

Can LLMs Spot Data Anomalies That Make Table QA Impossible?

Yucheng Wei^{1*}, Shenzhi Yang^{1*}, Ning Wang¹, Jinchang Zhang¹, Yuhang Yang¹,
Haobo Wang^{123†}, Liangyu Zha⁴, Wentao Ye³, Liyao Li³,
Cheng Peng^{1†}, Gang Chen^{3†},

¹School of Software Technology, Zhejiang University

²Hangzhou High-Tech Zone (Binjiang) Institute of Blockchain and Data Security

³State Key Laboratory of Blockchain and Data Security, Zhejiang University

⁴Institute of Computing Innovation, Zhejiang University

{weiyucheng, yangshenzhi, wanghaobo}@zju.edu.cn

Abstract

Existing tabular reasoning benchmarks typically assume that input tables are clean and error-free, which contrasts sharply with real-world data pipelines where noise and anomalies are pervasive. While humans routinely validate tables and reject conclusions when anomalies render a question unanswerable, it remains unclear whether large language models (LLMs) exhibit similar anomaly awareness. In this work, we introduce **AnTBench**, a benchmark designed to systematically evaluate LLM behavior under anomalous tabular data. AnTBench consists of 1,571 curated instances covering five common and verifiable anomaly types. Experiments on 14 representative LLMs reveal a pronounced *capability-behavior gap*: even the strongest model achieves **76.07%** on explicit anomaly detection, yet only **19.03%** on spontaneous anomaly perception during standard table QA. As an inference-time mitigation, we introduce the Multi-Path Thinker, a multi-path reasoning framework. While this approach improves explicit anomaly detection, enforcing proactive verification for every query is practically prohibitive. We urge the community to move beyond inference-time patches and instill spontaneous anomaly awareness directly into the training and alignment phases. Our code and data are available at <https://github.com/Wei12097/AnTBench>.

1 Introduction

Tabular data is one of the most common and fundamental information formats in real-world applications (Borisov et al., 2024; Raffel et al., 2019), and is widely used in critical domains such as finance (Zhang et al., 2024; Islam et al., 2023) and healthcare (Kang et al., 2025; Shi et al., 2024). Recent advances in LLMs (Jaech et al., 2024; Team, 2025a; DeepSeek-AI, 2025) have led to substantial

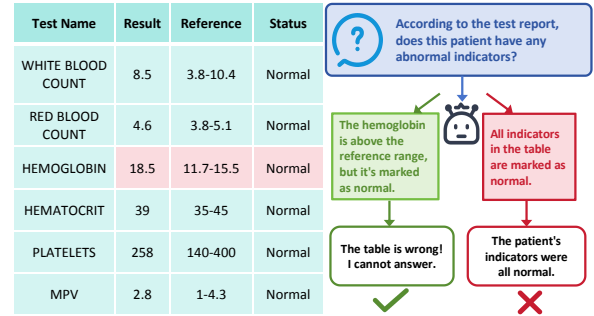


Figure 1: Reasoning over tabular data containing errors.

improvements on tabular reasoning benchmarks, indicating clear progress toward human-level table understanding. However, strong benchmark performance does not necessarily imply that LLMs truly understand tables in a human-like manner. A critical yet often overlooked limitation of existing table reasoning benchmarks is their implicit assumption that input tables are *clean and error-free*. In real-world settings, this assumption rarely holds (Hernández and Stolfo, 1998; Rahm and Do, 2000; Panko, 2016). Tables frequently contain errors introduced through manual data entry, data transformation pipelines, or upstream analytical processes (Panko, 1998).

In practice, humans do not assume a table is inherently correct. Instead, they adopt a cautious reasoning process when interpreting tables. In high-stakes scenarios, humans perform cross-validation, such as checking row-col consistency or comparing related values, before drawing conclusions. If inconsistencies arise, they tend to reject the conclusion and seek further verification rather than blindly trusting the data (Campbell, 2010). A central question naturally follows: *Can LLMs similarly spot data anomalies that make table QA impossible?*

In real-world scenarios, tables may contain anomalies that fundamentally invalidate the question itself. However, it remains unclear whether LLMs can autonomously perform such human-like

*Equal contribution.

