest passenger?	false	boolean	Name, Age	category, number
What's the class of the oldest passenger?	first	category	Name, Age	category, number
What's the lowest fare paid?	10.2	number	Fare	number
Who are the passengers under 30?	[Lil Lama, Cody Lama]	list[category]	Name, Age	category, number
What are the fares paid by passengers under 30?	[30.25, 10.2]	list[number]	Age, Fare	number, category

Table 2: Types of Question-Answer pairs present in our benchmark.

	Name	Rows	Cols	Domain	Source (Reference TODO)
1	IBM HR	1470	35	Business	IBM (Subhash, 2018)
2	TripAdvisor Reviews	20000	10	Travel and Locations	TripAdvisor (Li, 2014)
3	World Bank	239461	20	Business	World Bank (The World Bank, 2025)
4	Taxonomy	703	8	Health	IAB (IAB Tech Lab, 2017)
5	Open Food Facts	9483	204	Business	OpenFoodFacts (OpenFoodFacts, 2025)
6	Cost of Living	121	8	Travel and Locations	Kaggle (Myrios, 2024)
7	College Admissions	500	9	Social Networks and Surveys	Kaggle (Sacharya, 2024)
8	Med Cost	1338	7	Health	Kaggle (Peker, 2024)
9	Lift	3000	5	Sports and Entertainment	Kaggle (Waqi786, 2024)
10	Mortality	400	7	Health	Kaggle (Rajanand, 2024)
11	NBA	8835	30	Sports and Entertainment	Kaggle (Kumar, 2023)
12	Gestational	1012	7	Health	Kaggle (Banerjee, 2024)
13	Fires	517	11	Social Networks and Surveys	Kaggle (Rostami, 2024)
14	Coffee	149116	17	Business	Kaggle (Ibrahim, 2024)
15	Books	40	13	Business	Kaggle (Chowdhury, 2023)
	Total	**438909**	**391**		

Table 3: Datasets included in the test set with their number of rows and columns, as well as their domain and source reference.

Category Type	**Number of Categories**
boolean	129
category	74
number	156
list[number]	91
list[category]	72
Total	522

Table 4: Types present in the test set.

3 Pilot Task

In the original DataBench paper (Osés-Grijalba et al., 2024), we compared two approaches based on LLaMA (Touvron et al., 2023) (including the code version) and ChatGPT. Specifically, we examined two different zero-shot approaches and tested multiple prompts for each over DataBench Lite. These models were evaluated in the 65 datasets included in DataBench Lite.

The first approach, referred to as *In-Context Learning*, involved including the entire dataset in the prompt, formatted as a CSV, and then directly asking the intended question while specifying the expected response format.

The second approach, called *Code*, functioned as a Python code-completion task, where the model was instructed to complete a function. This function was given a structured representation of the dataset—including at least its column names—and could only use PANDAS and NUMPY to perform the task.

Overall, the *In-Context Learning* approach produced worse results and exhibited more hallucinations. However, it performed relatively better on certain subsets (such as boolean datasets) and was generally faster since it did not rely on a code interpreter. In contrast, the code-based approach interacted with the data by generating code through completion models to execute the required operations.

A key challenge with the first approach was managing hallucinations and ensuring that users could verify the correctness of the response in real-world applications. On the other hand, the main challenge with code-based methods was providing sufficient information about the dataset to allow accurate code generation. For example, it was often necessary to supply the model with column names to enable proper data access.

A summary of the results in the pilot task is provided in Table 12 in the Appendix. Overall, the results indicated that the task remained unsolved, with accuracy scores in the early tests generally below 50% for small open source models, which were the focus of our approach. This made the approaches unreliable for most applications. However, there was potential for improvement through more grounded methods, including more refined strategies and fine-tuning techniques that were not explored in the pilot study.

4 Task Organization

The task was run in two phases, making use of the Codabench platform. Both DataBench and DataBench Lite were evaluated on all submissions at the same time, but participants could make a submission with either of them, or both.

