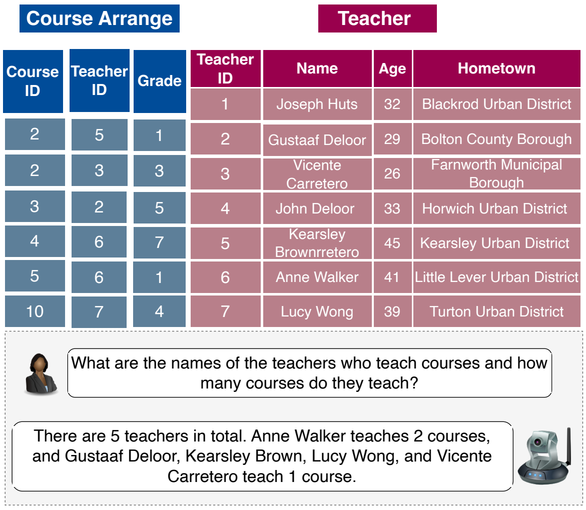
QFMTS: Generating Query-Focused Summaries over Multi-Table Inputs

Anonymous EACL submission

Abstract

Query-focused summarization (QFS) has been well-studied in the context of text-based data, However, QFS over semi-structured data such as tables remains under-explored. Existing studies primarily focus on single-table context, thus limiting the capability to handle complex multi-table scenarios. In this paper, we introduce a novel query-focused multi-table summarization task (QFMTS), where generation models should produce comprehensive query-focused summaries from multi-table contexts. This requires the models to perform arithmetic and multi-table operations such as *join* and *intersect*. To facilitate this task, we automatically collect the QFMTS dataset by leveraging large language models (LLMs) as data annotators. The dataset consists of $6,404$ query-summary pairs, each accompanied by multiple tables. Our quality evaluation, including automatic and human evaluation, illustrates the high quality of the dataset. To demonstrate the efficacy of the dataset, we experiment with state-of-the-art models, including open-source generation models and closed LLMs, on QFMTS. Experiment results and qualitative analysis reveal the significant challenges of the proposed task.



Course Arrange

Course ID	Teacher ID	Grade
2	5	1
2	3	3
3	2	5
4	6	7
5	6	1
10	7	4

Teacher

Teacher ID	Name	Age	Hometown
1	Joseph Huts	32	Blackrod Urban District
2	Gustaaf Deloor	29	Bolton County Borough
3	Vicente Carretero	26	Farnworth Municipal Borough
4	John Deloor	33	Horwich Urban District
5	Kearsley Brownrretero	45	Kearsley Urban District
6	Anne Walker	41	Little Lever Urban District
7	Lucy Wong	39	Turton Urban District

What are the names of the teachers who teach courses and how many courses do they teach?

There are 5 teachers in total. Anne Walker teaches 2 courses, and Gustaaf Deloor, Kearsley Brown, Lucy Wong, and Vicente Carretero teach 1 course.

Figure 1: A example of query-focused multi-table summarization. One should combine the information of the teachers from the two tables to answer the query.

1 Introduction

Query-focused summarization (QFS) aims to generate summaries from given contexts to answer a user question (Xu and Lapata, 2020, 2021; Vig et al., 2022; Zhang et al., 2023). This enables personalized response generation tailored to users' specific information needs. In spite of being extensively explored with text-based data, query-focused summarization over semi-structured data such as tables remains under-explored. The only existing research (Zhao et al., 2023a) introduces query-focused table summarization over a single input table. However, it does not handle multiple-table questions, thus limiting its capability in multi-table scenarios. Since these scenarios pose additional challenges such as multi-table operations *join*, *intersect*, *union*, etc.

To address these limitations, we propose query-focused multi-table summarization, a new task for summarizing information across multiple tables. For example, as shown in Figure 1, the user question "*What are the names of the teachers who teach courses and how many courses do they teach?*" involves identifying *names of teachers* to be associated with the column **Name** in the table `Teacher`, and *number of courses they teach* to be associated with the column **Course ID** in the table `Course Arrange`. Next, an association between the two table columns needs to be established using the common column **Teacher ID**, which is present in both tables to compute the final answer. In addition, models need to summarize all the answers by aggregating the number of teachers and providing the names and corresponding courses. This task not only involves the challenges of the multi-table question answering (QA) (Pal et al., 2023), such as measuring the faithfulness and correctness of the generated text, but also requires the generated text to be coherent and fluent.

To facilitate this new task, we release a new dataset, QFMTS (**Q**uery **F**ocused **M**ulti-**T**able **S**ummarization), comprising of $6,404$ query-summary pairs, each accompanied by multiple input tables. We design automatic data generation using large language models (LLMs) and conduct automatic and manual quality verification. Specifically, we adopt LLMs as data annotators using proper instructions and few-show demonstrations. Our quality evaluation, including automatic and human evaluation, demonstrates the high quality of the dataset with respect to *completeness*, *faithfulness*, and *fluency*. To investigate the efficacy of our dataset, we experiment with state-of-the-art models, including open-source models and closed LLMs. In particular, to instruct LLMs to perform the task, we first decompose the task into two sub-tasks, including multi-table QA and summarization. Then, we design two promoting strategies, namely single-stage and multi-stage prompting to address subtasks differently. Our experimental results show that open models fine-tuned on our dataset outperform closed LLMs, such as GPT-3.5 (Ouyang et al., 2022). Extensive qualitative analysis demonstrates that multi-table scenarios are much more challenging compared to single-table scenarios, indicating that there is large room for improvement.

To the best of our knowledge, we are the first to address the task of query-focused multi-table summarization, our main contributions are summarized as follows:

(1) We introduce the task of query-focused summarization over multiple tables, which requires models to generate summaries tailored to users' information needs. Our task generalizes to complex multi-table scenarios with operations, such as *join*, *intersect*, etc.

