DBQR-QA: A Question Answering Dataset on a Hybrid of Database Querying and Reasoning

Rungsiman Nararatwong,[1] Chung-Chi Chen,[1]
Natthawut Kertkeidkachorn,[2] Hiroya Takamura,[1] Ryutaro Ichise[3,1]
[1]Artificial Intelligence Research Center, AIST, Japan
[2]Japan Advanced Institute of Science and Technology
[3]Tokyo Institute of Technology
{r.nararatwong, takamura.hiroya}@aist.go.jp
c.c.chen@acm.org, natt@jaist.ac.jp, ichise@iee.e.titech.ac.jp

Abstract

This paper introduces the Database Querying and Reasoning Dataset for Question Answering (DBQR-QA), aimed at addressing the gap in current question-answering (QA) research by emphasizing the essential processes of database querying and reasoning to answer questions. Specifically designed to accommodate sequential questions and multi-hop queries, DBQR-QA more accurately mirrors the dynamics of real-world information retrieval and analysis, with a particular focus on the financial reports of US companies. The dataset's construction, the challenges encountered during its development, the performance of large language models on this dataset, and a human evaluation are thoroughly discussed to illustrate the dataset's complexity and highlight future research directions in querying and reasoning tasks.

1 Introduction

Question answering (QA) is a fundamental task in the field of Natural Language Processing (NLP). Previous studies have primarily focused on text-based QA (Rajpurkar et al., 2016; Chen et al., 2021a; Gaim et al., 2023). Recently, attention has shifted towards tabular-based QA (Zhang et al., 2020a; Pal et al., 2023) and hybrid QA (Chen et al., 2020; Zhu et al., 2021; Chen et al., 2022). These studies base all experiments on the necessary tables provided as input. However, in real-world scenarios, answering questions like "What is the difference in revenue between Apple Inc. and Meta Platforms, Inc. in 2023?" (Q1) requires two steps: querying and reasoning. Specifically, models need to first **query** the revenues of the two companies in 2023 and then perform mathematical **reasoning** to answer the question. To address the gap in previous studies concerning the querying step, we propose the **Data**base **Q**uerying and **R**easoning Dataset for **Q**uestion **A**nswering (DBQR-QA).

	TAT-QA	FinQA	ConvFinQA	DBQR-QA
Reasoning	✓	✓	✓	✓
Multi-Step Reason		✓	✓	✓
Conversation			✓	✓
Multi-Hop				✓
Querying				✓

Table 1: Comparison of DBQR-QA with existing datasets.

Taking this a step further, people often ask sequential questions during discussions. For example, a follow-up to Q1 could be "Did that figure increase from 2022?" (Q2). To answer Q2, models must first identify the coreference (i.e., "that" refers to the difference in revenues between the two companies). The next steps involve querying the data for 2022, calculating the difference for that year, and comparing the answers to Q1 and Q2. Another follow-up question might be "Which company had the highest revenue in the Technology Sector in 2023?" This question, while related to Q1, does not require information from Q1 to answer. However, it demands a multi-hop querying capability, as it involves navigating from the industry to specific companies. This is considered a multi-hop query because the model must first identify the technology sector, then find companies linked to it. In DBQR-QA. This paper explores model capabilities in both scenarios.

Previous studies (Zhu et al., 2021; Chen et al., 2021b, 2022) have shown that company financial reports are excellent resources for investigating the proposed task. These reports contain numerous tables and figures, and investors and analysts frequently discuss them. Additionally, comparing reports between companies is a common practice. Following this rationale, we also base our dataset on the financial reports of US companies. Another advantage is that all these data are open access, enabling future research to extend our efforts in both querying and reasoning tasks. Table 1 provides a

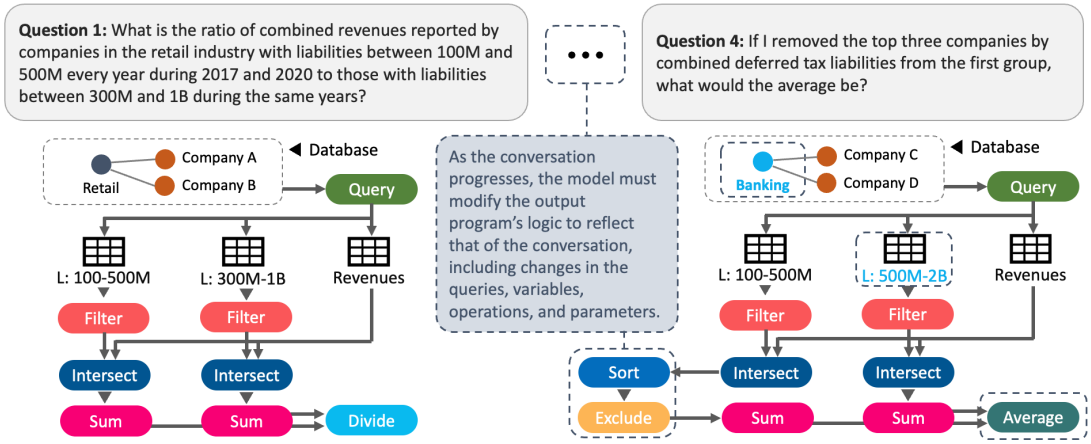

Figure 1: Example in DBQR-QA.

comparison of the proposed DBQR-QA with existing datasets. The TAT-QA dataset (Zhu et al., 2021) is designed for hybrid financial tabular and textual data, and the FinQA dataset (Chen et al., 2021b) targets numerical reasoning tasks in finance. ConvFinQA (Chen et al., 2022) introduces the concept of sequential questions based on the FinQA dataset. DBQR-QA advances this line of research by introducing querying tasks and multi-hop QA, moving closer to real-world application scenarios.

In this paper, we detail the construction of DBQR-QA and the challenges encountered in building this dataset. We also investigate the performance of large language models (LLMs) on this dataset and compare its difficulty with that of existing datasets. Furthermore, we provide a human evaluation to highlight issues in both querying and reasoning steps, identifying potential challenges for future research.

