SEC-FINTABLES: Evaluating Large Language Models for Detecting Logical Inconsistencies on Tabular Data

Shuyan Ke, Qiong Wu, Hui Li*, Liujuan Cao
Key Laboratory of Multimedia Trusted Perception and Efficient Computing
Ministry of Education of China, Xiamen University
{keshuyan, qiong}@stu.xmu.edu.cn, {hui,caoliujuan}@xmu.edu.cn

Abstract

Large language models (LLMs) are increasingly deployed in high-stakes domains reliant on tabular data (e.g., financial reporting), where undetected logical inconsistencies such as mismatched totals and components can lead to critical errors. Yet, the ability of LLMs to identify such inconsistencies remains poorly understood, hindered by the absence of standardized evaluation frameworks and cell-level annotated datasets. To bridge this gap, we propose a comprehensive benchmark SEC-FINTABLES comprising 103,395 real-world and error-injected table instances, alongside a novel evaluation protocol that decomposes inconsistency detection into granular sub-tasks. Through evaluating both proprietary and open-source LLMs on SEC-FINTABLES, we find that contemporary LLMs exhibit only partial competence in detecting logical inconsistencies. Our study reveals key limitations and improvement opportunities for LLMs. We believe SEC-FINTABLES and our evaluation protocol can serve as a practical resource for advancing reliable inconsistency detection of LLMs in tabular reasoning. We release SEC-FINTABLES at https://github.com/XIEFOX/SEC-Fintables.

1 Introduction

In recent years, large language models (LLMs) have been increasingly applied to high-stakes domains that rely heavily on tabular data, such as financial statements (Kim et al., 2025a; Yang et al., 2024; Kim et al., 2025b), medical laboratory reports (Zhang et al., 2025; Adam et al., 2024; Bagci et al., 2025), and government statistics (Aggarwal et al., 2025; Suleymanli et al., 2025). Reliable use in these domains requires LLMs to recognize logical constraints within structured data. For example, values must satisfy constraints such as additive and ratio relations, range restrictions, and cross-dimensional consistency. Violating such constraints directly undermines the credibility and reliability of outputs in high-stakes decision-making.

Logical inconsistencies can be defined as data values that violate their logical constraints. In finance, detecting them enables automated auditing of inter-statement linkages, curbing systemic risks at their source. In healthcare, it reduces the risk of misdiagnosis based on test results. In government statistics, it safeguards the credibility of published outputs. Detecting such inconsistencies requires LLMs to understand multi-level table headers, interpret their semantics accurately, and perform implicit unit normalization.

Nevertheless, a systematic framework for evaluating LLMs' ability to detect logical inconsistencies in structured tabular data remains lacking. This limitation arises from two major factors:

1. First, structural and logical reasoning is inherently challenging for LLMs. To illustrate, we examine GPT-5 (OpenAI, 2025) on a financial statement (Fig. 1). GPT-5 failed to identify the U.S. entry due to incomplete verification of contributing components and the misinterpretation of normal column-wise variations as inconsistencies, highlighting the difficulty of reasoning over logical constraints in structured values.

2. Second, there is a scarcity of datasets with strong logical constraints and fine-grained annotations. Such resources are essential for benchmarking LLMs' reasoning over tables. Existing table-based QA benchmarks such as SciDQA (Singh et al., 2024) and HybridQA (Chen et al., 2020b) primarily focus on semantic understanding and isolated question answering, but do not evaluate whether models can detect logical inconsistencies

To bridge the gap, we construct a benchmark SEC-FINTABLES, the first benchmark specifically designed for logical inconsistency detection in real-

*Corresponding Author.

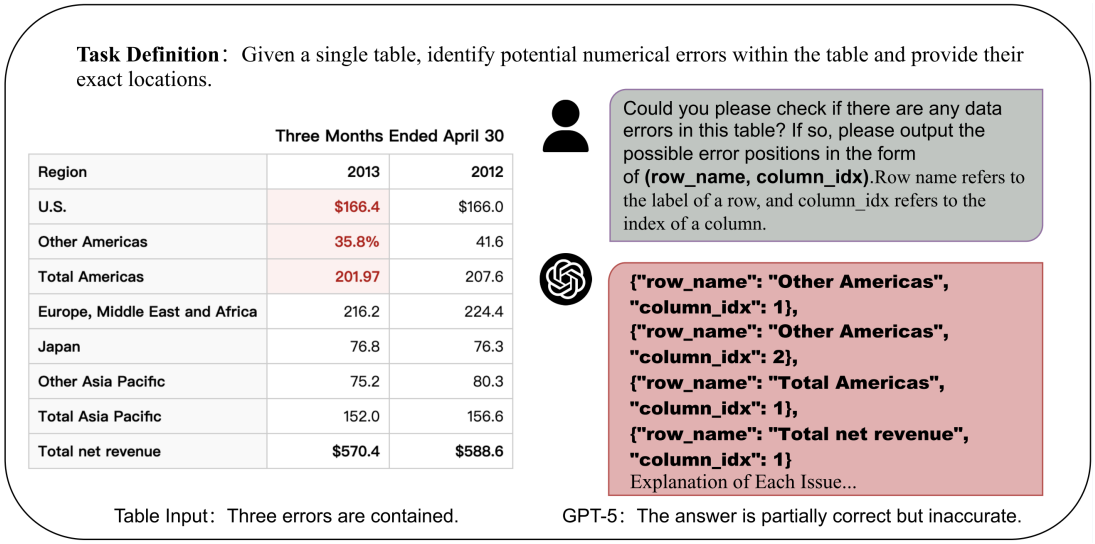

Figure 1: An example showing that GPT-5 struggles with logical inconsistency detection.

world financial tables, and define an evaluation protocol for detecting logical inconsistencies in structured tables. Financial reports are chosen because they exhibit clear structure, explicit semantics, and tightly coupled internal relations. SEC-FINTABLES is built from U.S. Securities and Exchange Commission (SEC)[1] quarterly filings, providing a large-scale, manually verified dataset with well-defined logical constraints. Building upon this benchmark, we decompose logical inconsistency detection into four sub-tasks: (1) Error Count Prediction, (2) Missing Count Estimation, (3) Error Categorization, and (4) Error Positioning. Considering that state-of-the-art LLMs still struggle to achieve reliable performance even in single-table scenarios, as observed in our evaluation, we focus on the single-table setting, which establishes the foundational step toward more complex multi-table reasoning. The more complex, cross-table setting is left as future work.

Using SEC-FINTABLES, we systematically evaluate a range of LLMs, including the proprietary model GPT-5 (OpenAI, 2025), open-source models such as LLaMA 3.1 (Meta AI, 2024), Qwen series (Qwen2.5 (Qwen Team, 2024a), Qwen3-235B (Qwen Team, 2025) and QWQ-32B (Qwen Team, 2024b)) and DeepSeek-R1 (DeepSeek Team, 2024). Results reveal that these LLMs exhibit only partial competence in detecting logical inconsistencies. For example, DeepSeek-R1, reaches a limited IoU of 0.325 in error localization, suggesting the difficulty of precise logical localization in tabular reasoning. In summary, our contributions are:

- We construct SEC-FINTABLES, a manually curated benchmark containing over 100k instances derived from U.S. SEC filings, enriched with controlled errors and verified for quality.
- We design an evaluation protocol to assess the ability of mainstream LLMs to detect logical inconsistencies in structured tabular data, formulated as four sub-tasks: predicting error counts, estimating missing counts, categorizing error types, and localizing erroneous cells.
- We perform a systematic evaluation of several mainstream LLMs on SEC-FINTABLES, providing insights into future research on enhancing LLMs' ability to detect logical inconsistencies.

2 Related Work

2.1 Early Work on Table Understanding

Early research on table understanding primarily focused on fundamental tasks such as table structure recovery (Xue et al., 2021; Kim et al., 2016; Wagner et al., 2015), cell classification (Tong et al., 2022; Ghasemi-Gol et al., 2019), and entity alignment (Yin et al., 2020; Fetahu et al., 2019). These approaches typically relied on rule-based systems or shallow machine learning techniques, aiming to restore explicit structures rather than supporting complex reasoning. Several representative datasets are proposed for structural recovery and basic semantic tasks, including SciTSR (Qiao et al., 2021) for structure recognition, TabFact (Chen et al., 2020a) for fact verification, FinTabNet (Zheng et al., 2021) for financial table parsing, and TURL (Papotti, 2022) for relational

[1] https://www.sec.gov/edgar.shtml

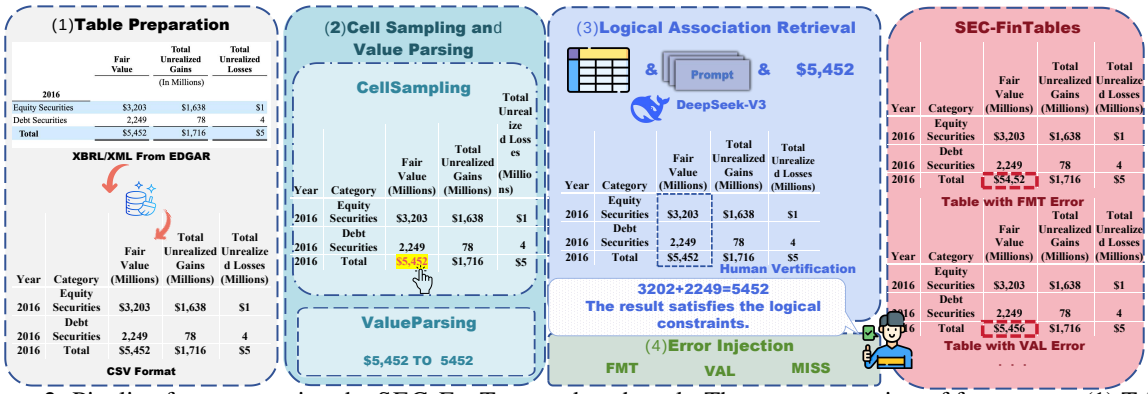

Figure 2: Pipeline for constructing the SEC-FINTABLES benchmark. The process consists of four stages: (1) Table Preparation, (2) Cell Sampling and Value Parsing, (3) Logical Association Retrieval, and (4) Error Injection

table pretraining.