(2) We release a multi-table summarization dataset, QFMTS, comprising of $6,404$ question-summary pairs, each accompanied by multiple tables. We design an automatic data generation and evaluation process that facilitates large-scale data development.[1]

(3) We benchmark our task with state-of-the-art models, including closed-source and open-source LLMs, to demonstrate the efficacy of our dataset. Our extensive analysis demonstrates that handling multi-table scenarios is much more challenging compared to single-table scenarios. This suggests the need for further research efforts.

2 Related Work

Table-to-Text Generation Table-to-text generation involves generating an informative description or summary for a given table. Existing studies (Chen et al., 2020a; Suadaa et al., 2021; Liu et al., 2022a; Zhao et al., 2023b) primarily focus on summarizing the entire table. However, adapting these methods to multi-table scenarios that require combinations of tables is not straightforward. Additionally, users often possess specific information-seeking needs for partial information from tables. This motivates the need for query-focused table summarization. Query-focused single-table summarization was introduced by QTSUMM (Zhao et al., 2023a). While our work generalizes to a more realistic multi-table setting, introducing the challenges of multi-table reasoning.

Table Question Answering Table QA requires answering questions from table(s) (Pasupat and Liang, 2015; Yin et al., 2020; Liu et al., 2022c; Nan et al., 2022; Zhang et al., 2020a; Pal et al., 2022). Our work is inspired by `MultiTabQA` (Pal et al., 2023), which introduces generating tabular answers from multiple tables. We adapt the task to query-focused summarization that aligns with practical applications, such as conversational assistants and search engines (Ma et al., 2023).

3 QFMTS Task

3.1 Task Formulation

We formulate the QFMTS task as query-focused multi-table summarization, where the goal is to produce a fluent and informative summary from a question over multiple tables. Specifically, given a user question q and a set of input tables $\mathcal{T} = \{t_1, \ldots, t_k\}$, an effective query-focused multi-table summarizer reasons over $\mathcal{T}$ constrained by q and generates a paragraph-level summary s. The summary s should be factually correct and fluent.

3.2 Research Questions

In the context of the query-focused multi-table summarization task (QFMTS), we aim to answer the following research questions:

[1] We only release the validation set during the review: https://anonymfile.com/VByB/qfmts-valid.jsonl. The whole data and code will be released upon publication.

RQ1 How can we automatically generate the QFMTS dataset using large language models?

RQ2 How can we evaluate the quality of the QFMTS dataset?

RQ3 How do neural models, including open and closed source models, perform in the QFMTS task?

RQ4 To what extent do multi-table contexts bring challenges to neural models compared to single-table contexts?

4 QFMTS Dataset

We build our dataset on top of a subset of the multi-table QA dataset (Pal et al., 2023). The dataset comprises $6,715$ training and 985 validation samples, with each sample consisting of an SQL query, an associated natural language question, one or more input tables, and an answer table. The natural language question is a rewrite of the corresponding SQL query, and the answer table is a tabular answer to the question. The goal of the original task is to generate the answer table for either the natural language question or the SQL query based on the input tables. We repurpose this dataset to a query-focused multi-table summarization setting. To achieve this, we reframe the task's goal to generate an answer summary given the natural language question and input tables. We observe that more than 10% samples contain exceptionally large tables ($> 10,000$ tokens). This results in input truncation due to the models' maximum sequence length constraint, leading to sub-optimal generated results. To address this issue and ensure compatibility with different models, we exclude samples with a total number of input tokens (question and tables) exceeding $2,000$. This results in $5,721$ training and 683 validation samples, which we utilize to construct our dataset.

4.1 Summary Generation

Previous work in query-focused summarization over single tables, such as QTSumm (Zhao et al., 2023a), relies on human annotation to ensure the correctness of the summaries in table reasoning. However, manual annotation is time-consuming and costly. To address these limitations, we explore **RQ1** by designing an automatic annotation process to synthetically generate summaries for our dataset. Recent studies (Ding et al., 2023; Laskar et al., 2023b; He et al., 2023) have demonstrated that LLMs with proper instructions and demonstrations achieve competitive performance compared to crowdsourced workers. Thus, we employ LLMs as data annotators for summary generation. Specifically, we design a straightforward task of table-to-text transformation. However, instead of using input tables, we utilize the *ground-truth answer tables* for answer table-to-text transformation. This simplified task does not involve complex multi-table reasoning but ensures that the generated summaries are grounded in answer tables, thus ensuring the correctness and faithfulness of summaries. Finally, we instruct ChatGPT to perform this transformation via the public OpenAI API.[2] The overall annotation cost of API usage is approximately 30 dollars.

Input Formulation The input provided to ChatGPT is the ground-truth answer table. However, ChatGPT only accepts text-formatted inputs. Hence, we should first serialize the answer table within the prompt. There are several options, including markdown-style (Zhao et al., 2023a) and linearized flattening (Liu et al., 2022c; Pal et al., 2023). After manual inspection, we found that linearized flattening performs the best. For linearized flattening, a table is flattened to a sequence with sentinel words. For instance, a table named t with m rows and n columns is flattened as follows:

Prompt 4.1: Table formatting

<table_name> t **col:** h_1 | ... | h_n **row** 1**:** $c_{1,1}$ | ... | $c_{1,n}$ **row** i**:** $c_{i,1}$ | ... | $c_{i,n}$ ... **row** m**:** $c_{m,1}$ | ... | $r_{m,n}$

where h_j is j-th column header, and $c_{i,j}$ is the i-th row and j-th column cell.