2 Related Work

Question-answering has posed a long-standing challenge for language models, prompting the proposal of numerous new datasets to explore and enhance the capabilities of these models. This section presents works associated with the three aspects of our dataset: Tables, reasoning, and conversation.

LLMs have made significant advances in reasoning QA, as demonstrated by the DROP dataset (Dua et al., 2019) for reading comprehension and arithmetic QA, the GSM-8K dataset (Cobbe et al., 2021) for grade-school math word problems, the MMLU benchmark (Hendrycks et al., 2021) for multiple-domain multiple-choice questions, and the NumHG (Huang et al., 2024) for number-focused headline generation.

Tabular QA is another area that requires reasoning skills. Notable datasets in this field include the TAT-QA dataset (Zhu et al., 2021) for hybrid financial tabular and textual data, the FinQA dataset (Chen et al., 2021b) for numerical reasoning task in finance, and the FeTaQA dataset (Nan et al., 2022) for free-form table QA. Built on top of TAT-QA and FinQA, our dataset expands the scope of reasoning by incorporating querying and reasoning in problem-solving.

Numerous conversational QA datasets target large knowledge bases as they allow for diverse multi-hop questions. Some notable datasets in this domain include SQA (Iyyer et al., 2017) for Wikipedia tables, CSQA (Saha et al., 2018) for reasoning over a knowledge base, and ConvQuestions (Christmann et al., 2019) covering five domains. Non-KB QA datasets also pose significant challenges, for instance, CoQA (Reddy et al., 2019) for machine comprehension and QuAC (Choi et al., 2018) in dialog contexts. Despite extensive research on conversational QA for many years, its tabular and reasoning aspects still require further attention, as evident in ConvFinQA (Chen et al., 2022). This dataset explores the chain of numerical reasoning in single-table conversational QA. Our dataset raises the complexity in table manipulation, cross-question chain of reasoning, and multiple-table QA from a financial knowledge graph.

3 Dataset Construction

3.1 Overview of DBQR-QA

Figure 1 presents an example of the newly proposed DBQR-QA dataset. Concurrently, Figure 2 provides a comparative view with examples from previously established datasets. It is noteworthy

that each question within the TAT-QA dataset only requires a single-step operation. The majority of questions in both the FinQA and ConvFinQA datasets necessitate the execution of one to four steps for calculation, invariably involving the use of two numbers sourced either directly from the data or derived from preceding steps. The proposed DBQR-QA dataset incorporates questions that require both database querying and complex multi-step table manipulations. These tasks are complicated further by a multi-branch chain of reasoning, where each question in the sequence introduces, alters, or eliminates queries, variables, operations, and parameters. This progressive complexity not only challenges the model's capacity for memorization but also tests its ability to adapt and refine its logic throughout the conversation.

Here is an example of the question and gold label for this question in our dataset:

Question: During the period from 2016 to 2021, what were the top two years Itron reported the highest R&D Expense?

Gold label:

```
company_1 = 'ITRON, INC.'
concept_1 = 'us-gaap:ResearchAndDevelopmentExpense'
var_q1_p1 = get_company_facts(company_1, concept_1,
start=year_1, end=year_2)
var_q1_p2 = sort(var_q1_p1, ascending=False,
axis='columns', by=[[company_1, concept_1]])
var_q1_p3 = k_end(var_q1_p2, axis='rows',
direction='first', k=2)
var_q1_p4 = headers(var_q1_p3, axis='columns',
level=0)
```

3.2 Question Preparation

The questions in the proposed DBQR-QA were sourced from the TAT-QA (Zhu et al., 2021) and FinQA (Chen et al., 2021b) datasets, which were manually crafted and annotated by financial experts. The limited variety of reasoning operations within these datasets resulted in many questions bearing similarities. We addressed this by grouping similar questions and developing a template-based representation. Specifically, we created 30 text-to-template examples to identify financial concepts, temporal elements (e.g., years), numerical values, and units (e.g., millions) from a question, then utilized BART (Lewis et al., 2020) to extract these elements and generate templates. For instance, the question "What was the total intangible assets in 2019?" was abstracted to "What was the total [concept] in [year]?" This abstraction process involved calculating string similarity scores, grouping the templates accordingly, and refining them to suit the graph database context, beyond mere tabular data.

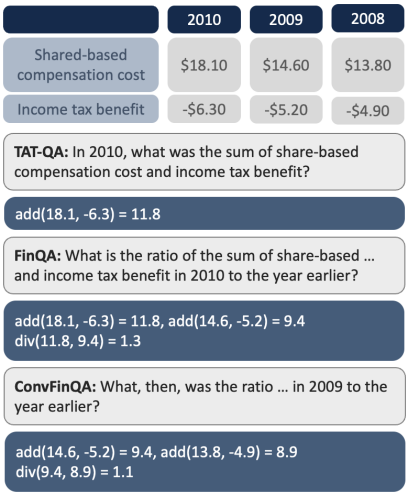

Figure 2: Comparison of the three relevant datasets.

Similar to ConvFinQA, our dataset transforms questions from FinQA into a conversational format but stands out by incorporating table manipulation throughout the reasoning process. Different from ConvFinQA which only uses six simple arithmetic operators, such as addition, subtraction, multiplication, and division, our operators include 26 operators in Pandas DataFrame. [1] This setting enables more complex and expressive queries than previous datasets. By leveraging Pandas' extensive functionality, our approach also provides scalability for a wide range of applications.

Upon establishing the question templates, we populated them with entities (e.g., companies), financial concepts, and numerical data, aligning the financial terminology with the US-GAAP taxonomy[2]. We prioritized terms based on their occurrence frequency in the questions and selected those represented in the graph to ensure the correctness of the generated answers. We then defined a set of operations and combined them to formulate a program for each question, marking the initial stage of annotation.