†Corresponding authors.

error awareness and validation in reasoning without explicit prompts. When these anomalies go unnoticed, models generate fluent yet fundamentally incorrect answers (Rawte et al., 2023; Huang et al., 2025), leading to serious consequences in high-risk domains such as finance or healthcare (Figure 1). Despite the prevalence and impact of anomalous tables in practice, most existing benchmarks overlook robustness against erroneous tables. Although recent efforts such as RoBuT (Zhao et al., 2023) and RADAR (Gu et al., 2025) explore robustness, they primarily focus on structural variations or explicitly prompted error types. More subtle internal consistency errors—such as calculation or statistical inconsistencies—and the ability of models to proactively recognize unanswerable cases without external guidance remain largely unaddressed.

To this end, we propose **AnTBench**, a novel **Anomalous Table Benchmark** for systematically evaluating the robustness of LLMs when confronted with erroneous or inconsistent tabular data. AnTBench contains 1,571 question-relevant anomaly instances spanning five error categories. It supports two complementary tasks: **Anomaly Perception**, which evaluates whether models can *spontaneously* recognize anomalies during standard table QA, and **Anomaly Detection**, which explicitly instructs models to identify anomalies.

Extensive evaluations on AnTBench reveal a critical and alarming vulnerability: **without explicit reminders of the existence of data anomalies, models fail catastrophically**. The performance is extremely poor across the board: small models like Qwen3-8B completely collapse with scores hovering in the single digits (<5%), and even the most advanced models fail to exceed 20% (GPT-5 achieves only 18.40%). More importantly, this exposes a fundamental “**Capability–Behavior**” gap. Although models can achieve higher performance when explicitly prompted to detect anomalies (GPT-5 achieves 68.33%), this capability does not naturally emerge during standard QA. In other words, some models *can* detect errors, but they *do not choose to do so* unless instructed. Instead, they tend to proceed with reasoning over flawed data and produce confident but incorrect answers.

We further propose Multi-Path Thinker, a multi-path reasoning framework. While it yields consistent gains under explicit detection settings, we emphasize that such methods rely on *external intervention* and do not fundamentally resolve the lack of intrinsic error awareness. In practice, it is infea-

sible to rely on explicit anomaly-checking prompts for every query. Therefore, we argue that anomaly awareness should be treated as a *first-class capability* and be integrated into the training of LLMs, rather than being introduced solely at inference time. We hope this work highlights the urgency of developing models that can *proactively question the validity of their inputs*, especially in real-world settings where data is often imperfect.

2 Related Work

2.1 Table Question Answering Benchmarks

Table Question Answering (Table QA) is a central task for evaluating a model’s ability to understand and reason over structured data. Early benchmarks like WikiTableQuestions (Pasupat and Liang, 2015) and TabFact (Chen et al., 2020) primarily focused on simple fact retrieval. Recent advancements have escalated difficulty by introducing complexity across multiple dimensions, ranging from multi-modal contexts (Foroutan et al., 2025; Zheng et al., 2024; Mathur et al., 2024) and multi-hop reasoning chains (Wu et al., 2025a; Kim et al., 2025; Pal et al., 2023) to intricate hierarchical structures (Wu et al., 2025b; Zhu et al., 2025; Cheng et al., 2022). Despite these advances, a common limitation persists: most existing benchmarks implicitly assume that tables are clean and error-free. In contrast, real-world tables often contain errors. Prior benchmarks rarely evaluate whether models can detect or remain robust to errors, leaving a gap between benchmark performance and real-world reliability.

2.2 Table Robustness Benchmarks for LLMs

Several recent studies have begun to investigate model robustness under table perturbations and degraded data quality. ToRR (Ashury-Tahan et al., 2025) analyzes the impact of table serialization formats and prompt variations on reasoning consistency, while RoBuT (Zhao et al., 2023) introduces adversarial perturbations such as header substitutions, column reordering, and content noise. BBEH (Kazemi et al., 2025) incorporates real-world “dirty data” factors including missing values, duplicates, and structural variations. RADAR (Gu et al., 2025) programmatically injects data artifacts and logical conflicts, but mainly evaluates models under explicit correction-oriented instructions. While informative, these benchmarks largely focus on structural changes or surface-level, easily detectable errors, and often rely on explicit prompts

to guide error handling. More subtle *internal table consistency errors*, such as calculation mistakes, statistical inconsistencies, or violations of knowledge, as well as scenarios in which table errors are inherently non-correctable and render the associated questions unanswerable, remain underexplored. To this end, we introduce AnTBench, a new benchmark designed to evaluate model robustness from a deeper and more realistic perspective.