Furthermore, we found that relying solely on the answer table is often insufficient to generate a self-contained summary, as it lacks the essential contextual information. However, we observe that Such missing contextual information can be extracted from the user question. For instance, the answer table lacks contextual information to demonstrate that semester `Summer 2010` has the *most registered students* in response to the question "*What is the semester in which most students registered? Show both the name and the id.*". Thus, we use the question as additional input to ensure the complete-

semester_name	semester_id
summer 2010	2

[2] gpt-3.5-turbo-0613

ness of the summary. Accordingly, an example summary is "*The semester with the most student registrations is the summer 2010 semester, with a semester ID of 2.*"

Instruction Design Our instruction follows the standard few-shot prompting technique (Brown et al., 2020). Specifically, we first write a comprehensive annotation guideline, including a description of the expected summary's discourse structure and length requirement. We observe that a more precise guideline leads to better generation quality. To provide further clarity to ChatGPT, we manually write summaries for a small number of examples as few-shot demonstrations (we use 5-shot in our experiments). The structure of the prompt for summary generation is shown in Prompt 4.2, and the complete prompt can be found in Appendix A.

> Prompt 4.2: Summary generation
>
> **Instruction:** A comprehensive guideline including input formats, expected summary's discourse structure, and length requirement.
>
> **Demonstrations:**
> Few-shot human-written demonstrations.
>
> *The input question and answer table.*

4.2 Quality Verification

We address **RQ2** by developing both automatic and manual quality evaluations to assess the quality of the generated summaries. We define three primary desiderata for quality verification as follows:

- **Faithfulness:** All statements in the summary should be factually consistent with the ground-truth answer table.
- **Completeness:** The summary should include all the information needs of the user, i.e., *all* facts in the ground-truth answer tables are present in the summary. Partial information from the answer table is deemed incomplete.
- **Fluency:** The summary is clear, articulate, and easy to understand by humans.

In our experiments, we employ standard sequence similarity metrics to measure completeness. Since there are no ideal metrics for faithfulness and fluency, we conduct a human evaluation to measure them. The results of the quality evaluation are shown in Table 1.

Automatic Evaluation We automatically evaluate the completeness of the generated summaries using the answer tables. However, as discussed in section 4.1, the answer table alone provides insufficient contextual information for completeness evaluation. Thus, we further evaluate the summary with respect to the question in addition to the answer to measure completeness.

Specifically, we adopt the lexical similarity score ROUGE (Lin, 2004), a widely used metric in table-to-text generation (Lin et al., 2023). As our primary focus lies in assessing the presence of information from the question and answer table within the summary, we report recall versions of ROUGE-1 and ROUGE-L, termed, ROUGE-1-R and ROUGE-L-R, respectively. We define ROUGE-1_{Q}-R and ROUGE-L_{Q}-R as the estimate of the lexical completeness of the summary regarding the question as the reference text, and ROUGE-1_{T}-R and ROUGE-L_{T}-R as the lexical completeness regarding the answer table. As shown in Table 1, ROUGE-1_{T}-R and ROUGE-L_{T}-R surpass 90, indicating that the generated summaries cover most facts, such as numerals and named entities from the answer tables. We also observe that ROUGE-1_{Q}-R and ROUGE-L_{Q}-R exhibit lower scores than table-based scores, yet exceeding 75. This indicates that the summaries retain the majority of keywords in the questions. The lower question-based scores may be because of stop-words and rewrite of the question to a declarative statement, where stop-words may be replaced or removed.

Human Evaluation To evaluate the faithfulness and fluency of the summaries, we randomly sampled 100 examples from the training and validation set, respectively. Three annotators who are well-versed in SQL and fluent in English were engaged to assess the faithfulness and fluency of the summaries with respect to the corresponding questions, input tables, and answer tables. We instructed the annotators to assign a binary label for faithfulness, commonly used in table-to-text generation (Chen et al., 2020b). They were to rate a summary as 1 if it is faithful to the associated answer table without containing hallucinations; otherwise, it is assigned 0. The annotators were also provided the question and input tables for interpretability. Following Zhao et al. (2023a), we estimate fluency with a 5-point Likert scale from 1 to 5, where a higher value indicates better fluency. The final human judgment for each example was computed by averaging the scores assigned by the three annotators. As shown in Table 1, the total of generated summaries are *faithful* with a high score of 0.99 and *flu-*

Split	Automatic Evaluation (%)				Human Evaluation†	
	ROUGE-1$_{Q}$-R	ROUGE-L$_{Q}$-R	ROUGE-1$_{T}$-R	ROUGE-L$_{T}$-R	Faithfulness	Fluency
Train	86.86	74.52	93.24	91.45	0.99	4.73
Valid	86.22	74.37	93.62	91.75	1.00	4.81
Total	86.79	75.50	93.28	91.48	0.99	4.77

Table 1: Evaluation results of the quality of QFMTS. We report recall scores for both ROUGE-based scores and BERTScore. †represents that we randomly sample 100 examples from the training and validation set, respectively.

ent with a score of 4.77, indicating more than 99% summaries are faithful to the associated answer tables. To measure the inter-annotator agreement, we adopt Fleiss Kappa (Fleiss, 1971). We obtained Kappa scores of 0.97 and 0.80 for faithfulness and fluency, respectively, indicating substantial agreement and almost perfect agreement, respectively.

Data Post-processing Based on the results of the quality evaluation, we manually revise samples with lower ROUGE scores. In particular, when their ROUGE-L$_{Q}$-R scores are below the threshold 0.7 or ROUGE-L$_{T}$-R scores are below the threshold 0.9. we revised them to ensure that they include sufficient information in the questions and answer tables.

4.3 Dataset Analysis

We analyze the generated dataset on the number of input tables and make comparisons with existing related datasets. Existing work on query-focused table summarization, QTSumm (Zhao et al., 2023a), is contemporary with ours but focuses on the simpler single-table setting.[3] As QTSumm is only the query-focused table summarization dataset, we compare our dataset, QFMTS, with QTSumm in Table 2. On average, our dataset contains 1.5 input tables per question whereas QTSumm is focused on only 1 input table. Our summaries are shorter, averaging 54.8 words compared to 67.7 of QTSumm, since our summaries only contain essential contextual information and the answers. Additionally, our QFMTS contains both single numeric operations such as `sum`, `average`, and multiple-table operations such as `join`, `intersect`, which are not present in QTSumm.