2.2 LLM Benchmarks for Table Reasoning

With the rise of LLMs, the focus of research on tabular data has shifted from local structure recovery to semantic reasoning over tables. Datasets such as InfoTabS (Gupta et al., 2020), OTT-QA (Chen et al., 2021), FeTaQA (Nan et al., 2022), TAT-QA (Zhu et al., 2021), and HiTab (Cheng et al., 2022) are proposed to evaluate contradiction detection, open-domain QA, free-form answer generation, numerical reasoning, and hierarchical aggregation. These benchmarks advanced evaluation from structural analysis to higher-level reasoning, but they remain centered on QA and text generation tasks.

Distinction of Our Work. Despite the advances of benchmarks for table reasoning, existing work seldom evaluates whether internal constraints of tables are violated (e.g., assess whether subtotals match their components or whether ratios are valid). Such an ability to detect logical inconsistencies is critical in high-stakes domains such as finance and accounting. Unlike existing benchmarks, our work directly targets logical inconsistency detection within structured financial tables. Specifically, we (1) reformulate the task beyond QA toward identifying internal inconsistencies, and (2) propose a multi-dimensional evaluation protocol covering Error Count Prediction, Missing Count Estimation, Error Categorization, and Error Positioning.

3 Construction of SEC-FINTABLES

To facilitate fine-grained evaluation of LLMs in detecting logical inconsistencies, we construct a new benchmark, SEC-FINTABLES, following the pipeline illustrated in Fig. 2.

The benchmark is derived from 20 years of quarterly financial reports submitted by 500 publicly listed companies to the U.S. SEC, encompassing standard forms of financial statements, including balance sheets, cash flow statements, and income statements, which are publicly available and can be freely used for research purposes. These reports are stored in XBRL format, which ensures standardized structure and well-defined internal logic, making them well-suited for constructing high-quality tabular data with strong logical constraints.

In the *Table Prepraration* stage, We first extract structured financial tables from the original XBRL-based XML filings. To reduce the attention burden on LLMs caused by verbose XML formatting, we clean and normalize the tables and convert them into a unified CSV format. To ensure data quality, we retain only tables with well-defined semantics and standardized structure, resulting in a curated set of 11,134 authentic and high-quality table instances. We provide the distribution of tokens per table (using the GPT-2 tokenizer), highlighting variation in input sizes, as shown in Fig. 3.

In the *Cell Sampling and Value Parsing* stage, we initiate the process by randomly sampling a target cell containing a numerical value. Subsequently, each financial value is normalized into a canonical numeric representation, while preserving formatting metadata to allow for later restoration.

In the *Logical Association Retrieval* stage, we leverage a large language model (DeepSeek-V3) to identify the sampled target cell and its logically associated positions within the financial table. The entire table, along with the sampled cell, is serialized into plain text and provided to the model. We prompt the model (see Appendix A) to infer semantic constraints and retrieve all related cells that may be logically affected if the target value is perturbed. The model outputs these associated positions as a

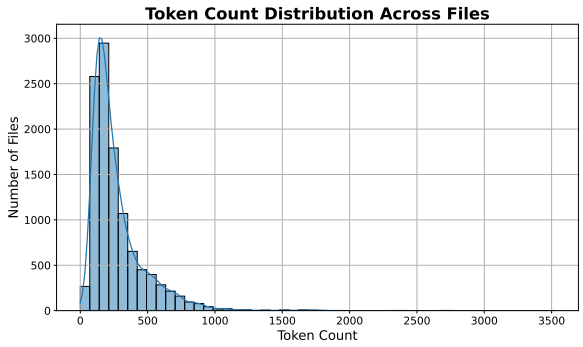

Figure 3: Distribution of token counts across table files.

set of row names and column indices, following the annotation format detailed in Appendix B. All retrieved annotations undergo manual cross-checking to ensure correctness. Specifically, for the 1,000-table test set, LLM initially identified 3,990 associated positions; Manual correction provided 4,364 refined positions, resulting in 534 total modifications (where each addition or removal counts as one modification).

In the *Error Injection* stage, we introduce three types of errors at the sampled positions: value violations (VAL), missing values (MISS), and formatting inconsistencies (FMT). We programmatically inject errors using predefined rules, randomly selecting error types while prioritizing those not yet present in the table to ensure diversity. The distribution of these three error types across the test set is summarized in Tab. 1. For Val errors, we apply predefined change scopes that perturb the original values within specified ranges (e.g., ±1%, ±10%, and ±50%). Modified positions and their logically associated locations are recorded as the annotation, including exact locations (`errLoc`), along with their types as the annotation error types (`errType`). After all sampled positions are processed, duplicates in `errLoc` are removed, and the annotations: total number of errors (`errNum`) and count of missing values (`missNum`) are finalized to complete the table-level annotation. Fig. 4 shows an illustration of these transformations, contrasting cleaned tables with their perturbed counterparts.

Based on the sampled subset of 7,097 and the automated pipeline, we generate synthetic tables at a real-to-synthetic ratio of 1:13, resulting in 92,261 annotated table instances. Including the original 11,134 real tables, the complete SEC-FINTABLES comprises a total of 103,395 instances.The original 11,134 tables are used as error-free reference samples in the evaluation.

Change Scope	VAL	FMT	MISS
1%	286	400	380
10%	292	393	378
50%	299	369	397

Table 1: Distribution of injected error types under different settings of change scope.

4 Evaluation Protocol

Our evaluation protocol is designed to evaluate three essential capabilities for detecting logical inconsistencies in structured financial tables:

1. **Recognition of constraints:** Models must verify explicit and implicit constraints, such as additive and ratio relations, range restrictions, and cross-dimensional consistency. Our evaluation protocol requires models to output all elements involved in the violated constraint, which guarantees solution uniqueness and ensures comprehensive error detection. This approach avoids attribution bias by not favoring models that tend to blame specific cell types (e.g., result cells versus component cells), and instead assesses whether models can correctly identify the complete set of constraint-related elements. This challenges LLMs to perform structural reasoning, effectively capturing table structures and comprehending the semantics of headers with domain-specific knowledge.

2. **Detection of anomalies:** Models must identify the exact cell locations in a table that violate logical constraints. This requires constraint grounding, structure–semantics alignment, and multi-step structural reasoning.

3. **Localization and classification:** Models must localize erroneous cells and classify their types, which reflects a model's capability for error-type understanding and integrated reasoning–classification.

To systematically evaluate the ability of LLMs to detect logical inconsistencies in structured tabular data, we design a multi-dimensional evaluation protocol that captures three core capabilities through four complementary sub-tasks, as illustrated in Fig. 5.

- **Error Count Prediction:** This task evaluates the model's ability to estimate the total number of erroneous cells in a given table. The evaluation protocol employs regression-based metrics, specifically Accuracy (errNumAcc), RMSE (errNumRMSE), MAE (errNumMAE), and R^2

	Three Months Ended		Six Months Ended	
	June 30		June 30	
	2023	2022	2023	2022
Total segment gross contribution	$1,031,053	$967,714	$2,008,163	$1,865,229
Costs and expenses:				
Cost of services and product development – unallocated (1)	15,286	15,728	18,666	27,536
Selling, general and administrative	680,168	604,911	1,337,258	1,222,815
Depreciation and amortization	46,613	47,664	93,244	96,013
Acquisition and integration charges	1,973	2,289	3,341	4,496
Gain from sale of divested operation	3,906	—	-135,410	—
Operating income	283,107	297,122	691,064	514,369
Interest expense and other, net	-18,983	-21,171	-48,740	-23,359
Gain on event cancellation insurance claims	—	—	3,077	—
Less: Provision for income taxes	66,081	71,026	151,575	113,570
Net income	$198,043	$204,925	$493,826	$377,440

(a) Cleaned table.

	Three Months Ended		Six Months Ended	
	June 30		June 30	
	2023	2022	2023	2022
Total segment gross contribution	$1,031,053	$967,746	$2,008,163	$1,865,229
Costs and expenses:				
Cost of services and product development – unallocated (1)	15,286	15,728	18,666	27,536
Selling, general and administrative	680,168	604,911	1,337,258	1,222,815
Depreciation and amortization	46,613	47,664	93,244	96,013
Acquisition and integration charges	1,973	2,289	3,341	4,496
Gain from sale of divested operation	3,906	—	-135,410	—
Operating income	283,107	297,122		514,369
Interest expense and other, net	-18,983	-21,171	-48,740	-23,359
Gain on event cancellation insurance claims	—	—	3,077	—
Less: Provision for income taxes	66,081	71,026	151,575	113,570
Net income	$198,043	$204,925	$493,826	$377,440

(b) Table with numerical and missing value errors.

Figure 4: Comparison of cleaned and perturbed tables. Red boxes mark injected errors (e.g., numerical or missing values). Pink and yellow regions indicate their affected scope. Examples of inputs and targets in SEC-FINTABLES.