In the platform we only evaluated accuracy against a ground truth set for both subtasks, allowing participants to freely build any systems they would like to solve the task. They were informed of requirements to share the type of model used, code and general descriptions in order to qualify for the final ranking. Rankings for both subtasks were also made separately, and special rankings were provided for small models of up to 9 billion parameters and open source models in general.

Evaluation script. Participants were also provided with a Python package (databench_eval[3]) to make the process of making a submission more streamlined in case they wished to use it. Due to the open-ended nature of the task, we opted for an open-source evaluation function that heuristically evaluates the output provided by the users against the ground truth depending on the type expected. The function returns either *true* or *false* for a given pair of response and ground truth and according to the expected semantic value of the response. We then add up the percentage of *true* values to compute the accuracy. This evaluation carries a number of checks to allow for smaller models to commit some small format errors that would otherwise hinder their performance, such as trimming down extra spaces in the response or allowing for drifts in calculations smaller than the second decimal position for numerical values. This function was open-sourced from the start and received feedback from participants which resulted in a number of small changes which can be seen through the history of the GitHub repo.

Given that LLMs generate text of any kind, and the almost infinite possibilities of format changes for a given answer, we also performed a manual evaluation of the results shown in the final ranking in order to ensure small formatting mistakes were mitigated as much as possible. This manual evaluation in the competition phase was done for the top 10 results for each category, and resulted only in very small changes which did not in the end affect the rankings.

[3]https://github.com/jorses/databench_eval

Development Phase Running from the 8th of September 2024 to the 9th of January 2025. In this phase participants were only provided with the full train and development sets, including tags on the columns where the answer was to be extracted from and the type expected of the answer. They were free to make as many submissions as they wanted and had full access to their scores. A public ranking was available on the platform where participants could freely choose to display their results, but were not forced to do so. Participants were provided with a minimal **baseline** script that could be executed locally without any GPU on most consumer hardware and yielded 28.05 and 30.22 of accuracy for DataBench and DataBench Lite respectively with the use of a 4bit quantized version of the stable-coder-3b model (TheBloke and StabilityAI, 2023). This baseline is still available in the GitHub page for the evaluation benchmark.

Competition Phase The full blind test set, including only the questions from the test set, was released on 9 January 2025. The competition ran until 31 January 2025. During this phase, participants were allowed only three submissions and the public ranking was not available. After the competition, participants who wished to take part in the final ranking had to fill out a Google form containing the information on their system described earlier.

5 Participating systems

106 teams submitted valid results to the Codabench January Competition phase. All of them were informed of the requirement to complete a form describing their approach in order to participate in the ranking. Out of those 106 teams, 51 chose to fill the form, which are the teams included in this paper. Finally, 35 teams submitted a system description paper to be included in the proceedings, and more details about the approaches can be found on their specific articles.

Among all participants, thirty-five teams used an approach based purely on open-weight models, while sixteen teams used a proprietary model. A separate category was created for open-source models with fewer than 9 billion parameters, as these models can efficiently run after 4-bit quantization is performed run on CPU on modern consumer hardware and were the focus of our pilot task.

In general, most successful teams implemented a code-based LLM approach with few-shot prompt-