3.3 Automatic-Answer Annotation

To utilize the responses annotated by financial experts in TAT-QA and FinQA, we leveraged Re-

[1] https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.html

[2] https://www.sec.gov/info/edgar/edgartaxonomies

Question Type (N)	Example
Simple (100)	During the six-year period ending 2016, which year should be excluded so that Brown & Brown Insurance's average deferred revenues over the remaining years reaches its highest value possible?
Complex (100)	Suppose we can choose to set a minimum threshold of 100M, 300M or 500M for purchase obligations in 2019, what is the maximum average contractual obligations in the same year we can get from companies on the list?
Multi-Table (50)	Considering all companies, which average is higher? 0) earnings per share (basic) through 2019 and 2021 or 1) earnings per share (diluted) through 2020 and 2022?
Multi-Hop (100)	Which of the following industry-level criteria yields higher 2018-2020 average net cash provided by operating activities for the industries involved: 0) industries with year-on-year increase in average net cash used in financing activities throughout the period, or 1) at least once during the period?
Instruction (50)	Get the 2018-2021 current federal income taxes of companies with common stakeholders to companies in the real estate investment trusts industry. Get another list for marketing and advertising expense during 2019 through 2022, but all companies on the second list must reported current federal income taxes above 100K every year during the period. What is the difference between the overall averages of the two lists?

Table 2: Example of different types of questions in DBQR-QA. N represent the number in the data set.

portKG, a knowledge graph constructed from financial report tables formatted as XBRL documents[3]. This integration facilitates complex tasks requiring extensive data interlinkage by storing the data within the graph. The graph's querying mechanism subsequently converts the results into tables, enabling manipulation during reasoning steps. Through ReportKG, automatic-answer annotations for generated questions are accessible. For instance, a question from TAT-QA, "How much revenue came from LinkedIn in 2018?" transforms into "How much profit came from Apple in 2023?" in our dataset. The annotation process in TAT-QA involves extracting a triple (revenue, LinkedIn, 2018) to respond to the question, whereas, in our context, the automatic-answer annotation comprises a triple (profit, Apple, 2023), thus providing a preliminary answer.

3.4 Answer Verification

We employed Amazon Mechanical Turk workers to verify our automatic-answer annotations. Their task was to read the questions and create a program (a series of tabular operations) using data queried from the database. The system then compared their program's answer against ours. If there was a mismatch, the workers needed to explain which program – or whether the question – was incorrect. This approach mitigated the risk of our interpretation of the questions influencing theirs, as would have been the case if we had requested them to verify our programs.

To control the annotation quality, the eligible workers attained a minimum score of 70% across three qualification tests. Additionally, they offered adequate explanations for all answer discrepancies, demonstrating their ability to identify the issues. We considered a question valid by a majority consensus. We analyzed the workers' feedback and determined the questions that had been identified as incorrect, such as those involving the possibility of a negative value measuring the "difference" between two quantities (e.g., how much different A is from B), necessitated additional clarification. We finalized minor revisions to the questions, completing our dataset construction.

[3] https://xbrl.us/

4 Dataset Statistics

With advancements in NLP, zero-shot learning has emerged as a prominent method in the study of LLMs. Our proposed model, DBQR-QA, is specifically designed for this context, aiming to evaluate the ability of LLMs to generate code for querying and reasoning. Instead of creating a large dataset for supervised learning, we have developed a carefully constructed test set for precise evaluation. In the ConvFinQA benchmark, the test set consists of 434 instances. In line with these benchmarks, DBQR-QA provides a similar scale with 400 QA pairs.

The proposed DBQR-QA is categorized into five subsets based on question type and complexity, and it introduces a structured variety of question types aimed at examing querying and reasoning skills. These categories are crafted to delve into the complex aspects of financial datasets, focusing on distinct objectives and complexities. Below, an introduction to the five unique question types in our dataset is provided, alongside examples in Table 2

to demonstrate their nature and the analytical skills required.

Type 1: Simple Query with Specific Companies (Simple)

This type entails straightforward questions about specific companies, necessitating the direct extraction of data and basic arithmetic for solution derivation. A representative question might involve optimizing financial metrics over a specified period, like identifying a year to exclude to maximize a named insurance company's average deferred revenues.

Type 2: Complex Query with Unspecified Companies (Complex)

Here, the complexity increases with the companies of interest not specified, incorporating conditional thresholds for financial metrics. The task involves selecting criteria to maximize or optimize a financial parameter among a group of companies. For instance, figuring out the year with the maximum average contractual obligations, based on varying minimum thresholds for purchase obligations.

Type 3: Reasoning Steps Requiring Multiple Tables (Multi-Table)

This category requires synthesizing data from multiple tables to answer questions involving comparative analysis or financial metric aggregation over various periods or conditions. It assesses the ability to navigate and reason with interconnected datasets, such as comparing average earnings per share across different years while accounting for basic versus diluted shares.

Type 4: Multi-hop Query (Multi-Hop)

Multi-hop queries necessitate a sequence of logical steps and deductions to conclude. These questions often involve industry-level analysis, such as comparing averages or trends across different criteria or periods. An example question might ask which industry-level criterion results in a higher average net cash provided by operating activities, requiring knowledge of temporal trends and industry characteristics.

Type 5: Instruction QA (Instruction)

Instruction-based questions present complex scenarios that guide the analyst through a series of data retrieval and analysis tasks across various dimensions, like time, industry, and financial metrics. These questions mimic real-world data analysis tasks, requiring high comprehension and the ability to follow multi-step instructions to compare and contrast averages or identify trends among specific company groups.

These five question types collectively offer a thorough framework for querying and reasoning skills with financial datasets, spanning from simple arithmetic to intricate reasoning and multi-step problem-solving.

5 Experimental Setup

5.1 Task Design

In our experiment, we aim to ask LLMs to answer the given question by (1) querying data from ReportKG and (2) performing mathematical reasoning with the queried data. For querying, we ask LLMs to generate Cypher programs, where Cypher is a graph query language that lets the user retrieve data from the graph. For reasoning, we ask LLMs to generate Python programs by using the Pandas package for reasoning.[4]

5.2 Implementation Details

In our experiment, we benchmark the performance of Llama 2, PaLM 2, GPT-3.5 and GPT-4. Recognizing that multiple programs can be generated for a single question, we manually evaluate and execute each program to determine its correctness. We classify the outcomes into three categories: pass (producing the correct answer), fail (yielding an incorrect answer), and crash (inability to execute).