We further analyze the presence of multiple input tables in Table 3. Our training data contains both single and multiple input tables, with 58.8% of questions over a single table, 31.5% of samples over 2 tables, and 9.7% over more than 3 tables. The validation set displays a similar pattern with 56.6% samples over single tables, 37.2% samples over 2 tables, and 6% of samples over 3 tables.

Dataset	Statistics			Reasoning	
	#Q	#Table per Q	#Words in Summ	Num	Multi-Table
QTSumm	5,625	1.0	67.7	✓	✗
QFMTS	6,422	1.5	54.8	✓	✓

Table 2: Comparison of QFMTS with existing query-focused table summarization dataset. QFMTS is the only dataset that allows for both numeric and multi-table reasoning. Q, Summ, and Num indicate question, summary, and numeric, respectively.

Split	# Input tables		
	1	2	3+
Train	3,383 (58.8%)	1,810 (31.5%)	557 (9.7%)
Valid	392 (56.9%)	256 (37.2%)	41 (6.0%)
Total	4,387 (58.6%)	2,066 (32.1%)	598 (9.3%)

Table 3: Data statistics by the number of input tables. We have 5,721 training and 683 validation examples in total.

5 Methodology

We employ a variety of state-of-the-art models to demonstrate the efficacy of the QFMTS dataset, including open-source encoder-decoder models and closed-source LLMs. The open-source models are fine-tuned on our dataset while the LLMs are instructed with few-shot prompting (Brown et al., 2020).

5.1 Few-shot Prompting

We directly instruct LLMs, such as GPT-3.5 (Ouyang et al., 2022), to produce the expected summaries with few-shot demonstrations and without updating their parameters. To ensure the effectiveness of few-shot prompting, we decompose our task into two sub-tasks. The first sub-task involves *answering a question from tables*, where LLMs are

[3] At the time of writing, this dataset has not been released yet.

instructed to perform reasoning over multiple tables to obtain a list-like answer. The second sub-task requires *writing a summary of the answer* obtained in the first sub-task. We explore two prompting strategies to tackle both sub-tasks: *one-stage* and *multi-stage* prompting. One-stage prompting requires the LLM to perform the two sub-tasks in a single prompt while multi-stage prompting requires the LLM to perform sequential sub-tasks with independent prompts for each sub-task.

One-stage Prompting In one-stage prompting, we instruct the LLMs to complete two sub-tasks step by step within a single prompt. The prompt comprises two parts. First, we prompt the LLM to answer the question from the tables. Note that the answer is a list-like answer rather than a table. We follow the chain-of-thoughts (CoT) style prompts (Wei et al., 2022; Kojima et al., 2022) by adding "Let's think step by step" to enhance the reasoning abilities of the LLM. Second, we instruct the LLM to write a summary based on the question and previously generated answers. We ensure that the task description is similar to the instruction used in the annotation process, described in subsection 4.1, so the generated summarises share the same structure as reference summaries. The outline of the prompt is shown in Prompt 5.1, and the complete prompt is shown in Appendix A.

Prompt 5.1: One-stage prompting

Instruction: A introduction of input formats and objectives to complete two tasks below step by step.

Task 1: Answering Question from Tables
A task description with CoT for obtaining a list-like answer.

Task 2: Write Summary for Answers
A task description for generating a summary.

Demonstrations:
Few-shot human-written demonstrations.

The input question and tables.

Multi-stage Prompting In multi-stage prompting, the LLM completes two independent sub-tasks in a sequential manner. In the first stage, we instruct the LLM to only answer the given question, without instruction to generate a summary or specific constraints on the answer format. This results in the LLM generating a list-like response. In the second stage, the questions and the generated answers from the first stage are used to prompt the LLM to generate a summary. The outline of prompts is shown in Prompt 5.2 and 5.3. The complete prompts are shown in Appendix A.

Prompt 5.2: Stage 1: Multi-table QA

Instruction: A introduction of input formats and objectives.

Task: Answering Question from Tables
A task description for obtaining a list-like answer.

Demonstrations:
Few-shot human-written demonstrations.

The input question and tables.

Prompt 5.3: Stage 2: Summarization

Instruction: A introduction of objectives.

Task: Write Summary for Answers
A task description for generating a summary.

Demonstrations:
Few-shot human-written demonstrations.

The input question and generated answers.

Discussion The one-stage method is straightforward yet more challenging since the LLM is required to perform both table reasoning and summary generation in an end-to-end manner. In contrast, the multi-stage method provides more flexibility by enabling the LLM to focus on one sub-task at a time. Furthermore, the comparison between the single and the multi-stage method can be used to analyze the relative difficulty between multi-table summarization and multi-table QA.

5.2 Fine-tuning

In addition to prompting the LLM to produce summaries directly, we also fine-tune open-source encoder-decoder models on QFMTS. Specifically, the input to the models is the concatenated sequence of the question and all linearized input tables. The table names are appended to the respective tables to disambiguate among different input table content. The final input sequence for a sample with k tables is represented as $question\ [table_1] \ldots [table_k]$ where $[table_i]$ is the linearized representation of input table i as shown in prompt 4.1. The output of the model is the query-focused summary.