Rank	Team	Accuracy
1	TeleAI	95.02
2	AILS-NTUA	89.85*
3	SRPOL AIS	89.66
4	sonrobok4	89.46*
5	langtechdata61	88.12*
6	AILS-NTUA	87.16
7	Core Intelligence - Accuris	87.16*
8	HITSZ-HLT	86.97
9	Firefly	86.40*
10	G-MACT	86.02
11	SBU-NLP	85.63
12	111dut	85.44*
13	Oseibrefo-Liang	84.67
14	ITU-NLP	84.10
15	grazh	83.72
16	Howard University- AI4PC	81.42
17	QleverAnswering-PUCRS	81.03
18	I2R-NLP	80.65
19	langtechdata61	80.46*
20	anotheroption	80.08
21	Exploration Lab IITK	79.69
22	CCNUNLP	79.50
23	Tabular_lllm_njupt	79.50*
24	Sherlok	79.31
25	Saama Technologies	78.35
26	ScottyPoseidon	76.63^{sm}
27	NEST	76.05*
28	MINDS	72.41
29	MRT	70.50
30	Aestar	70.50*
31	Dataground	68.97^{sm}
32	IUST_Champs	68.77
33	LyS Group	67.62
34	NexGenius	65.64^{sm}
35	pp78107049iir	65.13*
36	langtechdata61	64.94*
37	Tree-Search	64.56^{sm}
38	TableWise	63.98
39	Myo Thiha	62.45
40	serrz	58.05*
41	tabaqa_team	54.79
42	nevvton	52.87
43	Basharat Ali	43.10^{sm}
44	AlphaPro	38.46
45	TSOTSA	37.74*
46	CAILMD-24	36.40
	baseline	26.00
47	Laughter	10.54
48	Jadavpur University	9.20*
49	Laughter	8.24^{sm}
50	TQASSN	7.85^{sm}
51	SUT	3.70^{sm}
52	fahimebehzadi	1.64^{sm}

Table 5: Subtask A) DataBench Rankings - Proprietary models marked with an asterisk after accuracy, small open source models with sm.

ing, coupled with some innovations such as self-correction and table-tailored prompting. In the following, we describe the teams that achieved the highest scores in each of the categories (see the following section for more details on the experimental results).

TeleAI The team from the Institute of Artificial Intelligence (TeleAI), China Telecom Corp Ltd, achieved the highest accuracy in the DataBench benchmark, with 95.02% on DataBench and 92.91% on DataBench Lite. Their approach leveraged a structured reasoning framework combining program-aided query refinement and code generation to enhance structured data understanding.

According to their own description provided in the *Google Form*, the task is implemented using the ReAct prompting approach (Yao et al., 2023). First, Python processes table data, and an LLM generates natural language descriptions that provide an overview of the table's content, explain column names, identify data types, define value ranges, and include row examples. This process is referred to as *Table Schema Generation*. Within each reasoning cycle in ReAct prompting, the *thought* component represents the original or refined query. The *action* stage follows a structured, program-assisted approach involving the *Query Expansion & Linking - Schema Refinement - CoT Generation - Code Generation & Execution* pipeline. Query Expansion decomposes the original query into finer sub-queries, identifying relevant columns and entity values. Schema Refinement extracts key structures from the dataset to simplify large tables. The *observation* step records the results of program execution. After completing each thought-action-observation cycle, the LLM determines whether the answer can be inferred. If the answer is sufficient, the process transitions to the *Answer Summary* module, which generates a structured response; otherwise, the query is refined for another iteration.

Rank	Team	Accuracy
1	TeleAI	92.91
2	AILSNTUA	88.89*
3	langtechdata61	88.70*
4	SRPOL AIS	86.59
5	Firefly	86.21*
6	SBU-NLP	86.02
7	OseibrefoLiang	86.02
8	HITSZ-HLT	85.82
9	sonrobok4	85.25*
10	ITU-NLP	85.06
11	tabaqa_team	84.87
12	GMACT	84.48
13	111dut	83.14*
14	Howard University- AI4PC	80.46
15	Tabular_lllm_njupt	80.46*
16	QleverAnswering-PUCRS	80.27
17	Sherlok	79.69
18	NEST	79.12*
19	Saama Technologies	78.93
20	AILS-NTUA	78.54
21	anotheroption	77.97
22	I2R-NLP	77.20
23	CCNUNLP	76.82
24	langtechdata61	76.05*
25	ScottyPoseidon	74.71^{sm}
26	MINDS	74.14
27	Aestar	71.65*
28	IUST_Champs	69.73
29	Dataground	69.35^{sm}
30	langtechdata61	69.16*
31	LyS Group	68.97
32	TableWise	68.77
33	CAILMD-24	67.43
34	NexGenius	66.22^{sm}
35	Tree-Search	64.94^{sm}
36	pp78107049iir	64.56*
37	TSOTSA	62.26*
38	Myo Thiha	60.73
39	Exploration Lab IITK	58.81
40	AlphaPro	53.85
41	nevvton	53.26
42	Basharat Ali	43.87^{sm}
43	grazh	36.78
44	SUT	34.38^{sm}
45	MRT	33.91
46	serrz	30.90*
47	Laughter	30.00
	baseline	27.00
48	TQASSN	15.13^{sm}
49	Laughter	10.73^{sm}
50	Jadavpur University	9.96*
51	Core Intelligence - Accuris	9.77*
52	fahimebehzadi	1.42^{sm}