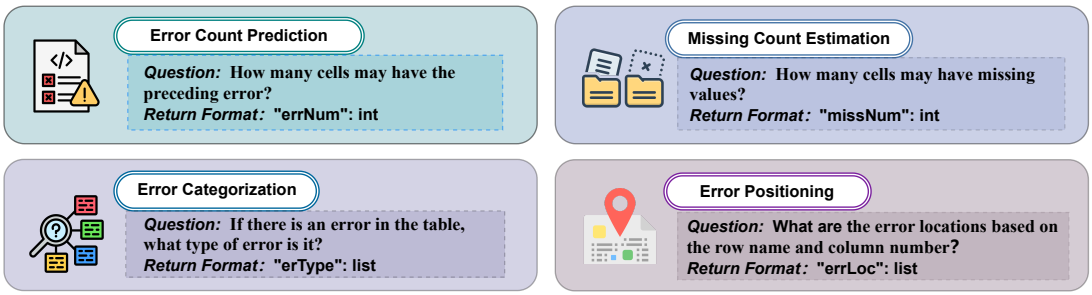

Figure 5: Exemplifying inputs and outputs of four evaluation sub-tasks.

(errNumR^2), to effectively measure both exact-match accuracy and prediction deviations from the ground truth.

- **Missing Count Estimation:** This task evaluates the model's capability to accurately estimate the total number of missing or removed cells (i.e., cells that are required but absent) in the given table. The evaluation protocol utilizes regression-based metrics, including Accuracy (missNumAcc), RMSE (missNumRMSE), MAE (missNumMAE), and R^2 (missNumR^2) for nuanced analysis of both exact estimation and robustness to deviation.

- **Error Categorization:** It evaluates the model's capability to distinguish different semantic error types, i.e., VAL, FMT, and MISS. Precision and F1 scores are reported for each class to reflect class-wise performance.

- **Error Positioning:** It locates the exact row names and column indices of the erroneous cells. It reflects the model's ability to correctly localize the erroneous cells in the table. The evaluation protocol reports cell-level Precision and the Set Intersection over Union (IoU) between predicted and ground-truth error spans across all tables.

We design a unified few-shot prompt template for all models as shown in Appendix C. Each prompt includes clear task definitions, detailed explanations of three error types (VAL, FMT, and MISS), and representative reasoning examples that guide the model through step-by-step error identification, categorization, and localization. Acknowledging that enforcing strict JSON-only outputs may degrade model performance, we follow prior work (Tam et al., 2024) by allowing free-form generation. The output is subsequently post-processed into a structured JSON object containing `errNum`, `missNum`, `errType`, and `errLoc`. For completeness, evaluation results under the direct JSON-output setting are provided in Appendix D.

During evaluation, all models are tested under three levels of change scope, enabling a comprehensive and systematic assessment.

5 Experiment

5.1 Experimental Settings

For the evaluations, we focus on several representative LLMs, including GPT-5, LLaMA 3.1 models of varying scales (8B and 70B) (Meta AI, 2024), Qwen 2.5 series (7B to 72B) (Qwen Team, 2024a), Qwen3-235B (Qwen Team, 2025),

QWQ-32B (Qwen Team, 2024b), and DeepSeek-R1 (DeepSeek Team, 2024). All models are evaluated under their default settings, without tool access. We use a unified prompt template (Appendix C), following a two-step process: free-form response generation and structured JSON summarization of detected inconsistencies. Each experiment is repeated three times. Tab. 2 and Tab. 3 report the mean values under three levels of Change Scopes $\in 0.01, 0.1, 0.5$, which control the magnitude of perturbations for VAL-type errors, while the corresponding variances are reported in Appendix E. We further include a trivial baseline that always predicts no missing values for the missNum task (Table 4).

5.2 Performance of Mainstream LLMs

5.2.1 Main Observations

We evaluate the ability of mainstream LLMs to detect logical inconsistencies in financial tables, focusing on the four sub-tasks defined in Sec. 4. The main observations are summarized below:

GPT-5 demonstrates a notable advantage in missing value estimation tasks. At the change scope of 10%, GPT-5 outperforms DeepSeek-R1 on the `missNum` task, achieving higher accuracy (0.872 vs. 0.823) while substantially reducing RMSE (1.878 vs. 2.170) and MAE (0.469 vs. 0.600). It also yields a less negative R^2 score (-12.036 vs. -16.409), indicating stronger overall robustness in missing value estimation.

DeepSeek-R1 exhibits comprehensive leading performance in structured reasoning tasks such as error localization and type classification. Across all change scopes, it consistently achieves the highest Precision and IoU in error localization (e.g., Precision = 0.420 at 10%). In error-type classification, it also delivers strong results, attaining the top MISS F1 (0.838) and competitive VAL F1 (0.695) at the change scope of 10%. These metrics underscore its balanced and superior capability in both fine-grained localization and semantic error categorization.

Qwen3-235B demonstrates strong performance in semantic error classification, consistently achieving top-tier F1 scores across different error types under various change scopes.

QWQ-32B exhibits a distinct strength in localization precision, consistently achieving the second-highest Precision score in error localization across all change scopes (e.g., 0.395 at 50%), while its performance in other quantitative and semantic tasks remains moderate.

The Qwen 2.5 series shows scaling gains in semantic and estimation tasks but not in structural reasoning. While its performance in `missNum` and `errType` improves with increasing model size, `errLoc` precision remains low and even drops at the largest scale, indicating that scaling fails to enhance structural reasoning.

5.2.2 Case Study

To better illustrate key observations, we provide a case study of a test sample in Appendix F.

5.2.3 Discussion

The observations from the evaluation collectively reveal that *progress in table-based logical inconsistency detection remains fragmented*. Scaling improves semantic discrimination and missing value estimation, but contributes little to fine-grained structural reasoning. Conversely, specialized models can excel at precise localization, yet struggle with quantitative estimation. Balanced models avoid extreme weaknesses but fail to dominate across dimensions.

Moreover, the experimental results reveal several important insights that advance our understanding of LLMs for logical consistency reasoning:

Non-monotonic Scaling Effects: Qwen2.5-32B outperforms its larger 72B counterpart in error classification (precision = 0.794), indicating that increased model capacity does not necessarily lead to better performance on tabular logical reasoning tasks.

Task-Specialized Capabilities Emerge: On missing value estimation (`missNum`), GPT-5 performs best, achieving the highest accuracy at the 10% change scope (0.872). DeepSeek-R1 excels in structural reasoning, achieving the best performance in both error count prediction (`errNum`, 0.205 at 10%) and error localization (`errLoc`, Precision/IoU = 0.420/0.325 at 10%). Qwen3-235B shows the strongest semantic classification capability, leading in `errType` F1 (e.g., FMT F1 = 0.651 at 10%).

Different Robustness to Change Scopes: The models show distinct trends in error localization performance with increasing change scope. Several models show moderate variation, including small increases in localization IoU with higher change scope (e.g., 0.276 to 0.301 for GPT-5 and

Change Scope	Model Name	errNum				missNum			
		Acc↑	RMSE↓	MAE↓	R^2 ↑	Acc↑	RMSE↓	MAE↓	R^2 ↑
0.01	GPT-5	0.163	6.996	4.879	-1.118	**0.861**	**1.848**	**0.452**	**-11.122**
	LLaMA-3.1-8B	0.112	12.297	7.119	-5.544	0.372	5.825	2.499	-119.387
	LLaMA-3.1-70B	0.125	10.163	5.926	-3.470	0.450	4.471	1.889	-69.926
	Qwen2.5-7B	0.109	7.892	5.704	-1.695	0.319	3.319	1.894	-38.085
	Qwen2.5-14B	0.091	6.539	4.783	-0.850	0.407	2.754	1.466	-25.920
	Qwen2.5-32B	0.159	**6.391**	**4.271**	**-0.768**	0.475	3.276	1.310	-37.091
	Qwen2.5-72B	0.161	7.005	4.441	-1.124	0.530	4.760	1.423	-79.408
	DeepSeek-R1	**0.189**	9.334	5.836	-2.771	0.830	2.056	0.583	-14.007
	Qwen3-235B	0.166	6.782	4.635	-0.991	0.814	2.117	0.535	-14.909
	QWQ-32B	0.109	9.850	6.358	-3.199	0.545	4.296	1.926	-64.493
0.1	GPT-5	0.172	7.544	5.122	-1.380	**0.872**	**1.878**	**0.469**	**-12.036**
	LLaMA-3.1-8B	0.105	10.710	6.940	-3.796	0.349	6.690	2.791	-164.429
	LLaMA-3.1-70B	0.123	9.104	5.631	-2.466	0.467	4.360	1.865	-69.268
	Qwen2.5-7B	0.095	8.678	6.272	-2.149	0.292	3.796	2.154	-52.262
	Qwen2.5-14B	0.101	6.750	4.699	-0.905	0.408	2.914	1.515	-30.377
	Qwen2.5-32B	0.143	**5.679**	**4.041**	**-0.349**	0.496	2.511	1.148	-22.308
	Qwen2.5-72B	0.156	6.001	4.199	-0.506	0.531	3.046	1.259	-33.304
	DeepSeek-R1	**0.205**	8.796	5.581	-2.236	0.823	2.170	0.600	-16.409
	Qwen3-235B	0.194	6.566	4.510	-0.803	0.817	2.000	0.550	-13.792
	QWQ-32B	0.110	10.700	6.846	-3.788	0.547	4.527	1.926	-74.734
0.5	GPT-5	0.182	7.010	4.791	-1.047	**0.867**	**1.829**	**0.443**	**-11.204**
	LLaMA-3.1-8B	0.096	12.652	7.303	-5.667	0.332	9.339	3.110	-316.980
	LLaMA-3.1-70B	0.128	10.452	5.998	-3.550	0.433	4.536	1.937	-74.014
	Qwen2.5-7B	0.094	8.191	5.948	-1.795	0.296	3.231	1.877	-37.070
	Qwen2.5-14B	0.114	6.990	4.893	-1.035	0.415	2.997	1.552	-31.750
	Qwen2.5-32B	0.132	6.169	**4.265**	-0.585	0.474	2.939	1.293	-30.492
	Qwen2.5-72B	0.165	**6.146**	4.323	**-0.573**	0.531	3.096	1.272	-33.953
	DeepSeek-R1	**0.201**	9.306	5.777	-2.607	0.829	1.960	0.562	-13.002
	Qwen3-235B	0.179	7.995	4.787	-1.663	0.813	2.693	0.618	-25.442
	QWQ-32B	0.109	10.230	6.393	-3.359	0.543	4.697	1.956	-79.436

Table 2: Results of Error Count Prediction (`errNum`) and Missing Count Estimation (`missNum`) under different levels of injected errors. Numbers in **bold** indicate the best results and underlined numbers are the second-best results for each metric within each error level. For ↑ metrics, higher is better; for ↓ metrics, lower is better.**Change scope applies only to value (VAL) errors; FMT and MISS do not involve value perturbation.**

0.259 to 0.282 for Qwen3-235B). In contrast, some models remain largely stable across change scopes (e.g., 0.321 at both 1% and 50% for DeepSeek-R1). Overall, error localization performance exhibits model-dependent behavior as the degree of change increases.