3 AnTBench

AnTBench is constructed under a principle of *targeted interference*, where carefully designed, problem-relevant anomalies are injected directly into tables so as to fundamentally invalidate the question itself (Figure 2). On this basis, we further design two complementary task paradigms that together enable a comprehensive assessment of model robustness under erroneous tabular inputs.

3.1 Task Definition

We formulate tabular anomaly evaluation as tasks with varying explicitness. Given a table T , a question Q , and an instruction (standard QA $\mathcal{I}_{QA}$ or anomaly detection $\mathcal{I}_{Detect}$), a model $\mathcal{M}$ outputs A .

Task 1: Anomaly Perception. Task 1 evaluates whether a model can *implicitly* perceive anomalous table data during standard table QA, without explicit anomaly detection instructions. Formally, the task is defined as $A = \mathcal{M}(T, Q, \mathcal{I}_{QA})$, where A is either a valid answer or a refusal when anomalies in T make the question unanswerable.

We define two variants to isolate the effect of prompt strictness. **Task 1a (Strict).** The model receives a standard QA prompt with no mention of anomalies; any refusal must arise *spontaneously*, representing the most stringent setting. **Task 1b (Permissive).** The model is additionally informed that anomalies *may* exist and is allowed to handle abnormal cases, but is not instructed to inspect the table or given any anomaly definitions. We adopt **Task 1b as the primary setting**, as it alleviates concerns that performance gains result from minor prompt wording changes while still requiring implicit anomaly perception. Task 1a is retained as a supplementary stress test.

Task 2: Anomaly Detection. Task 2 is an explicit control setting in which the model is instructed to examine the table for anomalies relevant to the question. Formally, it is defined as

$A = \mathcal{M}(T, Q, \mathcal{I}_{Detect}, \mathcal{D}_{err})$, where A is a judgment indicating whether T contains anomalies and $\mathcal{D}_{err}$ is the set of anomaly definitions. Unlike Task 1, this task evaluates the model’s *deliberate* anomaly identification under explicit instruction.

3.2 Anomaly Taxonomy and Definitions

To ensure a rigorous benchmark, AnTBench focuses on anomalies that are *implicit*, *verifiable*, and *prevalent*. Formally, an error in table T with schema $\mathcal{H}$ and cells $\mathcal{V}$ is *verifiable* if it can be identified solely from $(\mathcal{H}, \mathcal{V})$ via deterministic rules. As detailed in Appendix A.3, domain experts reviewed over 400 real-world tables across 23 domains and filtered 15 raw error types. Since explicit anomalies such as missing or bad values are often straightforward to detect, we exclude them from the main tasks and discuss them in Appendix C.1. This process yields five implicit and objectively verifiable anomaly categories, defined below.

-  **Knowledge Conflict:** Arises when the semantic association between two table entries contradicts a stable and canonically verifiable fact. For a pair of linked entries (v_{sub}, v_{obj}) (e.g., Entity “Paris” and Country “UK”) and the schema-specified relation r , let $\mathcal{K}_r$ denote the corresponding canonical factual validity function. A conflict exists if: $\mathcal{K}_r(v_{sub}, v_{obj}) = \text{False}$.
-  **Logical Conflict:** Arises when table entries violate intrinsic rules $\mathcal{R}$. For a tuple of entries (v_i, v_j) and a rule $r \in \mathcal{R}$ (e.g., $Start < End$), a conflict exists if: $r(v_i, v_j) = \text{False}$.
-  **Calculation Error:** Defined as arithmetic inconsistencies where a derived value v_{target} contradicts the result of a deterministic formula f (e.g., sum, ratio) applied to source operands $S \subset \mathcal{V}$. An error occurs if: $v_{target} \neq f(S)$.
-  **Header Name Error:** Represents a semantic misalignment where a header $h_j \in \mathcal{H}$ fails to accurately describe its corresponding data vector $\mathbf{v}_j$. Let $\mathcal{M}$ be a semantic compatibility function; an error is flagged if: $\mathcal{M}(h_j, \mathbf{v}_j) \rightarrow \text{Mismatch}$.
-  **Duplication Error:** Encompasses violations of uniqueness constraints in two forms: (1) *Identifier Duplication*, where distinct rows r_i, r_k share the same primary key ($id(r_i) = id(r_k)$); (2) *Content Duplication*, where distinct entities exhibit identical feature profiles ($\mathcal{V}_{i,:} \equiv \mathcal{V}_{k,:}$).