Table 6: Subtask B) DataBench Lite Rankings - Proprietary models marked with an asterisk after accuracy, small open source models under 9billion parameters with sm.

To improve query expansion and linking, the team fine-tuned their model using the DataBench train and dev sets. Data distillation from advanced LLMs, combined with Rejection Sampling, was applied to construct and select supervised fine-tuning (SFT) data. For model selection, they utilized the *Mistral-Large-Instruct-2407* LLM for the code generation module, while the *Qwen2.5-72B-Instruct* LLM handled other components. Their structured approach demonstrated superior performance in accurately interpreting and processing structured queries.

AILS-NTUA According to their own description in the google form, the AILS-NTUA team topped the ranks for the used code generation with exemplars for few-shot prompting, utilizing the proprietary *anthropic.claude-3-5-sonnet-20241022-v2:0* model. Their accuracy on DataBench was 89.85%, and on DataBench Lite, it was 88.89%.

ScottyPoseidon This team used the *unsloth/phi-4-unsloth-bnb-4bit* model with 8.48 billion parameters. They tackled the problem using code generation by providing the dataset schema and sample rows to the engine. Their approach leveraged multiple LLM models to build a system with Chain-of-Thought (CoT) reasoning, integrating an explainer, coder, and reviewer LLMs. This system collaboratively generated and refined code to produce an effective solution. They used the development set for validation.

6 Results

In this section, we present the main results of the competition for those that chose to submit a Google Form. Table 5 shows the results in the DataBench test set, while Table 6 shows the results in the reduced DataBench lite test set. Overall, the scores vary widely, which is expected given the large diversity of models and the completely different ap-

Rank	Team	Accuracy
1	ScottyPoseidon	76.63
2	Dataground	68.97
3	NexGenius	65.64
4	Tree-Search	64.56
5	Basharat Ali	43.10
	baseline	26.00
6	Laughter	8.24
7	TQASSN	7.85
8	SUT	3.70
9	fahimebehzadi	1.64

Table 7: Subtask A) Only open models under 9 billion parameters.

Rank	Team	Accuracy
1	ScottyPoseidon	74.71
2	Dataground	69.35
3	NexGenius	66.22
4	Tree-Search	64.94
5	Basharat Ali	43.87
6	SUT	34.38
	baseline	27.00
7	TQASSN	15.13
8	Laughter	10.73
9	fahimebehzadi	1.42

Table 8: Subtask B) Only open models under 9 billion parameters.

proaches.The biggest open source models ranked overall on par with the closed-source approaches, and small models came behind.

The best small open source approach, ScottyPoseidon, ranked 25th in the competition, and they become more common as we approach the bottom of the ranking. The best approach overall claims to use exclusively big open source models.

Baseline results. The result of executing the baseline we provided the participants with (see Section 4 for more details) over the test set data yielded 26.00 and 27.00 accuracy scores for DataBench and DataBench Lite respectively, making the difficulty of this task approximately the same as the proposed development set. The baseline used was intentionally very simple, mimicking the approach followed for the pilot task and getting similar results as the ones displayed in our original paper (Osés-Grijalba et al., 2024).

Results by answer type. We have also averaged all of the results of all submissions by type in Table 9. Performance across submissions seems to vary for each type, with lists of categories proving the hardest and boolean questions proving the easiest in both rankings.