Different Difficulty across Error Types: The difficulty of error-type classification is not uniform across categories. MISS-type errors are consistently the easiest to detect, achieving the highest F1 scores (up to 0.851 at the 50% change scope), whereas FMT and VAL errors are more challenging, with the best F1 scores remaining around 0.63–0.72.

Based on these observations, we suggest the following guidance on model selection:

For missing-value estimation (missNum), GPT-5 demonstrates the strongest performance and is a suitable candidate for task-specific fine-tuning in missing-value detection scenarios.

For error count prediction (errNum), DeepSeek-R1 achieves the best results and can be selected for further fine-tuning in accurate error quantity estimation.

For error localization (errLoc), DeepSeek-R1 also shows superior structural reasoning and is recommended for fine-tuning in applications requiring precise error positioning.

For error-type classification (errType), Qwen3-235B delivers the strongest semantic classification performance and can be further adapted for fine-grained error-type discrimination tasks.

Change Scope	Model Name	errType						errLoc	
		FMT		MISS		VAL			
		Prec.↑	F1↑	Prec.↑	F1↑	Prec.↑	F1↑	Precision↑	IoU↑
1%	GPT-5	0.489	0.631	0.674	0.795	**0.601**	**0.641**	0.284	0.276
	LLaMA-3.1-8B	0.405	0.528	0.400	0.492	0.283	0.401	0.204	0.116
	LLaMA-3.1-70B	0.475	0.551	0.436	0.559	0.299	0.420	0.230	0.173
	Qwen2.5-7B	0.426	0.561	0.415	0.526	0.310	0.453	0.170	0.125
	Qwen2.5-14B	0.437	0.598	0.454	0.605	0.351	0.471	0.158	0.110
	Qwen2.5-32B	0.520	0.620	0.467	0.621	0.360	0.479	0.171	0.161
	Qwen2.5-72B	**0.598**	0.633	0.503	0.649	0.364	0.471	0.125	0.188
	DeepSeek-R1	0.527	0.641	**0.748**	**0.848**	0.569	0.641	**0.421**	**0.321**
	Qwen3-235B	0.525	**0.659**	0.730	0.830	0.590	0.625	0.241	0.259
	QWQ-32B	0.490	0.632	0.537	0.693	0.441	0.564	0.347	0.211
10%	GPT-5	0.496	0.638	0.691	0.813	0.655	**0.722**	0.287	0.288
	LLaMA-3.1-8B	0.409	0.540	0.375	0.466	0.289	0.407	0.193	0.112
	LLaMA-3.1-70B	0.480	0.557	0.441	0.569	0.319	0.445	0.251	0.188
	Qwen2.5-7B	0.411	0.548	0.398	0.508	0.304	0.443	0.171	0.113
	Qwen2.5-14B	0.438	0.600	0.458	0.611	0.364	0.490	0.149	0.115
	Qwen2.5-32B	0.508	0.615	0.466	0.619	0.375	0.499	0.165	0.166
	Qwen2.5-72B	**0.574**	0.622	0.479	0.618	0.378	0.478	0.118	0.188
	DeepSeek-R1	0.534	0.643	0.729	0.838	0.606	0.695	**0.420**	**0.325**
	Qwen3-235B	0.524	**0.651**	**0.737**	**0.840**	**0.663**	0.714	0.237	0.285
	QWQ-32B	0.480	0.624	0.532	0.687	0.493	0.627	0.367	0.205
50%	GPT-5	0.467	0.609	0.678	0.800	0.608	0.697	0.305	0.301
	LLaMA-3.1-8B	0.376	0.507	0.403	0.487	0.290	0.406	0.227	0.117
	LLaMA-3.1-70B	0.444	0.527	0.445	0.567	0.313	0.436	0.257	0.183
	Qwen2.5-7B	0.400	0.545	0.423	0.532	0.315	0.459	0.177	0.109
	Qwen2.5-14B	0.408	0.567	0.479	0.626	0.378	0.503	0.163	0.117
	Qwen2.5-32B	0.481	0.584	0.474	0.623	0.393	0.516	0.188	0.169
	Qwen2.5-72B	**0.546**	0.609	0.506	0.645	0.384	0.489	0.131	0.194
	DeepSeek-R1	0.488	0.598	**0.758**	**0.851**	0.634	**0.720**	**0.416**	**0.321**
	Qwen3-235B	0.493	**0.630**	0.744	0.841	**0.654**	0.706	0.245	0.282
	QWQ-32B	0.450	0.594	0.550	0.704	0.492	0.624	0.395	0.229

Table 3: Fine-grained error categorization (`errType`) and error localization (`errLoc`) results under different error injection levels. For `errType`, we report Precision and F1 for each error category (FMT, MISS, VAL). For `errLoc`, we report Precision and IoU.**Change scope applies only to value (VAL) errors; FMT and MISS do not involve value perturbation.**

5.3 Cause of LLMs' Poor Performance

During the preprocessing for SEC-FINTABLES, SEC HTML filings are converted to CSV, which reduces tag-induced distractions and makes column semantics explicit. One assumption of our evaluation protocol is that the model can correctly interpret column headers. Without such an assumption, it is hard to judge whether poor performance stems from deeper reasoning failures or simply from poor understanding of column semantics.

To disentangle header understanding from reasoning and support the above assumption, we conduct an additional study on 10 sampled complex tables (200 cells) from the test set in SEC-FINTABLES. Given a numeric cell, we request models to retrieve its row and column names. Tab. 5 shows the results of three models, and we can observe that the interpretation accuracy is high, indicating that column semantics are generally well recognized. Hence, the cause of poor performance is not the misinterpretation of column headers, and the failures stem from reasoning beyond headers, such as error-type discrimination and cross-cell constraint verification.

5.4 Future Directions for Improvements

We suggest the following directions for targeted improvements:

Based on non-monotonic scaling effects, future work may examine the underlying mechanisms and

Change Scope	missNumAcc	missNumMAE	missNumRMSE	missNumR^2
1%	0.620	0.399	0.664	-0.565
10%	0.622	0.393	0.652	-0.571
50%	0.603	0.412	0.666	-0.619

Table 4: Performance of a trivial baseline that always predicts no missing values.

Model	Accuracy
DeepSeek-R1	0.887
GPT-5	**0.946**
Qwen2.5-32B	0.811

Table 5: Accuracy of header understanding.

how model scaling interacts with cross-cell dependency modeling and logical constraint verification, while exploring training and architectural designs that better align model capacity with structured tabular reasoning requirements.

Based on the emergence of task-specialized capabilities across models, future work may develop unified models that achieve balanced performance across structural reasoning, semantic classification, and quantitative estimation.

Based on model-dependent robustness to change scopes, future work may improve the stability of logical error localization under varying perturbation magnitudes.

Based on the differing difficulty across error types, future work may improve models' sensitivity to subtle numerical and formatting inconsistencies, which remain more challenging than missing-value detection.

6 Conclusion

This paper presents the first benchmark SEC-FINTABLES and an evaluation protocol for assessing the ability of LLMs to detect logical inconsistencies in tabular data. The observations from our evaluation reveal that progress in table-based logical inconsistency detection remains fragmented. Scaling improves semantic discrimination and missing value estimation, but contributes little to fine-grained structural reasoning. Conversely, specialized models can excel at precise localization, yet struggle with quantitative estimation. Balanced models avoid extreme weaknesses but fail to dominate across dimensions. These findings highlight the inherent trade-offs between structural awareness, semantic understanding, and numerical verification, and point toward promising directions for developing future models capable of robust logical reasoning over high-stakes structured tabular data. We believe SEC-FINTABLES and the evaluation protocol can serve as a practical resource for advancing reliable inconsistency detection of LLMs in tabular reasoning.

As this work focuses on a single-table setting, in the future, we plan to extend the benchmark and evaluation protocol to consider more complex, cross-table logical inconsistencies to better evaluate the logical reasoning ability of LLMs.

Limitations

Although our work establishes the first benchmark for evaluating the ability of LLMs to detect logical inconsistencies in structured tabular data, the current evaluation protocol primarily targets single-table inference and does not evaluate cross-table logical inconsistency detection. Considering that state-of-the-art LLMs still struggle to achieve reliable performance even in single-table scenarios, this work focuses on the single-table setting as the first step towards evaluating LLMs' logical inconsistency detection ability. The more complex, cross-table setting is left as future work.