5.3 Manual Evaluation

This section describes the rules for manual evaluation. There are two types of answers: text and number. An answer can be a single value or a set (of texts or numbers). Textual answers can be a comparison (higher, lower, or equal) or entities, including financial concepts defined in the taxonomy, companies, persons, industries, and states. There are no other types of textual answers. A human evaluator must disregard other contextual output or any information apart from the answer, whether they are correct or not. If the answer is a set of values, the label and predicted sets must be identical (the orders do not matter), i.e., all values must be in the answer, and the answer must not contain any other values. If the set of values is of particular years or entities, e.g., a set of company revenues

[4]Please refer to Appendix H for the prompt.

	Simple			Complex			Multi-Table			Multi-Hop			Instruction			Overall		
	Pass	Fail	Crash	Pass	Fail	Crash	Pass	Fail	Crash	Pass	Fail	Crash	Pass	Fail	Crash	Pass	Fail	Crash
Llama 2	5	57	38	1	18	81	0	18	82	0	28	72	0	20	80	0.5	28.7	70.3
PaLM2	13	15	72	9	37	54	2	34	64	0	19	81	0	43	57	8.7	35.0	56.3
GPT-3.5	41	38	21	26	40	34	6	38	56	3	56	41	0	60	40	8.7	57.1	34.2
GPT-4	70	18	12	48	29	23	14	44	42	11	80	9	0	43	47	16.1	43.2	40.5

Table 3: Experimental results (in percentage).

	LC	LP	SC	SP	D	ID	IE	MI	N	NC
PaLM 2	29.9	13.5	17.4	25.8	0.0	0.0	0.0	0.0	4.5	9.0
GPT-3.5	40.6	21.1	15.0	0.00	3.0	3.0	7.5	5.3	4.5	0.0
GPT-4	27.2	14.8	25.9	4.9	2.5	3.7	0.0	3.7	16.1	1.2

Table 4: Error type analysis (in percentage).

during a period, the predicted values must explicitly describe all correct years or entities.

5.4 Automatic Evaluation

Previous works, such as TAT-QA, FinqA, and ConvFinQA, developed automatic heuristic evaluators to evaluate the outputs. However, the programs generated by LLMs through prompting may include additional information, making it more challenging to compare automatically with absolute certainty. We found three common types of such cases in our dataset. (1) The model outputs the correct answer but in different wording than the label. (2) The model outputs the context that may confuse a heuristic evaluator. (3) The model outputs its chain of thought (even without prompting specifically). To ensure the reliability of automatic evaluation, we implemented and tested three evaluators against human judgments.

The first metric is heuristic and serves as a baseline. The rules are as follows:

1. *Single numeric answer*: Detect a number in the prediction. There must be only one number (excluding years). The prediction is correct if it matches the label to the second decimal digit.

2. *Set of numbers*: The counts of the numbers in prediction and the label must be the same. If true, all numbers in the label must have unique matching pairs.

3. *Year, a set of year, string, and a set of strings*: All the years/strings in the label must be in the prediction.

The second metric compares the reference answer and the prediction using the BERTScore through the models' embeddings (Zhang et al., 2020b). We used Microsoft's DeBERTa XLarge NMLI [5] as the base model, as recommended by the authors, due to its highest correlation with human evaluation. Given the reference answer and the prediction, the BERTScore algorithm computes the accuracy, precision, and F1 scores. We added a binary classification, where the output is 0 if the F1 score is less than 0.5, else 1.

The third metric employs GPT-4 as the evaluator. We first ask the model to compare the generated answer to human annotation (see Appendix B for the prompt). Next, we asked the model to score the answer by modifying the last part of the prompt to: "On a scale of 0 to 10, 0 = not at all and 10 = same, how similar are the two answers? Answer an integer number from 0 to 10 only. Do not explain or output anything else."

6 Experimental Result

6.1 From Querying to Reasoning

Table 3 presents our findings. Notably, GPT-4 consistently outperforms other models regardless of question complexity. Additionally, our results demonstrate that the proposed DBQR-QA poses greater challenges to LLMs.

In order to take an in-depth look at the problems that LLMs face when addressing the questions in the proposed dataset, we manually checked the problems that occur in Simple and Complex questions. This testing involves analyzing the output exceptions. We also checked the results in the intermediate steps. The following are the common problems we found:

1. *Logical error in the Python script (LP) or Cypher (graph) query (LC)*: This problem is due to the model misinterpreting the question or making mistakes in its reasoning steps.

[5] https://huggingface.co/microsoft/deberta-xlarge-mnli

	Simple			Complex			Multi-Table			Multi-Hop			Instruction			Overall		
	Pass	Fail	Crash	Pass	Fail	Crash	Pass	Fail	Crash	Pass	Fail	Crash	Pass	Fail	Crash	Pass	Fail	Crash
Llama 2	12	61	27	4	53	43	0	28	72	3	63	34	0	40	60	3.4	48.9	47.6
PaLM2	18	44	38	13	39	48	0	19	81	1	46	53	3	17	80	4.5	38.2	57.1
GPT-3.5	49	49	2	29	61	10	3	56	41	3	55	42	3	63	34	4.2	46.6	49.2
GPT-4	75	22	3	79	20	1	12	60	28	5	60	35	0	53	47	18.2	52.4	26.8

Table 5: Experimental results of generating programs when given tables (in percentage).

2. *Syntax or runtime error in the Python script (SP) or Cypher query (SC)*: This problem encompasses various instances, such as the model misunderstanding the languages' syntax, the code referring to undeclared variables, and the model misinterpreted data structure from the queries or prior operations.

3. *Negative output from the difference operation (D)*: The program outputs a negative answer when asked for differences between numbers.

4. *Inverse division (ID)*: Reversed numerator and denominator in division.

5. *Incorrect entity (IE)*: The query or code refers to irrelevant entities.