Category	**DataBench (Acc)**	**DataBench Lite (Acc)**
boolean	63.90	61.94
category	52.85	52.13
list[category]	46.93	45.89
list[number]	50.56	48.96
number	56.42	54.41
Overall	55.43	53.82

Table 9: Average accuracy in both suites across all submissions by type in the test set

In the following, we provide more details of participants' ranking in two additional categories we enabled in the task: (1) open models (including methods that rely on non-proprietary models and have at least their weights available) and (2) small models (with models consisting of lower than 9 billion parameters).

(1) Ranking of open models. Large open models in general rank pretty similarly to the proprietary ones as can be observed in Table 5 and Table 6. Open-only rankings are displayed in Table 10 and Table 11. The first 10 positions on both rankings contain 5 and 6 open models respectively, and in both the best performing team uses a purely open source approach. Large open source models have established themselves as a solid alternative to proprietary approaches for both subtasks.

(2) Ranking of small open models. Small open-weights models under 9 billion parameters lag behind both their open source and proprietary larger counterparts as we can see in Table 7 and Table 8. The best performance for these models, the **Scotty-Poseidon** team, ranks 25th in the general ranking for subtask A) and in the 26th position for task B). Only three others achieve over 60% accuracy in any of the task. This further showcases the need for more research to be done for small models in the field of Tabular QA in order to achieve performances similar to large models.

7 Conclusions and Future Work

In this paper, we presented the SemEval task on Question Answering over Tabular Data. The test set consisted of tabular datasets from different domains and questions about different types. The results of the competition suggest that existing models can answer these types of questions reliably, as long

Rank	Team	Accuracy
1	TeleAI	95.02
2	SRPOL AIS	89.66
3	HITSZ-HLT	86.97
4	G-MACT	86.02
5	SBU-NLP	85.63
6	Oseibrefo-Liang	84.67
7	ITU-NLP	84.10
8	grazh	83.72
9	Howard University- AI4PC	81.42
10	QleverAnswering-PUCRS	81.03
11	I2R-NLP	80.65
12	anotheroption	80.08
13	Exploration Lab IITK	79.69
14	CCNUNLP	79.50
15	Sherlok	79.31
16	Saama Technologies	78.35
17	ScottyPoseidon	76.63
18	MINDS	72.41
19	MRT	70.50
20	Dataground	68.97
21	IUST_Champs	68.77
22	LyS Group	67.62
23	NexGenius	65.64
24	Tree-Search	64.56
25	TableWise	63.98
26	Myo Thiha	62.45
27	tabaqa_team	54.79
28	nevvton	52.87
29	Basharat Ali	43.10
30	AlphaPro	38.46
31	CAILMD-24	36.40
	baseline	26.00
32	Laughter	10.54
33	Laughter	8.24
34	TQASSN	7.85
35	SUT	3.70
36	fahimebehzadi	1.64

Table 10: Subtask A) DataBench open rankings including small models and baseline without rank.

Rank	Team	Accuracy
1	TeleAI	92.91
2	SRPOL AIS	86.59
3	SBU-NLP	86.02
4	Oseibrefo-Liang	86.02
5	HITSZ-HLT	85.82
6	ITU-NLP	85.06
7	tabaqa_team	84.87
8	G-MACT	84.48
9	Howard University- AI4PC	80.46
10	QleverAnswering-PUCRS	80.27
11	Sherlok	79.69
12	Saama Technologies	78.93
13	AILS-NTUA	78.54
14	anotheroption	77.97
15	I2R-NLP	77.20
16	CCNUNLP	76.82
17	ScottyPoseidon	74.71
18	MINDS	74.14
19	IUST_Champs	69.73
20	Dataground	69.35
21	LyS Group	68.97
22	TableWise	68.77
23	CAILMD-24	67.43
24	NexGenius	66.22
25	Tree-Search	64.94
26	Myo Thiha	60.73
27	Exploration Lab IITK	58.81
28	AlphaPro	53.85
29	nevvton	53.26
30	Basharat Ali	43.87
31	MRT	33.91
32	SUT	34.38
33	Laughter	30.00
	baseline	27.00
34	TQASSN	15.13
35	Laughter	10.73
36	fahimebehzadi	1.42

Table 11: Subtask B) DataBench open rankings including small models and baseline without rank.

as they are tailored and specialized in the task. In general, out of the box models cannot solve the task and struggle for the most part, and the results suggest there is a need for specialised and in-domain trained solutions beyond LLMs. Moreover, smaller LLMs (below 9B parameters) are far from the best performing models, and reinforce the challenging nature of the task for general-domain models.