Ethical Considerations

As this work focuses on logical inconsistency detection using publicly available data and does not involve personal or sensitive information, the ethical risk is minimal.

Acknowlegments

This work is supported by the National Key Research and Development Program of China (No. 2025YFE0113500), the National Science Fund for Distinguished Young Scholars (No. 62525605), and the National Natural Science Foundation of China (No. 62572410, No. 62272401, and No. U22B2051).

References

Hammaad Adam, Junjing Lin, Jianchang Lin, Hillary Keenan, Ashia Wilson, and Marzyeh Ghassemi. 2024. Clinical information extraction with large language models: A case study on organ procurement. *AMIA Annual Symposium Proceedings*, 2024:115–123.

Vikram Aggarwal, Jay Kulkarni, Aditi Mascarenhas, Aakriti Narang, Siddarth Raman, Ajay Shah, and Susan Thomas. 2025. Information extraction from fiscal documents using llms. *Preprint*, arXiv:2511.10659.

Mehmet F Bagci, Yusuf Ozturk, Truong Nguyen, Brian D Modena, Toan Do, Samantha R Spierling Bagsic, and Anna L Ritko. 2025. Llm-driven lab result extraction from electronic health records. In *IEEE EMBS BHI 2025 Conference*. Published in BHI 2025 Proceedings.

Wenhu Chen, Ming-Wei Chang, Eva Schlinger, William Yang Wang, and William W. Cohen. 2021. Open question answering over tables and text. In *ICLR*. https://openreview.net/forum?id=MmCRswl1UYl.

Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. 2020a. Tabfact: A large-scale dataset for table-based fact verification. In *ICLR*. https://openreview.net/forum?id=rkeJRhNYDH.

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

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

DeepSeek Team. 2024. Deepseek-v3 technical report. *arXiv Preprint*. https://arxiv.org/abs/2412.19437.

Besnik Fetahu, Avishek Anand, and Maria Koutraki. 2019. Tablenet: An approach for determining fine-grained relations for wikipedia tables. In *WWW*, pages 2736–2742.

Majid Ghasemi-Gol, Jay Pujara, and Pedro A. Szekely. 2019. Tabular cell classification using pre-trained cell embeddings. In *ICDM*, pages 230–239.

Vivek Gupta, Maitrey Mehta, Pegah Nokhiz, and Vivek Srikumar. 2020. INFOTABS: inference on tables as semi-structured data. In *ACL*, pages 2309–2324.

Alex Kim, Maximilian Muhn, and Valeri Nikolaev. 2025a. Financial statement analysis with large language models. *Papers*.

Alex Kim, Maximilian Muhn, and Valeri Nikolaev. 2025b. Financial statement analysis with large language models. *Preprint*, arXiv:2407.17866.

Jeonghyeon Kim, Aran Park, and Sangjin Lee. 2016. Recovery method of deleted records and tables from ESE database. *Digit. Investig.*, 18:S118–S124.

Meta AI. 2024. The llama 3 herd of models. *arXiv Preprint*. https://arxiv.org/abs/2407.21783.

Linyong Nan, Chiachun Hsieh, Ziming Mao, Xi Victoria Lin, Neha Verma, Rui Zhang, Wojciech Kryscinski, Hailey Schoelkopf, Riley Kong, Xiangru Tang, Mutethia Mutuma, Ben Rosand, Isabel Trindade, Renusree Bandaru, Jacob Cunningham, Caiming Xiong, and Dragomir R. Radev. 2022. Fetaqa: Free-form table question answering. *Trans. Assoc. Comput. Linguistics*, 10:35–49.

OpenAI. 2025. Introducing gpt-5. https://openai.com/zh-Hans-CN/index/introducing-gpt-5/?utm_source=chatgpt.com. Accessed: 2025-12-XX.

Paolo Papotti. 2022. Technical perspective of TURL: table understanding through representation learning. *SIGMOD Rec.*, 51(1):32.

Liang Qiao, Zaisheng Li, Zhanzhan Cheng, Peng Zhang, Shiliang Pu, Yi Niu, Wenqi Ren, Wenming Tan, and Fei Wu. 2021. LGPMA: complicated table structure recognition with local and global pyramid mask alignment. In *ICDAR*, volume 12821, pages 99–114.

Qwen Team. 2024a. Qwen2.5 technical report. *arXiv Preprint*. https://arxiv.org/abs/2412.15115.

Qwen Team. 2024b. Qwq-32b: Embracing the power of reinforcement learning. https://qwenlm.github.io/blog/qwq-32b/.

Qwen Team. 2025. Qwen3 technical report. *arXiv Preprint*. https://arxiv.org/abs/2505.09388.

Shruti Singh, Nandan Sarkar, and Arman Cohan. 2024. Scidqa: A deep reading comprehension dataset over scientific papers. In *EMNLP*, pages 20908–20923.

Gadir Suleymanli, Alexander Rogiers, Lucas Lageweg, and Jefrey Lijffijt. 2025. Are llms ready to help non-expert users to make charts of official statistics data? *Preprint*, arXiv:2510.01197.

Zhi Rui Tam, Cheng-Kuang Wu, Yi-Lin Tsai, Chieh-Yen Lin, Hung yi Lee, and Yun-Nung Chen. 2024. Let me speak freely? a study on the impact of format restrictions on performance of large language models. *Preprint*, arXiv:2408.02442.

Sainan Tong, Derong Shen, Yue Kou, and Tiezheng Nie. 2022. CFCT: the cell function classification method for complex tables. In *HPCC/DSS/SmartCity/DependSys*, pages 2206–2213.

James Wagner, Alexander Rasin, and Jonathan Grier. 2015. Database forensic analysis through internal structure carving. *Digit. Investig.*, 14:S106–S115.

Wenyuan Xue, Baosheng Yu, Wen Wang, Dacheng Tao, and Qingyong Li. 2021. Tgrnet: A table graph reconstruction network for table structure recognition. In *ICCV*, pages 1275–1284.

Xinqi Yang, Scott Zang, Yong Ren, Dingjie Peng, and Zheng Wen. 2024. Evaluating large language models on financial report summarization: An empirical study. *CoRR*, abs/2411.06852.

Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. 2020. Tabert: Pretraining for joint understanding of textual and tabular data. In *ACL*, pages 8413–8426.

Xiao Yu Cindy Zhang, Melissa Fong, Wyeth Wasserman, and Jian Zhu. 2025. CaseReportCollective: A large-scale LLM-extracted dataset for structured medical case reports. In *Proceedings of the 24th Workshop on Biomedical Language Processing*, pages 249–262, Vienna, Austria. Association for Computational Linguistics.

Xinyi Zheng, Douglas Burdick, Lucian Popa, Xu Zhong, and Nancy Xin Ru Wang. 2021. Global table extractor (GTE): A framework for joint table identification and cell structure recognition using visual context. In *WACV*, pages 697–706.

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 *ACL/IJCNLP*, pages 3277–3287.

A Prompt designed to identify cells based on model-inferred semantic constraints.

We design a prompt to extract target cells and their logically associated positions from financial tables. The prompt is displayed in Fig. 14.

B Logical Association Retrieval

We provide an example of extracted target cells and their associated positions in Fig. 6.

```json
[
    "csv_name": "08004.csv",
    "target_value_row_name": "Net loss",
    "target_value_row_col": 1,
    "target_value": "$(124)",
    "correspond_area": [
        {
          "row_name": "Loss before income taxes",
          "col_idx": 1
        },
        {
          "row_name": "Income tax benefit",
          "col_idx": 1
        }
      ]
]
```

Figure 6: An example of logical association annotation for a sampled target cell in the table file `08004.csv`.

C Prompt Template for All Models

We design a prompt-based evaluation framework to assess the ability of large language models to detect logical inconsistencies. The framework adopts a two-step prompting strategy: the first step elicits free-form analysis of potential inconsistencies, while the second step requires structured JSON outputs for systematic evaluation. The two prompts are illustrated in Fig. 15 and Fig. 16.

D Evaluation under Direct JSON-Output Setting

To complement our main experiments that use two-stage outputs, we further conduct evaluation on the direct JSON-output setting. Under this setting, we evaluate LLMs' ability to detect logical inconsistencies using the prompt shown in Fig. 17. For fair comparison, all models are evaluated using deterministic decoding, with decoding settings reported in Tab. 6. As GPT-5 does not support setting the temperature to zero, we instead use GPT-4. The corresponding results are reported in Tabs. 7 and 8. Overall, DeepSeek-R1 exhibits a noticeable performance drop under the direct JSON-output setting, while other models show slight gains in Error Categorization and Missing Count Estimation but also experience mild degradation in Error Positioning and Error Count Prediction. The observations on the direct JSON-output setting are generally consistent with our observations in the main experiments (e.g., the persistent challenge of Error Positioning). Compared to the two-step setting used in the main evaluation, models show better accuracy on the direct JSON-output setting.

Parameter	Value
Temperature	0
Top-p	1.0
Top-k	1
Sampling	Off
Max tokens	10,000

Table 6: Decoding configuration used for all models under the direct JSON-output setting.

E Variance of Evaluation Results

Tab. 9 and Tab. 10 report the variance of the evaluation results computed over three independent runs in our main evaluation. The observed variances are relatively small across runs, indicating that our conclusions are unlikely to be affected by the probabilistic nature of LLM outputs.

F Case Study

In the following, we provide a representative case (ID 08037) at the change scope of 1% in Fig. 7 to illustrate key observations in our main evaluation. The ground truth contains a single MISS error in the "Diluted" row, as demonstrated in Fig. 8.