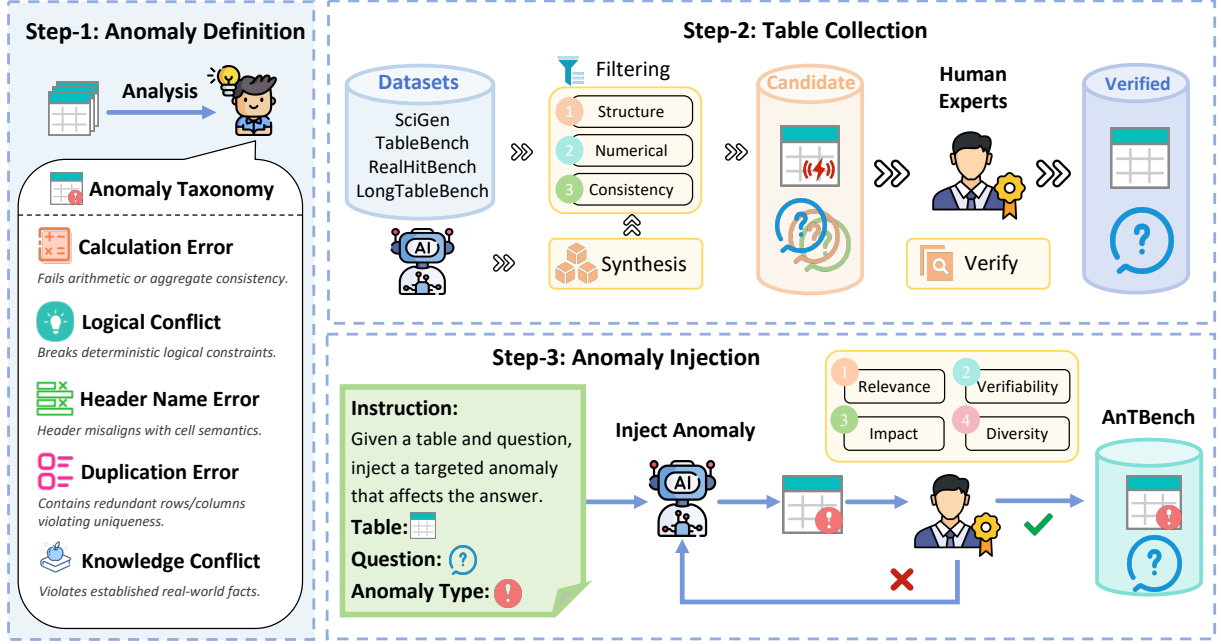


Figure 2: The construction process of AnTBench.

3.3 Anomaly Table Construction

Table Collection. We primarily compile raw tables from widely used datasets, including TableBench (Wu et al., 2025c), RealHitBench (Wu et al., 2025b), LongTableBench (Li et al., 2025), and SciGen (Moosavi et al., 2021). To better support anomaly injection, especially for error types that depend on implicit logical or numerical relationships, we additionally used LLMs to generate a set of tables with strong internal consistency constraints. For all tables, nine domain experts carefully reviewed and validated each table to ensure data correctness, a minimum size of 8 rows and 5 columns, rich numerical content, and verifiable internal consistency, such as summation constraints, ranking relations, or logical rules. Ultimately, we obtained 338 tables spanning diverse domains and a total of 1,571 high-quality questions closely tied to the defined anomaly types.

Anomaly Injection. For the construction of AnTBench, we further injected one of the five previously defined anomaly types into the table associated with each selected question, thereby simulating realistic anomaly scenarios. To avoid interference caused by multiple anomalies within a single table, we created an independent copy of the table for each question and injected only one anomaly type. This design ensures that anomalies introduced for one question do not affect others. To balance annotation quality, diversity, and

efficiency, we adopted a human-machine collaborative workflow combining LLM generation with expert validation. Specifically, each table-question pair was first processed by Gemini-2.5-Pro with carefully designed prompts to generate candidate anomaly-injected tables. We then recruited nine expert annotators with complementary backgrounds in tabular reasoning, data analysis, benchmark evaluation, and industrial LLM systems. Each candidate instance was independently evaluated according to four predefined criteria: question relevance, objective verifiability, answer-critical impact, and plausibility. Instances that failed any criterion, or for which expert consensus could not be reached, were discarded. Detailed annotator backgrounds, validation guidelines, and the exact annotation instructions are provided in Appendix A.4.