6. *Manual input (MI)*: Require manual input.

7. *Null handling (N)*: Fail to handle NULL.

8. *No code generated (NC)*

Table 4 details the distribution of each error type. It's worth noting that models exhibit a significant number of errors when querying tables and data using Cypher, while fewer mistakes occur in result calculations.

6.2 Results of Reasoning

From our error analysis in Section 6.1, it is evident that table queries, denoted by LC in Table 4, represent the predominant error type among the ten identified. In this section, we explore a modified experimental setting where models are not required to query tables but are directly provided with tabular data. This adjustment allows us to better assess a model's ability to generate programs for calculations without the necessity of table querying.

Table 5 shows that model performances improve across all metrics, regardless of the specific model or the complexity of the dataset subset in use. This improvement highlights the frequent failures attributed to incorrect table-querying programs. Notably, the accuracy of GPT-4 with the Complex subset significantly surpasses its performance when tasked with generating a complete program, emphasizing the critical role of correct data querying. However, GPT-4's performance remains consistent between the Simple and Complex subsets for raw calculations. This consistency suggests that our dataset does not introduce unrealistic or overly complex calculations, aiming instead to replicate real-world tasks commonly encountered in professional settings, from data querying to calculations.

6.3 Automatic Evaluation

We tested our three automatic evaluators against human annotation. Table 6 shows the models' performances on the two tasks. The numbers in parentheses are the differences between the models' predictions and human-annotated labels. Across all the experimental settings, BERTScore is much less predictive of human evaluation than the baseline and GPT-4. We found that the embeddings of numbers are similar to each other regardless of whether or not the numbers are the same. This issue led to the model outputting high F1 scores for incorrect numeric answers. While the baseline evaluator is highly accurate when most answers are wrong, its accuracy drastically drops as it fails to recognize correct answers.

Table 7 summarizes the metrics' performances by accuracy, precision, recall, and F1 score for binary classification and root mean squared error (RMSE) for continuous score values. The ground-truth values for RMSE are 0 for human-annotated labels indicating incorrect prediction and 1 for the correct ones. Based on the results shown in both tables, we conclude that GPT-4 is the best approach for automatic evaluation due to its low inaccuracy.

7 Discussion

Based on our experimental results, GPT-4 exhibits satisfactory performance in both Simple and Complex subsets. However, its performance is inferior in the remaining subsets. Therefore, in this section, we focus on discussing the Simple and Complex subsets to understand the nuances of LLMs' performances on the proposed dataset.

7.1 Coreference Resolution

A pivotal facet of conversational question answering is the model's ability to undertake coreference resolution. For instance, when presented with a

Task	Model	Gold Acc (Human)	Base Acc (Heuristic)		BERT				GPT-4			
					Acc		μ F1	σ F1	Acc		μ Scale	σ Scale
Full	Llama 2	0.03	0.02	(-0.01)	0.79	(+0.76)	0.60	0.15	0.04	(+0.01)	0.05	0.19
	PaLM 2	0.11	0.10	(-0.01)	0.80	(+0.69)	0.61	0.17	0.10	(-0.01)	0.12	0.31
	GPT 3.5	0.34	0.24	(-0.09)	0.86	(+0.52)	0.66	0.17	0.32	(-0.02)	0.36	0.45
	GPT 4	0.59	0.47	(-0.11)	0.93	(+0.34)	0.69	0.17	0.55	(-0.04)	0.60	0.45
Reasoning	Llama 2	0.08	0.07	(-0.01)	0.70	(+0.62)	0.58	0.16	0.07	(-0.01)	0.11	0.27
	PaLM 2	0.15	0.11	(-0.04)	0.78	(+0.62)	0.62	0.18	0.17	(+0.01)	0.22	0.39
	GPT 3.5	0.39	0.36	(-0.03)	0.98	(+0.59)	0.81	0.16	0.41	(+0.02)	0.48	0.48
	GPT 4	0.77	0.64	(-0.13)	0.92	(+0.15)	0.71	0.18	0.75	(-0.02)	0.78	0.37

Table 6: Comparison of human and automatic evaluations. The accuracies are execution accuracies. The GPT-4 scale ranges from 1 to 10 (the reported values are divided by 10).

Metric	Type	Acc	P	R	F1	RMSE
Base	Binary	0.93	0.95	0.90	0.92	-
BERT	Binary	0.42	0.60	0.56	0.41	-
	F1	-	-	-	-	0.56
GPT-4	Binary	0.98	0.98	0.97	0.97	-
	Scale	-	-	-	-	0.20

Table 7: Evaluation of the automatic metrics compared to human evaluation (micro F1). The accuracy, precision (P), recall (R), and F1 scores evaluate the metrics' binary output. The root mean squared error evaluates the metrics' continuous output between 0 to 1.

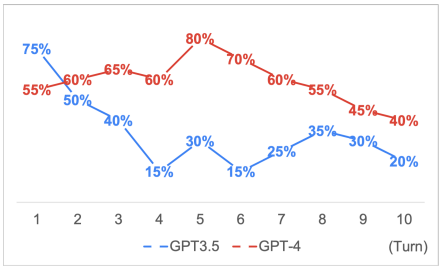

Figure 3: Pass ratio by turn.

follow-up question, "*What about 2021 and 2022?*" succeeding the query "*Between 2019 and 2020, what was the change in Gibraltar Industries gross profit?*", models must discern that the inquiry pertains to the "gross profit". To delve into the efficacy of models in this dimension, we executed a turn-based evaluation, the outcomes of which are illustrated in Figure 3.

The data suggests that GPT-3.5 excels during the initial turn of conversation but its performance tapers off in subsequent turns. Conversely, GPT-4 showcases commendable results during the first five turns, after which there's a discernible decline. These observations highlight two key points: firstly, coreference resolution remains a formidable challenge for LLMs. Secondly, the mean turn count for the existing dataset, ConFinQA, stands at a mere 3.67. This duration is inadequate to rigorously assess models like GPT-4. From this vantage point, our proposed DBQR-QA emerges as a more robust benchmark for future studies.