Year	2017 (A)	2016 (A)	2017 (B)	2016 (B)
Revenues	2,631,137	2,174,123	7,234,193	5,970,857
Gross profit	355,830	331,409	978,446	796,989
Selling, general and administrative expenses	205,025	184,014	614,819	541,328
Amortization of intangible assets	10,022	11,576	29,926	34,696
Net income from continuing operations	89,839	75,866	205,344	118,853
Net income attributable to common stock	89,303	75,456	204,112	117,913
Basic	0.56	0.48	1.29	0.73
Diluted	0.56		1.28	0.73

Figure 7: Exemplifying case (ID 08037) at the change scope of 1% in our main evaluation.

```json
{
   "errNum": 1,
   "errLoc": [
      {"row_name": "Diluted","col_idx": 2}
      ],
   "errType":[
            "MISS"
      ],
   "missNum": 1,
   }
```

Figure 8: Ground truth annotation.

We observe that LLMs (GPT-5, DeepSeek-R1, Qwen2.5-32B, Qwen2.5-72B, and QWQ-32B) correctly identify and localize the error.

Models of different scales consistently exhibit systematic confusion in identifying MISS-type errors. LLaMA3.1-8B and Qwen2.5-14B identify the MISS-type error as MISS, FMT, and VAL errors. LLaMA3.1-70B, Qwen2.5-7B and Qwen3-235B identify the MISS-type error as both MISS and VAL errors.

Even when the target cell is partially covered (e.g., by LLaMA-3.1-70B), localization precision remains insufficient. The `errLoc` outputs of LLaMA-3.1-8B, LLaMA-3.1-70B, Qwen2.5-7B, Qwen2.5-14B, and Qwen3-235B are shown in Figs. 9, 10, 11, 12, and 13, respectively.

Overall, we can observe that three recurring patterns emerge: (i) larger models are more reliable at recognizing error categories; (ii) smaller models tend to over-identify and confuse error types; and (iii) fine-grained localization is a universal challenge. While we only present one case due to space constraints, similar misclassification and localization issues can be consistently observed in other samples, confirming the representativeness of these findings.

```json
[
    { "row_name": "Amortization of intangible assets",
      "col_idx": 1 },
    { "row_name": "Diluted",
      "col_idx": 1 },
    { "row_name": "Selling, general and
administrative expenses",
      "col_idx": 1 },
    { "row_name": "Amortization of intangible assets",
      "col_idx": 2 },
    { "row_name": "Net income from continuing
operations",
      "col_idx": 1 },
    { "row_name": "Net income from continuing
operations attributable to common stock",
      "col_idx": 1 },
    { "row_name": "Earnings per share from continuing
operations: Basic",
      "col_idx": 1 },
    { "row_name": "Revenues",
      "col_idx": 1 },
    { "row_name": "Gross profit",
      "col_idx": 1 }
]
```

Figure 9: The `errLoc` output of LLaMA-3.1-8B.

```json
[
    { "row_name": "Diluted",
      "col_idx": 2 },
    { "row_name": "Earnings per share from continuing
operations:",
      "col_idx": 1 },
    { "row_name": "Earnings per share from continuing
operations:",
      "col_idx": 3 }
]```

Figure 10: The `errLoc` output of LLaMA-3.1-70B.
```

```json
[
    { "row_name": "Net income from continuing
operations",
      "col_idx": 2 },
    { "row_name": "Total revenues",
      "col_idx": 3 },
    { "row_name": "Employee compensation and
benefits",
      "col_idx": 1 },
    { "row_name": "Non-cash stock-based compensation",
      "col_idx": 1 },
    { "row_name": "Other operating expenses",
      "col_idx": 1 },
    { "row_name": "Amortization of intangible assets",
      "col_idx": 1 },
    { "row_name": "Interest",
      "col_idx": 1 }
]
```

Figure 11: The errLoc output of Qwen2.5-7B.

```json
[
    { "row_name": "Selling, general and
administrative expenses",
      "col_idx": 2 },
    { "row_name": "Basic Earnings Per Share",
      "col_idx": 4 },
    { "row_name": "Net income from continuing
operations",
      "col_idx": 3 },
    { "row_name": "Net income from continuing
operations",
      "col_idx": 2 }
]
```

Figure 12: The errLoc output of Qwen2.5-14B.

```json
[
    { "row_name": "Diluted",
      "col_idx": 2 },
    { "row_name": "Net income from continuing
operations",
      "col_idx": 1 },
    { "row_name": "Net income from continuing
operations",
      "col_idx": 2 },
    { "row_name": "Net income from continuing
operations",
      "col_idx": 3 },
    { "row_name": "Net income from continuing
operations",
      "col_idx": 4 },
    { "row_name": "Net income from continuing
operations attributable to common stock",
      "col_idx": 1 },
    { "row_name": "Net income from continuing
operations attributable to common stock",
      "col_idx": 2 },
    { "row_name": "Net income from continuing
operations attributable to common stock",
      "col_idx": 3 },
    { "row_name": "Net income from continuing
operations attributable to common stock",
      "col_idx": 4 }
]
```

Figure 13: The errLoc output of Qwen3-235B.

Change Scope	Model Name	errNum				missNum			
		Acc↑	RMSE↓	MAE↓	R^2 ↑	Acc↑	RMSE↓	MAE↓	R^2 ↑
0.01	GPT-4	**0.295**	**5.630**	**3.757**	**-0.372**	0.715	2.354	0.491	-18.663
	LLaMA-3.1-8B	0.117	6.074	4.663	-0.597	0.393	4.140	1.456	-59.831
	LLaMA-3.1-70B	0.174	6.284	4.468	-0.709	0.493	2.378	1.078	-19.071
	Qwen2.5-7B	0.093	7.378	5.543	-1.356	0.260	3.876	2.389	-52.304
	Qwen2.5-14B	0.144	5.697	4.142	-0.405	0.398	2.918	1.576	-29.206
	Qwen2.5-32B	0.149	6.709	4.397	-0.948	0.472	3.831	1.549	-51.090
	Qwen2.5-72B	0.197	7.073	4.465	-1.165	0.553	4.497	1.521	-70.778
	DeepSeek-R1	0.155	8.498	5.041	-2.126	0.548	4.585	1.626	-73.585
	Qwen3-235B	0.191	6.103	4.268	-0.612	**0.828**	**1.628**	**0.451**	**-8.407**
	QWQ-32B	0.111	11.112	7.137	-4.344	0.528	5.232	2.304	-96.147
0.1	GPT-4	**0.302**	**5.727**	**3.792**	**-0.371**	0.722	**0.920**	**0.412**	**-2.127**
	LLaMA-3.1-8B	0.192	6.591	4.907	-0.816	0.483	2.506	1.192	-22.204
	LLaMA-3.1-70B	0.164	6.514	4.445	-0.775	0.507	2.977	1.129	-31.752
	Qwen2.5-7B	0.080	8.215	6.119	-1.822	0.258	4.092	2.492	-60.903
	Qwen2.5-14B	0.148	6.093	4.177	-0.553	0.402	3.492	1.648	-44.064
	Qwen2.5-32B	0.173	6.253	4.235	-0.635	0.465	3.754	1.633	-51.083
	Qwen2.5-72B	0.212	6.259	4.191	-0.638	0.571	3.667	1.407	-48.695
	DeepSeek-R1	0.148	7.901	5.086	-1.611	0.551	3.575	1.617	-46.233
	Qwen3-235B	0.201	7.386	4.660	-1.281	**0.819**	1.922	0.499	-12.650
	QWQ-32B	0.103	11.570	7.373	-4.598	0.528	6.071	2.391	-135.207
0.5	GPT-4	**0.319**	**5.768**	**3.800**	**-0.386**	0.699	2.285	0.517	-18.037
	LLaMA-3.1-8B	0.192	6.846	5.178	-0.952	0.442	3.314	1.454	-39.036
	LLaMA-3.1-70B	0.183	6.544	4.508	-0.784	0.515	2.762	1.081	-26.817
	Qwen2.5-7B	0.067	8.276	6.224	-1.853	0.227	4.163	2.583	-62.200
	Qwen2.5-14B	0.152	7.061	4.603	-1.077	0.408	4.122	1.759	-60.953
	Qwen2.5-32B	0.151	6.749	4.486	-0.897	0.488	4.496	1.732	-72.712
	Qwen2.5-72B	0.188	6.696	4.402	-0.868	0.552	4.137	1.584	-61.402
	DeepSeek-R1	0.143	8.324	5.333	-1.886	0.560	3.701	1.661	-48.942
	Qwen3-235B	0.199	6.508	4.378	-0.764	**0.828**	**1.607**	**0.420**	**-8.422**
	QWQ-32B	0.112	11.514	7.236	-4.522	0.513	5.174	2.199	-96.620

Table 7: Results of Error Count Prediction (`errNum`) and Missing Count Estimation (`missNum`) under different levels of injected errors on the direct JSON-output setting. Numbers in **bold** indicate the best results and underlined numbers are the second-best results for each metric within each error level. For ↑ metrics, higher is better; for ↓ metrics, lower is better.