3.4 Dataset Statistics

We construct AnTBench, which contains 1,571 injected erroneous instances derived from 338 validated tables, paired with 1,571 questions. The overall statistics are shown in Table 1. The key characteristics of our benchmark are summarized below: (1) All tables originate from high-quality sources or are carefully synthesized, subsequently injected with designed anomalies. To ensure quality, nine experts dedicated approximately 160 hours to data cleaning and 240 hours to anomaly injection. (2) Unlike prior work that concentrates on table QA, our benchmark evaluates a model’s ability to

recognize implicit inconsistencies, which is critical for real world. (3) We define two tasks—Anomaly Perception and Anomaly Detection—to evaluate whether models can detect errors in tables under both natural QA settings and direct diagnostic prompts. Together, AnTBench provides a rigorous and comprehensive benchmark for assessing the robustness of LLMs to erroneous tabular data. More details are provided in Appendix A.

Property	Value
Unique Tables	338
Domains	23
Question Length (Avg.)	17.6
Columns per Table (Avg.)	11.9
Rows per Table (Avg.)	55.7
Ratio of Numerical Cells	54.30%
Ratio of Text Cells	44.10%
Ratio of Date/Time Cells	1.60%

Table 1: Overall statistics of the AnTBench dataset.

4 Can LLMs Spot Data Anomalies?

4.1 Experiment Setup

Baselines. We evaluated a total of 14 advanced models, spanning three categories: **(1) Vanilla models**, including Qwen3s (Yang et al., 2025), Gemma-3s (Team, 2025b) and Llama3s (Team, 2024); **(2) Reasoning-enhanced models**, including GPT-5, Gemini-2.5-Pro (Team, 2025a), DeepSeek-R1 (DeepSeek-AI, 2025) and DeepSeek-V3.1 (DeepSeek-AI, 2024); and **(3) Table-oriented models**, including TableGPT2 (Su et al., 2024), and TableLLM (Wu et al., 2025c).

Metrics and Dataset Configuration. For Task 1 (Anomaly Perception), all evaluations are conducted on an *exclusively erroneous dataset*. To rule out label bias from this setup, we additionally evaluate a mixed dataset in Appendix C.2. Task 1b (Permissive Perception) serves as the *primary experimental setting*. As the expected correct answer is “*Refuse to answer*”, we use **Exact Match (EM)** as the metric. We further validate the reliability of EM with an *LLM-as-a-Judge* (Zheng et al., 2023) protocol, which shows high consistency with EM results (see Appendix C.3). Task 1a (Strict Perception) serves as a supplementary stress test and is evaluated using the *LLM-as-a-Judge* protocol (see Appendix C.4). For Task 2 (Anomaly Detection), we evaluate on a mixture of erroneous and clean tables and report the **F1** score of the Anomaly class.

4.2 Main Results

We present the main experimental results in Table 2, with implementation details in Appendix B.

General Performance Limitations. The results in Table 2 reveal fundamental limitations in the robustness of contemporary LLMs when confronted with erroneous tables. Performance on Anomaly Perception is uniformly poor across all models. Most models fail to exceed **15%**, and even the state-of-the-art Gemini-2.5-pro only achieves **19.03%**. This indicates models rarely suppress answer generation despite critical inconsistencies in the table. This weakness is further amplified under stricter prompting conditions. As shown in Table 3, when cornered by strict instructions without the implicit allowance to refuse, models exhibit markedly lower resistance to anomalous data and a stronger tendency to hallucinate answers rather than withhold a response, underscoring the fragility of their anomaly awareness. We also observe a stratification by model scale. Small-scale models (< 32B), perform poorly on both tasks, with single-digit scores on Perception and typically below 20% on Detection. Although frontier models outperform smaller ones, their performance remains insufficient for high-stakes tabular reasoning scenarios.

General Performance Limitations

Current LLMs exhibit fundamental vulnerability to tabular errors and fail to spontaneously perceive data inconsistencies.