7.2 Fine-Grained Question Type Analysis

We further employ two strategies to categorize the questions into more fine-grained categories. The first method classifies the questions according to the functions adopted by their solution programs, whereas the second method depends on the judgment of the annotators.

We grouped functions defined for this dataset into six groups as follows:

1. *Querying functions (QUER)* consist of two functions: One for querying facts regarding a specific company and the other for querying facts associated with non-specified companies based on particular criteria.

2. *Logical functions (LOGI)* consist of nine functions for table manipulation. These operations include intersection, union, transpose, conditional replacement, conditional filtering, comparison, merging, excluding, and column stacking.

3. *Ordering functions (ORDR)* consist of five functions: sorting, reversing the table along the row or column axis, selecting rows or columns, selecting the nth element along the axis, and selecting the first or last k items.

4. *Arithmetic functions (CALC)* consist of seven functions: addition, subtraction, division, absolute operation, multiplication by a constant, subtraction by a constant, and division by a constant.

5. *Aggregation functions (AGGR)* consist of two functions: averaging and counting.

6. *Listing functions (LIST)* consist of two functions: One lists the table's header, and the other indicates an empty output.

The annotators' judgment aims to provide a more fine-grained analysis and consists of additional categories as follows:

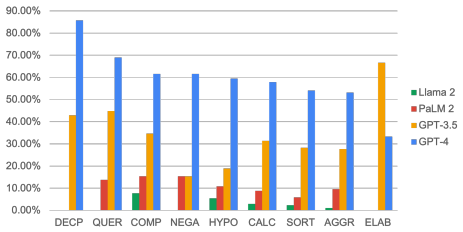

Figure 4: Performance by question type.

	TAT-QA	FinQA	ConvFinQA	DBQR-QA
GPT-4	61.73	68.79	76.48	59.00
Human	-	91.16	89.44	74.30

Table 8: Performance of GPT-4 and human in existing datasets and DBQR-QA.

1. *SORT* for questions that involve sorting numbers by financial concepts, companies, industries, years, among others.

2. *HYPO* for questions that ask for calculation in a hypothetical scenario. For example: "What if 2018 was twice what was reported, how much more can be added to 2022 to maintain the 50% margin?"

3. *COMP* for questions that ask to compare two or more numbers along one axis of a table, for example, companies' revenues in the year of interest compared to the year prior.

4. *NEGA* for questions that involve negation, i.e., "not." For example, "which years can I remove that will not increase the ratio?"

5. *DECP* for questions that intentionally deceive the model, for example, by referring to a financial concept unrelated to the one asked in the conversation.

6. *ELAB* for questions that ask to clarify the calculation. For example, "which companies are included in the calculation [of the previous question]?"

We delved deeper into the performance of models across various question categories. The outcomes of this analysis are illustrated in Figure 4. Notably, GPT-4 surpasses its counterparts in the majority of question categories. However, given that there are a mere three questions categorized under Elaboration (ELAB), the differentiation in performance for this type is marginal. Within the array of question types, GPT-4 showcases superior performance on DECP (Deceptive), QUER (Complex Queries), and COMP (Comparison) categories. The annotations for these question types will be included with the proposed MTCQA dataset, offering a foundation for future research to develop specialized solutions targeting each distinct question category.

7.3 Human Performance

Table 8 presents the performance comparison of GPT-4 and humans on existing datasets and the proposed DBQR-QA dataset. The results indicate that accurately answering queries in the DBQR-QA dataset poses greater challenges for both GPT-4 and humans, highlighting the need for future research to enhance performance in either the querying or reasoning aspects of the dataset.

8 Conclusion

By introducing a comprehensive dataset tailored to evaluate the querying and reasoning capabilities of LLMs, the DBQR-QA dataset challenges models with real-world scenarios that require both database queries and multi-step reasoning processes, reflecting the complexities encountered in professional financial analysis. Experimental results reveal that while LLMs like GPT-4 demonstrate potential in certain subsets, significant challenges remain, particularly in subsets requiring advanced reasoning and multi-step querying capabilities. This observation underscores the ongoing need for improvements in models and specialized training to bridge the gap between current capabilities and the demands of real-world data analysis tasks. Lastly, although our dataset focuses on the financial domain, interested researchers and developers can apply the creation and evaluation processes to use cases where tabular reasoning over a database can benefit question-answering. Companies and organizations can also apply our approach to build a QA system to conduct complex analyses of their databases for various purposes, for example, inventory/logistic/resource management, fraud/irregularity detection, system optimization, and supporting information for policy/decision-making. In addition to the dataset, all fine-grained annotations for analysis will be shared under the CC BY-NC-SA 4.0 license.[6]

Limitation

The paper's limitations can be summarized as follows: First, while our study adopts financial data in line with prior research, this specificity means that our findings are predominantly tailored to financial documents. Explorations into other domains rich in database, such as scientific research or the clinical realm, would be beneficial in subsequent studies.

[6]Research materials are available at: https://www.ai.iee.e.titech.ac.jp/DBQR-QA/

Second, our primary emphasis is on assessing the capabilities of LLMs in a zero-shot setting. Future investigations might consider broadening the scope to explore our dataset under varied experimental paradigms. Third, consistent with prior works, our dataset is confined to English. It's worth noting that LLMs might exhibit different behaviors across different languages, and we advocate for this consideration in future endeavors.

Acknowledgement

This study is partially based on the results obtained from a project JPNP20006, commissioned by the New Energy and Industrial Technology Development Organization (NEDO).

References

Chung-Chi Chen, Hen-Hsen Huang, and Hsin-Hsi Chen. 2021a. Nquad: 70,000+ questions for machine comprehension of the numerals in text. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 2925–2929.

Lingjiao Chen, Matei Zaharia, and James Zou. 2023. How is chatgpt's behavior changing over time? *arXiv preprint arXiv:2307.09009*.

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.