6 Experimental Setup

We utilize state-of-the-art open-source and closed-source models to benchmark QFMTS. For open-source models, we use pre-trained models `bart-base` (139M parameters), `bart-large` (406M parameters) (Lewis et al., 2020), and state-of-the-art multi-table QA model `MultiTabQA` (Pal et al., 2023) to evaluate the efficacy of our dataset. `MultiTabQA` generates answer tables to multi-table questions, which was trained on a multi-table QA task using a pre-trained `bart-base` model. We name the fine-tuned models `BART-Base-TS`, `BART-Large-TS`, and `MultiTabQA-TS`, with `TS` indicating that they are fine-tuned on our **T**able **S**umarization dataset. For closed LLMs, we explore `GPT-3.5` (Ouyang et al., 2022) and `GPT-4` (OpenAI, 2023) as backbones [4] in our experiments. Since the GPT family has demonstrated significant unsupervised performance in many downstream NLP tasks (Laskar et al., 2023a). We design zero- and few-shot settings to instruct `GPT-3.5` to produce summaries. However, due to budget constraints, we only include a few-shot setting for `GPT-4`. Details of the prompts can be found in Appendix A.

We fine-tune the open-source models on the QFMTS training set using the AdamW optimizer (Loshchilov and Hutter, 2019) for 64 epochs with a learning rate of $1e^{-4}$, batch size of 256, and the maximum sequence length of 1024. We randomly split 10% of the training set as a development set and choose the best-performing model based on the loss of development. All experiments are conducted on a single A6000 GPU. For `GPT-3.5` and `GPT-4`, we set the temperature and top-p to 0.1 and 0.95, respectively. We set max tokens of outputs to 700 and 400 for one-stage and multi-stage methods, respectively. We included both 0-shot and 3-shot settings.

Evaluation Metrics We evaluate the predicted summary with respect to the reference summary by estimating the similarity between them in different aspects, such as fluency and correctness. Following Zhao et al. (2023a), We adopt two lexical-based metrics, SacreBLEU (Papineni et al., 2002) and ROUGE-L (Lin and Hovy, 2003), and a semantic similarity metric, BERTScore (Zhang et al., 2020b). We report the F1 version for ROUGE-L (longest common subsequences) and BERTScore. We use `deberta-xlarge-mnli` (He et al., 2021) as the backbone for BERTScore.

Model	**SB**	**RL**	**BSc**
One-stage prompting			
`GPT-3.5` (0-shot)	32.99	58.72	59.07
`GPT-3.5` (3-shot)	37.94	62.23	64.94
Multi-stage prompting			
`GPT-3.5` (0-shot)	38.94	63.05	65.59
`GPT-3.5` (3-shot)	42.55	66.53	68.30
`GPT-4` (3-shot)	45.13	69.02	72.11
Fine-tuned on QFMTS			
`BART-Base-TS`	44.03	66.70	67.03
`BART-Large-TS`	47.33	68.98	70.10
`MultiTabQA-TS`	**50.67**	**72.38**	**72.59**

Table 4: Performance comparison of baseline models on the QFMTS validation set. They are either prompted with few-shot demonstrations or fine-tuned on the QFMTS training set. SB, RL, and BSc denote SacreBLEU, ROUGE-L, and BERTScore, respectively.

7 Results and Analysis

We explore **RQ3** by comparing the model performance on our dataset. We show the results in Table 4. We find that the fine-tuned models achieve better results compared to all instruction-tuned `GPT-3.5` variants. However, `GPT-4` exhibits competitive performance as `BART-Large-TS`. Among all models, `MultiTabQA-TS` achieves the highest performance. Note that `MultiTabQA-TS` has been fine-tuned on our dataset using the `bart-base` structure. Even though the `BART-Large-TS` is larger, `MultiTabQA-TS` exhibits better multi-table reasoning and summarization performance.

We also observe that multi-stage prompted `GPT-3.5` outperforms one-stage one by a large margin. Note that multi-stage prompting breaks down the task into two independent sub-tasks: (i) multi-table QA, and (ii) summarization. The results indicate that end-to-end query-focused multi-table summarization is much more challenging than multi-table QA. As the LLM focuses on only the multi-table reasoning sub-task in the first stage, it generates a correct answer more frequently. The follow-up sub-task of summarization is simpler and leads to better summaries compared to one-stage prompting.

To answer **RQ4**, we show the performance comparison between samples with single-table inputs

[4] `gpt-3.5-turbo-0613` and `gpt-4-0613`

Model	#Input Tables			
	1 (Single-table)		2+ (Multi-table)	
	R-L	**BSc**	**R-L**	**BSc**
GPT-3.5	68.57	70.82	63.63	64.95
GPT-4	69.48	73.05	68.30	**70.85**
MultiTabQA-TS	**74.64**	**74.95**	**69.30**	69.47

Table 5: Results of GPT-3.5 and GPT-4 with 3-shot multi-stage prompting, and MultiTabQA-TS regarding the number of input tables. R-L and BSc denote, ROUGE-L and BERTScore, respectively.

and multi-table inputs in Table 5. We observe that multiple input tables lead to a drop in all scores for all models. The drop is most significant for the smaller sized MultiTabQA-TS, and least for the largest sized GPT-4. Although reasoning across multiple tables is more challenging than single tables, model capacity diminishes this gap. However, instruction-tuned LLMs do not necessitate better table reasoning than the best-performing models fine-tuned on our dataset.

Qualitative Analysis To provide deeper insights into the efficacy and challenges of our task, we conduct a manual analysis of the summaries generated by MultiTabQA-TS on the QFMTS validation set, including success and failure cases. We observe that MultiTabQA-TS successfully performs arithmetic and multi-table operations in some cases. A success case illustrates this. For the question “*Which employee received the most awards in evaluations? Give me the employee name.*” over 2 input tables:

Employee

ID	Name	Age
1	**George Chuter**	23
2	Lee Mears	29
...	...	...

Evaluation

ID	Year _awarded	Bonus
1	2011	**3000**
2	2015	3200
1	2016	**2900**
...	...	...