The “Capability-Behavior” Discrepancy. An intriguing discrepancy exists between Anomaly Perception and Anomaly Detection performance. Several models exhibit a relatively decent capability in explicit detection but collapse in implicit perception. For example, Gemini-2.5-pro achieves an impressive **76.07%** on Detection task, but its Perception task score drops sharply to **19.03%**. This gap highlights a critical “capability-behavior” mismatch. While models possess the *latent capability* to recognize tabular anomalies when explicitly prompted to verify data, they fail to *spontaneously* trigger this critical thinking process during standard QA. Instead, models default to answering the question as posed, implicitly assuming the table to be correct. Such behavior reflects a broader alignment issue: current LLMs prioritize answer fluency over premise validation, leading to confident but invalid outputs when the input data itself is flawed.

Model	Task1: Anomaly Perception (EM)						Task2: Anomaly Detection (F1@Anomaly)					
	Calc.	Knowl.	Log.	Header	Dup.	Overall	Calc.	Knowl.	Log.	Header	Dup.	Overall
Vanilla Models												
Qwen3-8B	1.41	0.00	3.37	0.00	2.75	1.53	21.01	4.88	15.38	17.39	10.69	19.61
Qwen3-14B	1.03	0.00	1.45	2.08	1.01	1.06	19.56	13.95	10.10	29.03	0.00	18.05
Qwen3-32B	3.67	0.00	6.74	7.69	7.34	4.14	21.12	4.88	12.12	20.34	6.90	19.33
Llama3.1-8B-Instruct	0.55	0.00	1.12	1.92	0.00	0.57	14.69	13.11	15.07	13.95	5.75	14.07
Gemma-3-12b-it	1.09	0.00	1.12	3.85	2.75	1.27	23.91	4.88	21.82	39.47	9.52	23.09
Gemma-3-27b-it	3.04	0.00	2.25	5.77	3.67	3.06	26.83	18.18	23.42	47.73	17.39	26.72
Qwen3-Max	9.76	27.50	16.85	30.77	19.27	11.97	57.19	80.60	41.67	50.00	17.60	54.77
Table-oriented Models												
TableGPT2-7B	2.03	0.00	2.25	0.00	1.83	1.91	12.56	4.44	15.13	6.35	4.69	11.83
TableLLM-Qwen2-7B	0.47	0.00	0.18	1.17	0.84	0.49	3.08	0.00	1.17	13.08	3.69	3.28
TableLLM-Llama3.1-8B	0.33	0.00	0.16	0.77	0.11	0.32	0.08	0.00	1.12	0.00	0.00	0.13
Reasoning Models												
Qwen3-8B-thinking	3.12	2.50	5.62	3.85	6.42	3.50	47.55	17.39	31.01	50.94	26.25	45.07
Qwen3-14B-thinking	6.25	7.50	6.74	9.62	11.01	6.75	55.89	43.14	29.06	59.79	30.07	53.06
Qwen3-32B-thinking	6.71	2.50	8.99	19.23	17.43	7.89	51.87	36.73	31.03	54.17	24.64	49.17
GPT-5	15.53	52.50	24.77	34.81	26.61	18.40	71.54	85.71	53.44	59.79	28.57	68.33
Gemini-2.5-pro	16.55	42.50	19.26	42.52	28.44	19.03	77.59	91.89	65.06	66.07	64.15	76.07
DeepSeek-v3.1	10.38	17.50	11.41	25.09	22.94	11.97	56.37	59.65	38.02	58.43	32.65	54.19
DeepSeek-r1	10.85	30.00	14.61	26.92	17.43	12.54	57.83	78.79	46.88	55.42	27.97	56.24
Models with Multi-Path Thinker												
Qwen3-8B	26.00	25.00	20.22	28.85	14.68	24.95	42.07	36.36	37.04	41.86	26.67	40.75
Qwen3-8B-thinking	41.42	15.00	20.22	36.71	22.09	38.05	53.16	63.64	51.95	61.26	36.05	52.61
GPT-5	67.53	62.50	41.57	39.45	32.69	62.83	79.24	87.32	67.90	61.70	48.10	76.54

Table 2: Results on AnTBench. Task 1 is evaluated on erroneous tables only using EM (%). Task 2 is evaluated on both erroneous and clean tables using F1@Anomaly (%). Abbreviations: **Calc.** (Calculation Error), **Knowl.** (Knowledge Conflict), **Log.** (Logical Conflict), **Header** (Header Name Error), **Dup.** (Duplication Error).