Change Scope	Model Name	errType						errLoc	
		FMT		MISS		VAL			
		Prec.↑	F1↑	Prec.↑	F1↑	Prec.↑	F1↑	Precision↑	IoU↑
1%	GPT-4	**0.832**	0.512	0.749	0.704	0.554	0.344	0.115	**0.342**
	LLaMA-3.1-8B	0.424	0.529	0.406	0.454	0.391	0.402	0.112	0.116
	LLaMA-3.1-70B	0.588	0.425	0.443	0.495	0.416	0.480	0.161	0.203
	Qwen2.5-7B	0.432	0.562	0.425	0.538	0.402	0.539	0.205	0.112
	Qwen2.5-14B	0.483	0.568	0.470	0.609	0.487	0.562	0.137	0.147
	Qwen2.5-32B	0.516	0.614	0.490	0.639	0.519	0.580	0.173	0.178
	Qwen2.5-72B	0.680	0.653	0.543	0.660	0.481	0.526	0.125	0.220
	DeepSeek-R1	0.521	0.648	0.524	0.681	0.571	0.612	0.289	0.220
	Qwen3-235B	0.532	**0.664**	**0.761**	**0.854**	**0.695**	**0.632**	0.245	0.286
	QWQ-32B	0.492	0.631	0.545	0.702	0.543	0.623	**0.393**	0.203
10%	GPT-4	**0.827**	0.479	0.730	0.693	0.500	0.300	0.118	**0.358**
	LLaMA-3.1-8B	0.405	0.445	0.443	0.357	0.451	0.409	0.112	0.208
	LLaMA-3.1-70B	0.598	0.450	0.456	0.517	0.428	0.489	0.175	0.202
	Qwen2.5-7B	0.410	0.532	0.409	0.523	0.418	0.555	0.196	0.107
	Qwen2.5-14B	0.489	0.580	0.461	0.605	0.476	0.549	0.156	0.156
	Qwen2.5-32B	0.508	0.613	0.479	0.624	0.518	0.578	0.174	0.185
	Qwen2.5-72B	0.676	**0.657**	0.548	0.664	0.528	0.562	0.125	0.224
	DeepSeek-R1	0.521	0.647	0.534	0.695	0.601	0.657	0.301	0.230
	Qwen3-235B	0.507	0.641	**0.740**	**0.843**	**0.759**	**0.701**	0.242	0.269
	QWQ-32B	0.477	0.624	0.540	0.694	0.588	0.673	**0.392**	0.201
50%	GPT-4	**0.831**	0.446	0.757	0.685	0.591	0.351	0.117	**0.368**
	LLaMA-3.1-8B	0.397	0.445	0.396	0.298	0.411	0.375	0.129	0.209
	LLaMA-3.1-70B	0.500	0.379	0.486	0.534	0.453	0.498	0.169	0.217
	Qwen2.5-7B	0.408	0.545	0.404	0.512	0.426	0.567	0.179	0.094
	Qwen2.5-14B	0.459	0.555	0.476	0.609	0.516	0.577	0.146	0.159
	Qwen2.5-32B	0.494	0.600	0.506	0.651	0.546	0.589	0.183	0.188
	Qwen2.5-72B	0.631	0.620	0.562	0.668	0.553	0.597	0.124	0.224
	DeepSeek-R1	0.482	**0.614**	0.541	0.695	0.613	0.671	0.300	0.219
	Qwen3-235B	0.479	0.612	**0.770**	**0.855**	**0.750**	**0.714**	0.252	0.291
	QWQ-32B	0.453	0.600	0.562	0.713	0.594	0.689	**0.400**	0.205

Table 8: Fine-grained error categorization (`errType`) and error localization (`errLoc`) results under different error injection levels on the direct JSON-output setting. For `errType`, we report Precision and F1 for each error category (FMT, MISS, VAL). For `errLoc`, we report Precision and IoU.

Change Scope	Model Name	Variance of errNum↓				Variance of missNum↓			
		Acc	RMSE	MAE	R^2	Acc	RMSE	MAE	R^2
0.01	GPT-5	**0.030**	8.344	6.580	-1.637	0.710	0.906	0.380	-1.883
	LLaMA-3.1-8B	0.050	8.896	6.770	-1.997	0.320	4.123	2.220	-58.775
	LLaMA-3.1-70B	0.120	12.289	6.310	-4.720	0.360	7.490	2.740	-196.257
	Qwen2.5-7B	0.050	8.501	6.490	-1.737	**0.300**	2.883	1.550	-28.219
	Qwen2.5-14B	0.080	9.147	6.340	-2.168	0.370	2.575	1.430	-22.312
	Qwen2.5-32B	**0.030**	7.669	6.000	-1.228	0.410	2.480	1.290	-20.624
	Qwen2.5-72B	0.050	7.717	6.090	-1.255	0.410	1.382	0.870	-5.716
	DeepSeek-R1	0.040	**7.540**	**5.830**	**-1.153**	0.660	1.315	0.550	-5.083
	Qwen3-235B	**0.030**	9.884	7.040	-2.700	0.690	**1.044**	**0.410**	**-2.833**
	QWQ-32B	**0.030**	9.627	7.190	-2.509	0.440	1.873	0.930	-11.342
0.1	GPT-5	0.150	6.671	4.940	-0.360	0.850	**1.543**	**0.400**	**-6.267**
	LLaMA-3.1-8B	0.080	11.335	7.600	-2.927	0.410	7.284	2.910	-160.985
	LLaMA-3.1-70B	0.140	11.201	6.350	-2.835	0.470	3.841	1.870	-44.038
	Qwen2.5-7B	**0.060**	6.874	5.490	-0.444	**0.340**	2.528	1.470	-18.511
	Qwen2.5-14B	0.120	6.746	4.910	-0.391	0.350	3.151	1.730	-29.321
	Qwen2.5-32B	0.110	**6.355**	**4.700**	**-0.234**	0.430	2.912	1.360	-24.893
	Qwen2.5-72B	0.180	6.457	4.670	-0.274	0.600	2.740	1.070	-21.931
	DeepSeek-R1	0.190	7.201	4.930	-0.585	0.750	1.905	0.790	-10.084
	Qwen3-235B	0.190	8.188	5.630	-1.049	0.850	1.673	0.480	-7.550
	QWQ-32B	0.120	14.199	7.830	-5.162	0.500	4.188	2.000	-52.557
0.5	GPT-5	0.030	7.514	5.920	-1.114	0.820	**0.574**	**0.210**	**-0.087**
	LLaMA-3.1-8B	**0.020**	21.605	9.020	-16.480	0.390	20.087	2.950	-1328.018
	LLaMA-3.1-70B	0.060	**6.478**	**4.970**	**-0.572**	0.350	3.596	2.050	-41.589
	Qwen2.5-7B	0.070	7.749	6.090	-1.249	**0.340**	2.715	1.450	-23.275
	Qwen2.5-14B	0.090	7.495	5.530	-1.103	0.440	2.343	1.170	-17.083
	Qwen2.5-32B	0.080	6.984	5.190	-0.826	0.550	1.345	0.710	-4.962
	Qwen2.5-72B	0.040	7.805	6.140	-1.281	0.580	1.838	0.780	-10.133
	DeepSeek-R1	0.090	8.671	6.060	-1.815	0.740	1.631	0.560	-7.762
	Qwen3-235B	0.050	8.628	6.770	-1.788	0.770	0.949	0.340	-1.964
	QWQ-32B	0.060	8.856	6.380	-1.937	0.540	2.804	1.260	-24.889

Table 9: Variance of `errNum` and `missNum` metrics in main evaluation. lower is better for ↓.

Change Scope	Model Name	Variance of MISS↓		Variance of FMT↓		Variance of VAL↓		Variance of errLoc↓	
		Precision	F1	Precision	F1	Precision	F1	Precision	Average IoU
0.01	GPT-5	0.000175	0.000241	0.000977	0.000705	0.001416	0.001538	0.000454	0.000072
	LLaMA-3.1-8B	0.000038	0.000009	**0.000013**	**0.000013**	**0.000000**	0.000001	0.000209	**0.000001**
	LLaMA-3.1-70B	**0.000000**	0.000008	0.000028	0.000130	0.000001	**0.000000**	**0.000003**	0.000010
	Qwen2.5-7B	0.107744	0.051627	0.065095	0.038159	0.071829	0.008979	0.024401	0.025424
	Qwen2.5-14B	0.072464	0.026570	0.057971	0.009662	0.128019	0.033816	0.022123	0.017397
	Qwen2.5-32B	0.042222	0.008889	0.102222	0.044444	0.122222	0.035556	0.018857	0.029584
	Qwen2.5-72B	0.066667	0.022222	0.115556	0.060000	0.135556	0.022222	0.009947	0.028838
	DeepSeek-R1	0.051111	**0.000000**	0.073333	0.028889	0.046667	0.004444	0.024855	0.032677
	Qwen3-235B	0.000019	0.013715	0.001270	0.023438	0.003044	0.017382	0.001128	0.001115
	QWQ-32B	0.051680	**0.000000**	0.077519	0.036176	0.062016	0.010336	0.029795	0.029990
0.1	GPT-5	0.000069	0.000096	0.000086	0.000295	0.000037	0.000112	0.000059	0.000092
	LLaMA-3.1-8B	0.000237	0.000208	**0.000021**	0.000062	**0.000042**	0.000048	0.000069	**0.000013**
	LLaMA-3.1-70B	0.000128	0.000147	0.000010	**0.000001**	0.000001	0.000007	**0.000002**	**0.000000**
	Qwen2.5-7B	0.128889	0.048889	0.100000	0.037778	0.084444	0.017778	0.022846	0.022148
	Qwen2.5-14B	0.074074	0.022989	0.076628	0.017880	0.120051	0.022989	0.016425	0.034084
	Qwen2.5-32B	0.062222	0.013333	0.113333	0.044444	0.126667	0.028889	0.015214	0.026316
	Qwen2.5-72B	0.064444	0.022222	0.100000	0.037778	0.140000	0.040000	0.009611	0.036106
	DeepSeek-R1	0.037778	**0.000000**	0.066667	0.020000	0.040000	0.015556	0.022141	0.039706
	Qwen3-235B	**0.000638**	0.082115	0.018247	0.074855	0.027580	0.011332	0.005082	0.000220
	QWQ-32B	0.049673	**0.000000**	0.070588	0.010458	0.075817	0.007843	0.030235	0.025951
0.5	GPT-5	0.000002	0.000037	0.000756	0.000028	0.000462	0.000054	**0.000000**	0.000018
	LLaMA-3.1-8B	**0.000000**	0.000327	**0.000005**	0.000123	**0.000003**	0.000174	0.000904	0.000397
	LLaMA-3.1-70B	0.000019	**0.000010**	0.000012	**0.000002**	0.000015	**0.000024**	0.000189	**0.000013**
	Qwen2.5-7B	0.124444	0.046667	0.077778	0.031111	0.097778	0.024444	0.019960	0.024062
	Qwen2.5-14B	0.057971	0.019324	0.057971	0.004831	0.108696	0.031401	0.016165	0.024702
	Qwen2.5-32B	0.064444	0.020000	0.091111	0.026667	0.120000	0.024444	0.022042	0.037242
	Qwen2.5-72B	0.075556	0.015556	0.102222	0.037778	0.126667	0.040000	0.007754	0.041052
	DeepSeek-R1	0.057778	**0.000000**	0.082222	0.033333	0.035556	0.006667	0.021400	0.030052
	Qwen3-235B	0.014824	0.069435	0.005505	0.035949	0.014633	0.041109	0.001659	0.001732
	QWQ-32B	0.053030	0.007576	0.073232	0.017677	0.055556	0.007576	0.033837	0.028694