With the reference summary “*The employee who received the most awards in evaluations is **George Chuter**.*”, MultiTabQA-TS reasons over the 2 tables, perform the complex table operations, such as count and join. Particularly, the model finds two records of awards of George Chuter in the table Evaluation and aggregates the total number of awards. After joining the two tables, the model accurately identifies George Chuter as the person with the most awards, generating “*The employee who received the most awards in evaluations is **George Chuter**.*”.

A failure case of MultiTabQA-TS also illustrates the challenges of multiple-table scenarios. For the question “*What are the names of all European countries with at least 3 manufacturers?*” over 3 input tables:

Continents

Cont Id	Continent
1	America
2	**Europe**
3	Asia
4	Africa
5	Australia

Countries

Country Id	Country Name	Cont-inent
2	**Germany**	**2**
3	**France**	**2**
1	USA	1
8	**Korea**	**3**
...	...	...

Car Makers

Id	Maker	Full Name	Country
2	Volkswagen	Volkswagen	**2**
3	bmw	BMW	**2**
...	...	...	...
7	citroen	Citroen	**3**
...	...	...	...
14	opel	Opel	**2**
15	peugeaut	Peugeaut	**3**
16	renault	Renault	**3**
...	...	...	...
22	kia	Kia Motors	**8**

With the reference summary “*There are 2 European countries with at least 3 manufacturers. The names of these countries are France and Germany.*”, the model mistakenly generates "*There are 2 European countries with at least 3 manufacturers. The names of these countries are France and Korea.*". Even though this generated summary exhibits a high degree of *fluency*, it is only partially *faithful* and *complete* due to the incorrect inclusion of *Korea*, a country not located in Europe.

8 Conclusion

We present QFMTS, query-focused multi-table summarization that enables models to perform complex arithmetic and multi-table reasoning. We create the QFMTS dataset, comprising of 6,404 query-summary pairs, each accompanied by multiple input tables. We utilize LLMs for dataset generation by designing a simple task of transforming answer tables to summaries, which leads to high-quality summaries. We benchmark our dataset with both open-source models and closed-source LLMs. Experimental results show that smaller open-source models fine-tuned on QFMTS outperform LLMs by a large margin. We also highlight the greater complexity of multi-table scenarios compared to single-table scenarios. This suggests that there is large room for improvement in complex multi-table reasoning, and more research efforts are needed.

9 Limitations

The summaries on our QFMTS were automatically generated by GPT-3.5, despite being scalable and cost-effective, which may limit the diversity of the summaries regarding vocabulary or sentence structure compared to expert annotators. For few-shot prompting baselines, we used fixed few-shot demonstrations, which are easy to implement yet sub-optimal. Advanced demonstration selection methods, such as retrieval-augmented methods (Liu et al., 2022b; Rubin et al., 2022), have the potential to enhance generation capabilities. Furthermore, these baselines do not explore re-verifying the correctness of the answers before summary generation. Such a verification mechanism may boost the faithfulness of the summaries and can be explored in the future.

10 Ethical Considerations

The source questions and tables in QFMTS are derived from a multi-table QA dataset (Pal et al., 2023), which is openly access under the MIT license. It facilitates its usage for research purposes. The baseline models used in this paper include closed LLMs accessible via the commercial OpenAI API [5] and publicly available open-source models. In particular, we leverage Copilot primarily to assist with data processing code. We use ChatGPT to mainly correct grammatical errors and ensure the paper does not contain any of the generated text directly from ChatGPT.

References

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, Virtual*.

Wenhu Chen, Jianshu Chen, Yu Su, Zhiyu Chen, and William Yang Wang. 2020a. Logical natural language generation from open-domain tables. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7929–7942, Online. Association for Computational Linguistics.

Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. 2020b. TabFact: A large-scale dataset for table-based fact verification. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Bosheng Ding, Chengwei Qin, Linlin Liu, Yew Ken Chia, Boyang Li, Shafiq Joty, and Lidong Bing. 2023. Is GPT-3 a good data annotator? In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11173–11195, Toronto, Canada. Association for Computational Linguistics.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: Decoding-Enhanced Bert with Disentangled Attention. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Xingwei He, Zhenghao Lin, Yeyun Gong, A-Long Jin, Hang Zhang, Chen Lin, Jian Jiao, Siu Ming Yiu, Nan Duan, and Weizhu Chen. 2023. AnnoLLM: Making large language models to be better crowdsourced annotators. *CoRR*, abs/2303.16854.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *NeurIPS*.

Md Tahmid Rahman Laskar, M Saiful Bari, Mizanur Rahman, Md Amran Hossen Bhuiyan, Shafiq Joty, and Jimmy Huang. 2023a. A systematic study and comprehensive evaluation of ChatGPT on benchmark datasets. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 431–469, Toronto, Canada. Association for Computational Linguistics.

Md. Tahmid Rahman Laskar, Mizanur Rahman, Israt Jahan, Enamul Hoque, and Jimmy X. Huang. 2023b. CQSumDP: A ChatGPT-Annotated resource for query-focused abstractive summarization based on debatepedia. *CoRR*, abs/2305.06147.

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.

[5] https://openai.com/blog/openai-api

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Chin-Yew Lin and Eduard Hovy. 2003. Automatic evaluation of summaries using N-gram co-occurrence statistics. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 150–157.

Yupian Lin, Tong Ruan, Jingping Liu, and Haofen Wang. 2023. A survey on neural data-to-text generation. *IEEE Transactions on Knowledge and Data Engineering*.

Ao Liu, Haoyu Dong, Naoaki Okazaki, Shi Han, and Dongmei Zhang. 2022a. PLOG: Table-to-logic pretraining for logical table-to-text generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5531–5546, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2022b. What makes good in-context examples for GPT-3? In *Proceedings of Deep Learning inside out (DeeLIO 2022): The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 100–114, Dublin, Ireland and Online. Association for Computational Linguistics.