Model	Perception Score		
	Permissive	Strict	Perf. Drop
GPT-5	18.40	12.28	6.11
Gemini-2.5-pro	19.03	6.68	12.36
DeepSeek-r1	12.54	6.43	6.11
Qwen3-Max	11.97	7.90	4.07

Table 3: Impact of prompt constraints on anomaly perception. **Permissive**: Permissive Perception (Metric: EM). **Strict**: Strict Perception, (Metric: LLM-Judge).

The “Capability-Behavior” Discrepancy

LLMs possess latent error-detection capabilities, but fail to proactively activate them during normal reasoning.

The Impact of Reasoning Mechanisms. Comparisons between vanilla models and their reasoning-augmented counterparts demonstrate that reasoning mechanisms substantially improve robustness to tabular anomalies. Models with a thinking mode consistently outperform their vanilla versions, particularly on Anomaly Detection. For

example, Qwen3-8B-thinking achieves 45.07%, more than doubling Qwen3-8B (19.61%), while Qwen3-14B-thinking improves from 18.05% to 53.06%. Notably, these gains are significantly larger than those obtained by merely increasing model size within the vanilla setting. A larger vanilla model such as Qwen3-32B reaches only 19.33%, far below the smaller but reasoning-enabled Qwen3-8B-thinking(45.07%). This contrast suggests that for logic-intensive tasks such as tabular anomaly detection, *compute depth* (reasoning steps) is more critical than *parameter count*. The thinking mechanism likely serves as an explicit verification stage, enabling arithmetic cross-checks and logical consistency analysis that are difficult to realize in a vanilla model. Nevertheless, improvements on Anomaly Perception remain limited, indicating that reasoning alone is insufficient to guarantee spontaneous error awareness.

Impact of Thinking

Reasoning capability outweighs parameter scale for logic-intensive anomaly detection.

Other Analyses. We conduct several additional analyses to better understand both the benchmark and model behavior. First, we introduce a rule-based baseline to examine whether AnTBench can be trivially solved by generic data-cleaning heuristics. Specifically, we construct an LLM-assisted heuristic checker that generates executable rules for mathematical consistency, uniqueness constraints, schema-content alignment, and temporal or logical relations. On a balanced subset of 200 instances (40 per anomaly category), this baseline achieves an overall F1 score of approximately 50%, with category-level performance ranging from 28.57% to 58.33%. These results suggest that generic rule-based heuristics cannot reliably identify the question-relevant anomalies in AnTBench, as many cases require determining which table-specific relations or constraints are relevant to the associated question. We leave the complete implementation details, category-level results, and efficiency analysis to Appendix C.5. We then analyze model outputs to investigate why LLMs struggle to identify anomalies. We observe a prevalent pattern: models often perform shallow retrieval based on the question and directly output the extracted content, bypassing secondary verification of its logical correctness. Detailed case studies are provided in Appendix E. Furthermore, we explore how model performance correlates with table size and anomaly density. Results indicate that performance generally decreases as table size increases, whereas more frequent anomalies tend to improve detection performance. Extended analyses are provided in Appendix C.6 and Appendix C.7. Finally, to ensure the generality of our findings, we expand our evaluation to recently released frontier models. The results remain highly consistent with our primary conclusions; see Appendix C.8.

Discussion and Implications. Our findings reveal a deployment dilemma: although current models possess latent error-detection capabilities, they rarely activate them spontaneously during standard table QA. Explicit verification prompts can partially elicit this capability, but applying computationally intensive verification to every query would incur substantial additional inference cost and latency and is therefore not a scalable substitute for intrinsic anomaly awareness. In the long term, robust table reasoning requires anomaly awareness to be incorporated into pre-training or alignment; our preliminary training-time experiment in Ap-

pendix C.9 provides initial evidence in this direction. Before such training-based solutions become mature, however, selective inference-time safeguards may still be useful in high-risk applications where additional computation is acceptable. We therefore turn next to the Detection task and introduce an inference-time reasoning framework as a deliberately computation-intensive diagnostic and temporary mitigation.

5 Can Diverse Reasoning Enhance Anomaly Detection?

5.1 Multi-Path Thinker

As illustrated in Figure 3, unlike conventional approaches that rely on a single chain of thought, our framework steers the model to synthesize multiple solution paths for a given query. By enforcing a cross-verification mechanism among these independent trajectories, the framework identifies discrepancies in the outcomes and subsequently executes a source tracing process, serving as a rigorous filter for detecting latent anomalies in tabular data.