Table 10: Variance of errType (FMT, MISS, VAL) and errLoc metrics in main evaluation. Lower is better for ↓.

Task Definition:
Input an Excel table where rows are identified by names, and there are business logic relationships between them (e.g., the "Total Sales" row depends on the "Unit Price" and "Quantity Sold" rows).
Given a target numerical value, output a list of tuples (row_name, column_index) representing the value in the table that are logically associated with the target row based on predefined rules.
Identify all table regions that are associated with the given number. The association is primarily reflected in the implicit computational relationships, such as the number being derived from calculations involving numbers in other regions.

Requirement:
1. Think step by step. Give the reasoning process.
2. Column_index starts from 1.
3. Do not directly return the region corresponding to the input value.
4. Return the result in the following format and make sure the result format is correct:
```python
[(row_name, column_index)]
```
5. If there is no corresponding answer, output empty list.
6. Note that in the vast majority of cases, only data with the same column idx will have associations.
Here is an example.
In the example, 28,754 belongs to row "Total operating expenses", and it is calculated by the value of Research and development, 13,774, column 3,
and the value of Selling, general and administrative, 14,980, column 3.

Table:
<Example Table>

Target numerical value:
28,754

***Ouput*:**
[(Research and development, 3), (Selling, general and administrative, 3)]

Table:
<Query Table>

Target Numerical Value:
<Sampled Value>

Output:

Figure 14: The prompt designed to identify cells based on model-inferred semantic constraints.

Task definition:
Given the definition of the following error types:
MISS: Empty in the cell value that should have data
FMT: The data in the cell is incorrectly formatted. Common formatting errors are: $$714,925%% and so on.Usually a row or column has the same format.
VAL: Based on the semantic relationship of the table header, for example, Purchase Obligations(1)+Other Obligations(2)+Long term accrued income taxes(3)=Total, verify the correctness of each column value. If the value does not conform to a semantic relationship, there is a numerical error.Numerical errors are based on miscalculations.

Assume that a given table has only the following types of errors:
MISS FMT VAL

The format of the given Excel table is as follows:
<Example Table>

Requirement:
1. Think step by step. Give the reasoning process:
2. Given a table, determine if there are any errors.
3. If there is an error in the table, please output the error type:
4. Count the number of cells that may have the preceding error:
5. Count the number of cells with missing values:
6.Based on the semantic relationship of the table header, for example, Purchase Obligations(1)+Other Obligations(2)+Long term accrued income taxes(3)=Total, verify the correctness of each column value. If the value does not conform to a semantic relationship, there is a VAL error,Verify that values conform to semantic relationships by column:
7. Return the error location based on the row name and column number:
8. Please describe the four results using natural language. Do not use JSON formatting or any other structured format. You may organize the answer in any free-form manner.

The example is provided only to illustrate the correct answer; you do not need to replicate its output format.

Example analysis: (Investment income,2) There is an error in the format, (Depreciation, 1) there is loss of data, because in the first column: Employee compensation and benefits+Change in acquisition earn-out payables+Other operating expenses+Depreciation
+Amortization+Interest+Non-cash stock-based compensation is not equal to Total expenses, then (Employee compensation and benefits, 1), (Change in acquisition earn-out payables, 1), (Other operating expenses, 1), (Depreciation, 1), (Amortization, 1), (Interest, 1), (Non-cash stock-based compensation, 1), (Total expenses, 1) There is a value error, there is a duplicate Depreciation, and only one errLoc entry is recorded
In the example, $$15,823 is belong to row "Core commissions and fees",col_idx is 1
<Example Table>

Output:
{
"errNum": 9,
"errLoc": [
{"row_name": "Depreciation","col_idx": 1},
{"row_name": "Employee compensation and benefits", "col_idx": 1},
{"row_name": "Change in acquisition earn-out payables","col_idx": 1},
…
{"row_name": "Total expenses","col_idx": 1}
],
"errType": ["VAL", "MISS", "FMT"],
"missNum": 1
}

Input:

<Query Table>

Output:

Figure 15: Prompt design for logical inconsistency identification (Step 1): free-form response generation.

Task definition:
Given the definition of the following error types:
MISS: Empty in the cell value that should have data
FMT: The data in the cell is incorrectly formatted. Common formatting errors are: $$714,925%% and so on.Usually a row or column has the same format.
VAL: Based on the semantic relationship of the table header, for example, Purchase Obligations(1)+Other Obligations(2)+Long term accrued income taxes(3)=Total, verify the correctness of each column value. If the value does not conform to a semantic relationship, there is a numerical error.Numerical errors are based on miscalculations.

Assume that a given table has only the following types of errors:
MISS FMT VAL

The format of the given Excel table is as follows:
<Example Table>

The format of the given Excel table is as follows:
"$15,823 "(line name, column number) is (Core commissions and fees, 1)

In the example, $15,823 is belong to row "Core commissions and fees", col_idx is 1

Requirements:

1. Review the first analysis result and the original table
2. Verify each error location (row_name, col_idx) is correct
3. Validate error types (MISS, FMT, VAL) are correctly assigned
4. Count the total number of errors (errNum)
5. Count the number of missing values (missNum)
6. Ensure all error locations are listed in errLoc array
7. Output MUST be in strict JSON format only, no additional text or explanation

Output format (strict JSON only):

```
{
  "errNum": <number>,
  "errLoc": [
    {
      "row_name": "<row name>",
      "col_idx": <column index number>
    }
  ],
  "errType": ["MISS", "FMT", "VAL"],
  "missNum": <number>
}
```

Input Table:
<table_content>

First Analysis Result:
<first_result>

Output (strict JSON format only, no additional text):

Figure 16: Prompt design for logical inconsistency identification (Step 2): structured JSON summarization of detected inconsistencies.

Task definition:
Given the definition of the following error types:
MISS: Empty in the cell value that should have data
FMT: The data in the cell is incorrectly formatted. Common formatting errors are: $$714,925%% and so on.Usually a row or column has the same format.
VAL: Based on the semantic relationship of the table header, for example, Purchase Obligations(1)+Other Obligations(2)+Long term accrued income taxes(3)=Total, verify the correctness of each column value. If the value does not conform to a semantic relationship, there is a numerical error.Numerical errors are based on miscalculations.

Assume that a given table has only the following types of errors:
MISS FMT VAL

The format of the given Excel table is as follows:
<Example Table>

Requirement:
1. Think step by step. Give the reasoning process:
2. Given a table, determine if there are any errors.
3. If there is an error in the table, please output the error type:
4. Count the number of cells that may have the preceding error:
5. Count the number of cells with missing values:
6.Based on the semantic relationship of the table header, for example, Purchase Obligations(1)+Other Obligations(2)+Long term accrued income taxes(3)=Total, verify the correctness of each column value. If the value does not conform to a semantic relationship, there is a VAL error,Verify that values conform to semantic relationships by column:
7. Return the error location based on the row name and column number:

Example analysis: (Investment income,2) There is an error in the format, (Depreciation, 1) there is loss of data, because in the first column: Employee compensation and benefits+Change in acquisition earn-out payables+Other operating expenses+Depreciation
+Amortization+Interest+Non-cash stock-based compensation is not equal to Total expenses, then (Employee compensation and benefits, 1), (Change in acquisition earn-out payables, 1), (Other operating expenses, 1), (Depreciation, 1), (Amortization, 1), (Interest, 1), (Non-cash stock-based compensation, 1), (Total expenses, 1) There is a value error, there is a duplicate Depreciation, and only one errLoc entry is recorded
In the example, $$15,823 is belong to row "Core commissions and fees",col_idx is 1
<Example Table>

Output:
```
{
"errNum": 9,
"errLoc": [
{"row_name": "Depreciation","col_idx": 1},
{"row_name": "Employee compensation and benefits", "col_idx": 1},
{"row_name": "Change in acquisition earn-out payables","col_idx": 1},
…
{"row_name": "Total expenses","col_idx": 1}
],
"errType": ["VAL", "MISS", "FMT"],
"missNum": 1
}
```

Input:
<Query Table>

Output:

Figure 17: Prompt design for logical inconsistency identification with structured JSON output.