Qian Liu, Bei Chen, Jiaqi Guo, Morteza Ziyadi, Zeqi Lin, Weizhu Chen, and Jian-Guang Lou. 2022c. TAPEX: Table pre-training via learning a neural SQL executor. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Congbo Ma, Wei Emma Zhang, Mingyu Guo, Hu Wang, and Quan Z Sheng. 2023. Multi-document Summarization via Deep Learning Techniques: A Survey. *ACM Comput. Surv.*, 55(5):102:1–102:37.

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.

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

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *NeurIPS*.

Vaishali Pal, Evangelos Kanoulas, and Maarten Rijke. 2022. Parameter-efficient abstractive question answering over tables or text. In *Proceedings of the Second DialDoc Workshop on Document-Grounded Dialogue and Conversational Question Answering*, pages 41–53, Dublin, Ireland. Association for Computational Linguistics.

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.

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, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

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

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. 2022. Learning to retrieve prompts for in-context learning. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2655–2671, Seattle, United States. Association for Computational Linguistics.

Lya Hulliyyatus Suadaa, Hidetaka Kamigaito, Kotaro Funakoshi, Manabu Okumura, and Hiroya Takamura. 2021. Towards table-to-text generation with numerical reasoning. 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 1451–1465, Online. Association for Computational Linguistics.

Jesse Vig, Alexander Fabbri, Wojciech Kryscinski, Chien-Sheng Wu, and Wenhao Liu. 2022. Exploring neural models for query-focused summarization. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1455–1468, Seattle, United States. Association for Computational Linguistics.

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

Yumo Xu and Mirella Lapata. 2020. Coarse-to-fine query focused multi-document summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3632–3645, Online. Association for Computational Linguistics.

Yumo Xu and Mirella Lapata. 2021. Generating query focused summaries from query-free resources. 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 6096–6109, Online. Association for Computational Linguistics.

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, Online. Association for Computational Linguistics.

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 2020, Virtual Event, China, July 25-30, 2020*, pages 1537–1540. ACM.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2020b. BERTScore: Evaluating Text Generation with BERT. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Weijia Zhang, Svitlana Vakulenko, Thilina Rajapakse, Yumo Xu, and Evangelos Kanoulas. 2023. Tackling Query-Focused Summarization as A Knowledge-Intensive Task: A Pilot Study. In *Gen-IR@SIGIR 2023: The First Workshop on Generative Information Retrieval*.

Yilun Zhao, Zhenting Qi, Linyong Nan, Boyu Mi, Yixin Liu, Weijin Zou, Simeng Han, Xiangru Tang, Yumo Xu, Arman Cohan, and Dragomir Radev. 2023a. QT-Summ: A new benchmark for query-focused table summarization. *CoRR*, abs/2305.14303.

Yilun Zhao, Haowei Zhang, Shengyun Si, Linyong Nan, Xiangru Tang, and Arman Cohan. 2023b. Large language models are effective table-to-text generators, evaluators, and feedback providers. *CoRR*, abs/2305.14987.

A Prompts

Complete prompts used in the paper are shown below.

Appendix A.1: Complete prompt for summary generation

Instruction: You will be provided with a question and its tabular answer. Your task is to write a concise, fluent, and accurate summary for the given table. The table consists of m rows and n columns, following this format:
col: <column header 1> | <column header 2> | ... | <column header n> row 1: <value 1,1> | <value 1,2> | ... | <value 1,n> row 2: <value 2,1> | <value 2,2> | ... | <value 2,n> ... row m: <value m,1> | <value m,2> | ... | <value m,n>.
The summary should comprise two sections: 1) The initial segment first mentions the total number of data if there are 2 or more rows in the table. Then, it should rephrase the question as a declarative statement while retaining all relevant keywords. 2) The subsequent segment must include all the information, including numerical data and entities, from the table. Describe the table row by row without explaining the column headers. The summary should be a conventional text paragraph without any list, containing a minimum of 5 words while not exceeding 300 words in length.
You can refer to the demonstrations below. Each demonstration consists of a question, its tabular answer, and a human-written summary.

Demonstrations:
Question: What is the total number of singers?
Table: col: count(*) row 1 : 6
Summary: The total number of singers is 6.

Question: What is the abbreviation of the airline that has the fewest flights and what country is it in?
Table: col : Abbreviation | Country row 1 : AirTran | USA
Summary: The abbreviation of the airline that has the fewest flights is AirTran, and its country of location is the USA.

Question: List the maximum weight and type for each type of pet.
Table: col: max(weight) | PetType row 1 : 12.0 | cat row 2 : 13.4 | dog
Summary: There are 2 types of pets, which are the cat and the dog. The maximum weight of the cat is 12.0 and the maximum weight of the dog is 13.4.

Question: What are the different first names and ages of the students who do have pets?
Table: col : Fname | Age row 1 : Linda | 18 row 2 : Tracy | 19
Summary: There are 2 different students who do have pets. The first names and ages of the students are Linda with 18 years old and Tracy with 19 years old.
. . . . . .

Now follow the instructions and the demonstrated style above to write a concise, fluent, and accurate summary for the question and its tabular answer provided below:

Question: *input question here*
Table: *answer table here*

Appendix A.2: Complete single-stage prompt

Instruction: You will be given a question along with one or more tables to complete two tasks step by step. Each table contains a name and content with multiple rows and columns, formatted as follows:
col: <column header 1> | <column header 2> | ... | <column header n> row 1: <value 1,1> | <value 1,2> | ... | <value 1,n> row 2: <value 2,1> | <value 2,2> | ... | <value 2,n> ... row m: <value m,1> | <value m,2> | ... | <value m,n>.