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan Routledge, and William Yang Wang. 2021b. FinQA: A dataset of numerical reasoning over financial data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3697–3711, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zhiyu Chen, Shiyang Li, Charese Smiley, Zhiqiang Ma, Sameena Shah, and William Yang Wang. 2022. ConvFinQA: Exploring the chain of numerical reasoning in conversational finance question answering. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6279–6292, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wentau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018. QuAC: Question answering in context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2174–2184, Brussels, Belgium. Association for Computational Linguistics.

Philipp Christmann, Rishiraj Saha Roy, Abdalghani Abujabal, Jyotsna Singh, and Gerhard Weikum. 2019. Look before you hop: Conversational question answering over knowledge graphs using judicious context expansion. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, CIKM '19, page 729–738, New York, NY, USA. Association for Computing Machinery.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378, Minneapolis, Minnesota. Association for Computational Linguistics.

Fitsum Gaim, Wonsuk Yang, Hancheol Park, and Jong Park. 2023. Question-answering in a low-resourced language: Benchmark dataset and models for Tigrinya. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11857–11870, Toronto, Canada. Association for Computational Linguistics.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Jian-Tao Huang, Chung-Chi Chen, Hen-Hsen Huang, and Hsin-Hsi Chen. 2024. NumHG: A dataset for number-focused headline generation. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 12323–12329, Torino, Italia. ELRA and ICCL.

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

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*,

pages 7871–7880, Online. 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, Mutethia Mutuma, Ben Rosand, Isabel Trindade, Renusree Bandaru, Jacob Cunningham, Caiming Xiong, Dragomir Radev, and Dragomir Radev. 2022. FeTaQA: Free-form table question answering. *Transactions of the Association for Computational Linguistics*, 10:35–49.

Vaishali Pal, Andrew Yates, Evangelos Kanoulas, and Maarten de Rijke. 2023. MultiTabQA: Generating tabular answers for multi-table question answering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6322–6334, Toronto, Canada. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392.

Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. CoQA: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

Amrita Saha, Vardaan Pahuja, Mitesh M. Khapra, Karthik Sankaranarayanan, and Sarath Chandar. 2018. Complex sequential question answering: Towards learning to converse over linked question answer pairs with a knowledge graph. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18. AAAI Press.

Shuo Zhang, Zhuyun Dai, Krisztian Balog, and Jamie Callan. 2020a. Summarizing and exploring tabular data in conversational search. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, page 1537–1540, New York, NY, USA. Association for Computing Machinery.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020b. BERTScore: Evaluating text generation with BERT. In *Proceedings of the 2020 International Conference on Learning Representations (ICLR)*.

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

Concepts
Question: What is the amount of stated capital outstanding on December 31, 2019 Mentions: stated capital outstanding
Question: In which year was selling, general and administrative less than 100,000 thousands? Mentions: selling, general and administrative
Question: What is the difference between 2019 average rate of inflation and 2019 average rate of increase in salaries? Mentions: rate of inflation; rate of increase in salaries
Question Templates
Question: What is the amount of stated capital outstanding on December 31, 2019 Masked: What is the amount of [concept] on [date]
Question: In which year was selling, general and administrative less than 100,000 thousands? Masked: In which year was [concept] less than [number] thousands?
Question: What is the difference between 2019 average rate of inflation and 2019 average rate of increase in salaries? Masked: What is the difference between [year] average [concept] and [year] average [year]?

Table 9: Examples used for BLOOM 7B to extract concepts and question templates.

A Question Generation

The questions are all human-generated. However, we used BLOOM 7B[7] to find common financial concepts and question templates in the TAT-QA and FinQA datasets. This data helps us decide what concepts and types of questions we should include in the annotation. We gave BLOOM 30 examples for each task and ran the model on all samples in the datasets. Table 9 lists some examples we used. We then search the US GAAP and company-defined taxonomies for the extracted concepts. Table 10 shows the top five concepts found in both datasets. Table 11 shows the top five templates that start with "what," which account for 82.60% of all questions.

[7] https://huggingface.co/bigscience/bloom-7b1

Exact Match				80% Match			
US GAAP		Custom		US GAAP		Custom	
Concept	Counts	Concept	Counts	Concept	Counts	Concept	Counts
Revenue	369	Revenue	370	Retail revenue	455	Contract revenue	463
Unrecognized tax benefit	110	Net revenue	274	Revenue	372	Rental revenue	456
Gross profit	107	Net sale	254	Operating expense	233	Revenue	371
Interest expense	107	Operate income	112	Net cash provide by use in operating activity	210	Fee revenue	312
Contractual obligation	99	Unrecognized tax benefit	110	Net cash provide by use in investing activity	199	Rent revenue	305

Table 10: Top five concepts by numbers of mentions in the TAT-QA and FinQA datasets. The "80% Match" indicates mentions with at least 80% match to the concepts' title in the taxonomy. The custom column is for company-defined concepts.

Counts	Template	Example
496	What was [concept] in [year]?	What was the amount of raw materials in 2018?
218	What was the increase / (decrease) in [concept] - insurance	What was the increase / (decrease) in the net income - insurance segment from 2018 to 2019?
157	What was the [concept] in [entity] in [year] and	What was the Cost-reimbursement and fixed-price-incentive-fee in Defense Solutions, Civil and Health respectively?
144	What was the total [concept] in [year]?	What is the increase/ (decrease) in Loss per common share - basic and diluted from 2018 to 2019?
118	What does [concept] in [year] include?	What was the total intangible assets in 2019?

Table 11: Top five question templates for "what" questions (generated by BLOOM 7B) with at least 80% match.

B Automatic Evaluation

We used the following prompt for our GPT-4 evaluator, which compared the models' outputs to human annotation:

```
[System]
You are an evaluator. Given a series of
conversational questions, your task is to
compare an answer to the last question
predicted by an AI to an answer labeled by
a human.

[User]
Consider the following conversational
questions: {{questions}}

Compare the following answers to the last
question in the above conversation.
AI's answer: {{prediction}}
Human's answer: {{reference}}

Should we consider the two answers the
same? Answer "Yes" or "No" only. Do not
explain or output anything else.
```

C Fine-grained Question Type

Table 12 and Table 13 show the statistics of the simple and complex sections of the dataset. Note that a question could fit into several fine-grained categories.