In addition to this task, we are looking at expanding this benchmark to other language and domains, as well as other types of questions, including those ones that require different types of reasoning. This reasoning can be in the form of requiring information from different columns, or to perform operations beyond what is actually displayed in the table, for example.

Regarding the expansion of DataBench to languages different from English, we also released

DataBenchSPA for the Spanish language (Osés Grijalba et al., 2024), including an accompanying shared task[4]. This is an example of an expansion to a different language, but in general most languages do not have a suitable benchmark, and therefore there is plenty of room for future work in this area.

Limitations

The questions asked in this task are in general short and factual and do not require complex reasoning over the datasets, in large part because of the difficulty in developing a standard evaluation framework for longer more complex questions over tabular data that can be used in a competition. These questions are meant to provide a general baseline for models, but are by no means comprehensive of all types of questions that can be answered using tabular datasets.

Also, the sheer scope of different datasets used in a myriad of use cases would require us to keep growing our collection in order to provide a benchmark that is of relevance to most users. At the moment, the test set is rather small and would not be representative of all tabular data.

Acknowledgements

This work was partially supported by the grants PID2020-116118GA-I00 and TED2021-130145BI00 funded by MCIN/AEI/10.13039/501100011033. Jose Camacho-Collados is supported by a UKRI Future Leaders Fellowship.

[4] https://www.codabench.org/competitions/5538/

References

Rami Aly, Zhijiang Guo, Michael Sejr Schlichtkrull, James Thorne, Andreas Vlachos, Christos Christodoulopoulos, Oana Cocarascu, and Arpit Mittal. 2021. The fact extraction and VERification over unstructured and structured information (FEVEROUS) shared task. In *Proceedings of the Fourth Workshop on Fact Extraction and VERification (FEVER)*, pages 1–13, Dominican Republic. Association for Computational Linguistics.

Sourav Banerjee. 2024. Maternal mortality dataset. https://www.kaggle.com/datasets/iamsouravbanerjee/maternal-mortality-dataset. Accessed: 2025-03-01.

Wenhu Chen. 2023. Large language models are few(1)-shot table reasoners. In *Findings of the Association for Computational Linguistics: EACL 2023*, pages 1120–1130, Dubrovnik, Croatia. Association for Computational Linguistics.

Fahmida Chowdhury. 2023. Bookstore inventory: Best sellers and new releases. Accessed: 2025-01-07.

IAB Tech Lab. 2017. Content Taxonomy Version 2.0. Accessed: 2025-03-07.

Ahmed Mohamed Ibrahim. 2024. Coffee shop sales dataset. https://www.kaggle.com/datasets/ahmedmohamedibrahim1/coffee-shop-sales-dataset. Accessed: 2025-03-01.

Shivam Kumar. 2023. Nba stats dataset for the last 10 years. Accessed: 2025-01-07.

Sunjun Kweon, Yeonsu Kwon, Seonhee Cho, Yohan Jo, and Edward Choi. 2023. Open-wikitable: Dataset for open domain question answering with complex reasoning over table.

Jiwei Li. 2014. Hotel Review Dataset. Accessed: 2025-01-07.

Myrios. 2024. Cost of living index by country (number 2024). Accessed: 2025-03-01.