Task 1: Answering the Question from the Tables.
Your first task is to answer the question using only the information from the tables, such as numerical data and entities. This may involve performing arithmetic calculations and combining data from multiple tables if necessary. Please begin your response with "Answers:" and enumerate all discovered answers one by one, separating them with commas ",". Let's think step by step.

Task 2: Writing a Summary for the Answers.
Your second task is to write a concise, fluent, and accurate summary based on the answers generated in the first task. This summary should begin with the word "Summary:" and follow the guidelines as follows: 1) Introduction: Begin by using a numeral to indicate the total number of answers if there are two or more; Then, rephrase the question as a declarative statement while retaining all relevant keywords. 2) Body: Present all discovered answers one by one. The summary should be a standard paragraph format without using lists, containing a minimum of 5 words but not exceeding 300 words in length. You can refer to the demonstrations below. Each demonstration consists of a question, tables, and human-written answers, and a summary.

Demonstrations:
Question: Show the name for regions not affected.
Table 1: Name: region; Content: col : Region_id | Region_code | Region_name row 1 : 1 | AF | Afghanistan row 2 : 2 | AL | Albania row 3 : 3 | DZ | Algeria row 4 : 4 | DS | American Samoa row 5 : 5 | AD | Andorra row 6 : 6 | AO | Angola row 7 : 7 | AI | Anguilla row 8 : 8 | AQ | Antarctica row 9 : 9 | AG | Antigua and Barbuda row 10 : 10 | CY | Cyprus row 11 : 11 | CZ | Czech Republic row 12 : 12 | DK | Denmark row 13 : 13 | DJ | Djibouti
Table 2: Name: Affected Region; Content: col : Region_id | Storm_ID | Number_city_affected row 1 : 1 | 1 | 10 row 2 : 2 | 1 | 15 row 3 : 3 | 3 | 30 row 4 : 1 | 4 | 22 row 5 : 12 | 5 | 37 row 6 : 2 | 5 | 12
Answers: American Samoa, Andorra, Angola, Anguilla, Antarctica, Antigua and Barbuda, Cyprus, Czech Republic, and Djibouti are the names for regions not affected.
Summary: There are 9 regions that are not affected. These regions include American Samoa, Andorra, Angola, Anguilla, Antarctica, Antigua and Barbuda, Cyprus, Czech Republic, and Djibouti.
. . . . . .

Now follow the instructions and the demonstrated style above to complete the two tasks step by step for the question and tables provided below:

Question: *input question here*
Tables: *input tables here*

Appendix A.3: Complete prompt for stage 1

Instruction: You will be given a question along with one or more tables to complete the task below. Each table contains a name and content with multiple rows and columns, formatted as follows:
col: <column header 1> | <column header 2> | ... | <column header n> row 1: <value 1,1> | <value 1,2> | ... | <value 1,n> row 2: <value 2,1> | <value 2,2> | ... | <value 2,n> ... row m: <value m,1> | <value m,2> | ... | <value m,n>.

Task: Answering the Question from the Tables.
Your task is to answer the question using only the information from the tables, such as numerical data and entities. This may involve performing arithmetic calculations and combining data from multiple tables if necessary. Please begin your response with ’Answers:’ and enumerate all discovered answers one by one, separating them with commas ’,’. Let’s think step by step.

Demonstrations:
Question: Show the name for regions not affected.
Table 1: Name: region; Content: col : Region_id | Region_code | Region_name row 1 : 1 | AF | Afghanistan row 2 : 2 | AL | Albania row 3 : 3 | DZ | Algeria row 4 : 4 | DS | American Samoa row 5 : 5 | AD | Andorra row 6 : 6 | AO | Angola row 7 : 7 | AI | Anguilla row 8 : 8 | AQ | Antarctica row 9 : 9 | AG | Antigua and Barbuda row 10 : 10 | CY | Cyprus row 11 : 11 | CZ | Czech Republic row 12 : 12 | DK | Denmark row 13 : 13 | DJ | Djibouti
Table 2: Name: Affected Region; Content: col : Region_id | Storm_ID | Number_city_affected row 1 : 1 | 1 | 10 row 2 : 2 | 1 | 15 row 3 : 3 | 3 | 30 row 4 : 1 | 4 | 22 row 5 : 12 | 5 | 37 row 6 : 2 | 5 | 12
Answers: American Samoa, Andorra, Angola, Anguilla, Antarctica, Antigua and Barbuda, Cyprus, Czech Republic, and Djibouti are the names for regions not affected.
. . . . . .

Now follow the instructions and the demonstrated style above to complete the task step by step for the question and tables provided below:

Question: *input question here*
Tables: *input tables here*

Appendix A.4: Complete prompt for stage 2

Instruction: You will be given a question along with its one or more answers to complete the task below.
Task: Writing a Summary for the Answers.
Your task is to write a concise, fluent, and accurate summary based on the answers generated in the first task. This summary should begin with the word "Summary:" and follow the guidelines as follows: 1) Introduction: Begin by using a numeral to indicate the total number of answers if there are two or more; Then, rephrase the question as a declarative statement while retaining all relevant keywords. 2) Body: Present all discovered answers one by one. The summary should be a standard paragraph format without using lists, containing a minimum of 5 words but not exceeding 300 words in length.
You can refer to the demonstrations below. Each demonstration consists of a question, tables, and human-written answers, and a summary.

Demonstrations:
Question: Show the name for regions not affected.
Answers: American Samoa, Andorra, Angola, Anguilla, Antarctica, Antigua and Barbuda, Cyprus, Czech Republic, and Djibouti are the names for regions not affected.
Summary: There are 9 regions that are not affected. These regions include American Samoa, Andorra, Angola, Anguilla, Antarctica, Antigua and Barbuda, Cyprus, Czech Republic, and Djibouti.
. . . . . .

Now follow the instructions and the demonstrated style above to complete the task step by step for the question and answers provided below:

Question: *input question here*
Answers: *generated answers here*