Code	Description	Questions
QUER	Querying functions	200
LOGI	Logical operations (e.g., union)	143
CALC	Arithmatic calculation	135
ORDR	Ordering (e.g., sort, nth item)	129
AGGR	Aggregation (e.g., averaging)	129
LIST	Listing operations (e.g. headers)	49

Table 12: Fine-grained categories based on functions adopted by the solution programs.

Code	Description	Questions
CALC	Arithmetic calculation	102
AGGR	Aggregation (e.g., averaging)	94
SORT	Sorting	85
HYPO	Hypothetical questions	37
COMP	Comparison	26
NEGA	Negation	13
DECP	Deceptive questions	7
ELAB	Elaboration	3

Table 13: Fine-grained categories based on the annotators' judgment.

D Operations

Table 14 summarizes DBQR-QA's operation extension (n=26) compared to the TAT-QA (n=10), FinQA (n=10), and ConvFinQA (n=6) datasets, where n represents the number of operations.

TAT-QA
Arithmetic: Sum, count, average, multiply, divide, subtract, change ratio
Text: Span-in-text, cell-in-table, spans
FinQA
Arithmetic: Add, subtract, multiply, divide, exponential
Comparison: Compare
Table*: Sum, average, max, min
ConvFinQA
Arithmetic: Add, subtract, multiply, divide, exponential
Comparison: Compare
DBQR-QA
Arithmetic: Sum, subtract, divide, absolute
Comparison: Compare
Constant: Add, subtract, multiply, divide
Aggregation: Average, count
Selection: Selector, filter, first/last K, Nth item
Sorting: Sort
Multi-table: Merge, stack columns, intersect, union, exclude
Table Headers: Header
Edit: Reverse, replace, transpose, remove NaN

Table 14: Comparison of operations to other related datasets. *FinQA's table operations only apply to one table row. DBQR-QA supports full tabular operations.

E Instructions for Annotators

The description of the task given to the annotators is as follows:

1. Read a series of 5 - 10 conversational questions.
2. Build a program that performs numerical operations for each question using our annotation tools.
3. Locate numbers in a table that represent financial concepts mentioned in the questions.
4. Check our pre-annotated program against your submission.

The requirements for the Amazon Mechanical Turk workers are as follows:

1. Basic mathematical reasoning over tabular data (similar to applying formulas in Microsoft Excel).
2. Basic understanding of financial concepts.
3. Latest version of Google Chrome, Safari, or Microsoft Edge running on a desktop PC.

The completion time estimation given to the annotators is as follows:

1. If this is your first time seeing this task, you're about to take the first test, which:
    - consists of 5 questions,
    - requires you to watch a 4-minute training video and read additional instructions,
    - gives you hints on how to build programs, and
    - takes about 40 minutes on averageto complete.
2. If not, it's either:
    - your second test (5 questions) or
    - your third test or the actual annotation task (10 questions).

Stage	Task	Questions	Completion Time Estimation
1	Training & testing	5	40 mins
2	Training & testing	5	40 mins
3	Testing	10	1 hour
4	Actual task	10	$\leq$ 1 hour

Table 15: Completion time estimation for each stage of the annotation process.

Category	Llama 2	PaLM 2	GPT 3.5	GPT 4
ORDR	2.33	6.98	31.01	62.79
LIST	4.08	8.16	26.53	61.22
AGGR	3.10	11.63	27.91	59.69
QUER	3.00	11.00	33.50	59.00
LOGI	2.80	7.69	26.57	58.74
CALC	3.70	8.89	32.59	58.52

Table 16: Models' performance analysis by function type (in percentage) on the simple and complex sets.

Table 15 summarizes the completion time estimation, also shown to the annotators.

The rejection criteria given to the annotators for assignment submission are as follows:

1. Not giving a clear explanation when prompted (for example, we'll ask you to identify and explain the problems when your program and ours do not match)
2. Giving up multiple times with no reasonable explanation
3. Having no intention to complete the task (judging by the programs you generated and your explanation)

F Performance Analysis

In addition to the model's performance analysis by question type (defined using human judgment) in Section 7.2, we have conducted an additional analysis that measures the model's performance by function type, as shown in Table 16.

G GPT-3.5 and GPT-4

We evaluated the first two types of questions in our dataset in October 2023. The February 2024 version of the GPT models produced drastically different answers, resulting in significant performance drops. This change in the models' behavior has been studied previously (Chen et al., 2023), with the results indicating varying degrees of performance changes that can be substantial in some tasks, including mathematics and code generation.

H Prompt for Program Generation

The following is the part of the prompt we used for all LLMs tested:

```
Suppose we have a graph database stored in Neo4j. The
graph contains nodes of type Company, Concept, Fact,
Industry, State, and Person with the following
relations:
1. (Company)-[HAS_CONCEPT]->(Concept)
2. (Concept)-[HAS_FACT]->(Fact)
3. (Company)-[BELONG_TO_INDUSTRY_OF]->(Industry)
4. (Company)-[HAS_STATE_LOCATION]->(State)
5. (Person)-[IS_DIRECTOR_OF]->(Company)
6. (Person)-[]->(Company)

Company, Industry, State, and Person have one common
property: "name".
Concept refers to a financial concept and has one
property: “name”.
Fact refers to the value of that financial concept
reported in a particular year and has two properties:
“number” and “year”.
Companies may not report facts (values) every year.
The relations do not have any property.
The last relation, (Person)-[]->(Company), represents
a stakeholder relation.

Neo4j URI = “bolt://localhost:11017”
username = “...”
password = “...”

Consider the following conversational questions:
{{questions}}

Write a Python script using the “neo4j” package to
answer the last question.
Although the above questions are part of a
conversation, answer the last question only.
The script must print the exact answer to the screen
and nothing else.
Do not write anything else at all except the code.
Do not explain anything.
```

I AI Assistant

We utilized GPT-4 to assist with our research and writing. However, the content remains original, and we meticulously reviewed and revised the tool’s output prior to publication.