Linyong Nan, Chiachun Hsieh, Ziming Mao, Xi Victoria Lin, Neha Verma, Rui Zhang, Wojciech Kryściński, Hailey Schoelkopf, Riley Kong, Xiangru Tang, Mutethia Mutuma, Ben Rosand, Isabel Trindade, Renusree Bandaru, Jacob Cunningham, Caiming Xiong, Dragomir Radev, and Dragomir Radev. 2022a. FeTaQA: Free-form table question answering. *Transactions of the Association for Computational Linguistics*, 10:35–49.

Linyong Nan, Chiachun Hsieh, Ziming Mao, Xi Victoria Lin, Neha Verma, Rui Zhang, Wojciech Kryściński, Hailey Schoelkopf, Riley Kong, Xiangru Tang, Mutethia Mutuma, Ben Rosand, Isabel Trindade, Renusree Bandaru, Jacob Cunningham, Caiming Xiong, Dragomir Radev, and Dragomir Radev. 2022b. FeTaQA: Free-form table question answering. *Transactions of the Association for Computational Linguistics*, 10:35–49.

OpenFoodFacts. 2025. Openfoodfacts mercadona product dataset. Accessed: 2025-01-07.

Jorge Osés Grijalba, Luis Alfonso Ureña López, Jose Camacho-Collados, and Eugenio Martínez Cámara. 2024. Towards quality benchmarking in question answering over tabular data in spanish. *Procesamiento del lenguaje natural*, 73:283–296.

Jorge Osés-Grijalba, Luis Alfonso Ureña-López, Eugenio Martínez Cámara, and Jose Camacho-Collados. 2024. Question answering over tabular data with databench: A large-scale empirical evaluation of llms. In *Proceedings of LREC-COLING 2024*, Turin, Italy.

Panupong Pasupat and Percy Liang. 2015. Compositional semantic parsing on semi-structured tables. *Preprint*, arXiv:1508.00305.

Yasin Peker. 2024. Medical cost analysis project. https://github.com/yasin-peker/Medical-Cost-Analysis-Project. Accessed: 2025-03-01.

Rajanand. 2024. Predict mortality/death rate. https://www.kaggle.com/datasets/rajanand/mortality. Accessed: 2025-03-01.

Anita Rostami. 2024. Montesinho forest fire prediction dataset. https://www.kaggle.com/datasets/anitarostami/montesinho-forest-fire-prediction-dataset. Accessed: 2025-03-01.

Mohan Sacharya. 2024. Graduate admissions dataset. https://www.kaggle.com/datasets/mohansacharya/graduate-admissions. Accessed: 2025-03-01.

Pavan Subhash. 2018. IBM HR Analytics Employee Attrition & Performance. Accessed: 2025-01-07.

The World Bank. 2025. World Bank Procurement Contract Awards Dataset. Accessed: 2025-03-07.

TheBloke and StabilityAI. 2023. stable-code-3b-gguf. Accessed: 2025-03-07.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. *Preprint*, arXiv:2302.13971.

Waqi786. 2024. Powerlifting data. https://www.kaggle.com/datasets/waqi786/powerlifting-data. Accessed: 2025-03-01.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022. Emergent abilities of large language models. *Transactions on Machine Learning Research*. Survey Certification.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.

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

A Pilot Task Performance

Table 12 shows the results of code-based and in-context learning system in the DataBench pilot task (Osés-Grijalba et al., 2024).

prompt,model	AVG	boolean	category	number	list[category]	list[number]
Code Prompt						
codellama-7b	27.4	45.8	16.8	43.3	14.2	17.2
codellama-13b	31.0	53.4	25.2	46.7	18.8	11.1
chatgpt3.5	**63.0**	52.7	**73.3**	**75.9**	**56.7**	**56.5**
Z-ICL Prompt						
llama-2-7b	14.8	38.4	21.7	8.9	4.3	0.8
llama-2–13b	20.7	60.9	23.3	14.8	2.7	1.6
chatgpt3.5	33.4	**65.5**	36.8	31.5	18.7	14.3

Table 12: Accuracy in the pilot task for DataBench Lite by type of answer and number of columns used, with type format errors in parentheses.