5.1.1 Multi-Path Generation

The first stage, Multi-Path Generation, is designed to mitigate the limitations of single-chain reasoning by encouraging the model to analyze the problem from multiple, complementary perspectives. Formally, given an input query Q and a table context T , the model is prompted to construct a set of k distinct reasoning plans:

$$\mathcal{P} = \{P_1, P_2, \dots, P_k\},$$

where each P_i corresponds to a unique strategic viewpoint for solving the task.

To promote genuine diversity rather than superficial paraphrasing, we explicitly encourage the model to examine the table from fundamentally different logical angles, enabling more comprehensive coverage of potential failure modes. Such diversity is critical for exposing inconsistencies that may remain undetected under a single reasoning strategy.

5.1.2 Multi-Path Reasoning

After generating the reasoning plans, the framework proceeds to the execution phase. In a *single model interaction*, the model is instructed to follow each reasoning plan P_i sequentially and produce a corresponding answer a_i :

$$\begin{aligned} a_i &= \text{Reason}(Q, T, P_i \mid \mathcal{C}), \\ \mathcal{A} &= \{a_1, a_2, \dots, a_k\}. \end{aligned}$$

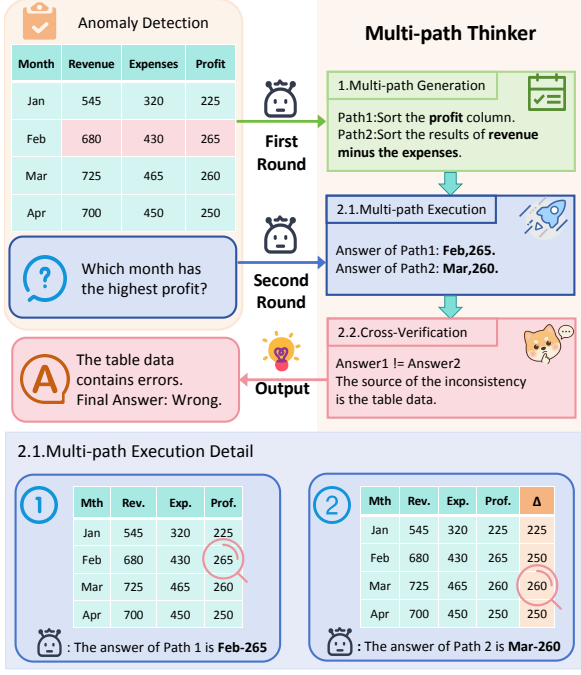


Figure 3: The overall framework of Multi-Path Thinker.

where $\mathcal{C}$ denotes the shared conversational context for all reasoning paths and $\mathcal{A}$ represents the set of candidate answers. This formulation ensures that all candidate answers are generated under identical contextual conditions, eliminating variance introduced by separate model calls.

Applied to *Anomaly Detection*, the framework then cross-verifies multiple answers. Since valid paths on a consistent table should converge, divergence signals potential conflicts. However, mere disagreement is inconclusive; the model must subsequently *trace the source of inconsistency*. It distinguishes between intrinsic table errors (e.g., constraint violations) and external noise (e.g., stochasticity), reporting an anomaly only if the conflict stems directly from the data. Formally, let S denote the diagnosed source of inconsistency. The final decision function is defined as

$$D(\mathcal{A}) = \begin{cases} \text{"Wrong"}, & |\mathcal{A}| > 1 \wedge S = T, \\ \text{"Correct"}, & \text{otherwise.} \end{cases}$$

By combining multi-path consistency checking with source-aware diagnosis, this framework enables effective detection of table anomalies.

5.2 Experiment Results

Effectiveness of Multi-Path Thinker. As shown in Table 2, our Multi-Path Thinker substantially improves Anomaly Detection across various models (Qwen3-8B: 19.61% $\rightarrow$ 40.75%; GPT-5: 68.33%

Model Name	Method	F1 (%)
Qwen3-8B	Vanilla (CoT)	19.61
	Self-Refine	28.19
	Ours (Multi-Path)	40.75
Qwen3-8B-thinking	Vanilla (CoT)	45.07
	Self-Refine	40.72
	Ours (Multi-Path)	52.61

Table 4: Performance comparison of CoT, Self-Refine, and our Multi-Path Thinker on Anomaly Detection.

$\rightarrow$ 76.54%) by introducing diversified reasoning trajectories. To further validate this mechanism, we compare it with multi-turn Self-Refine (SR) (Madaan et al., 2023), which yields limited gains and can even degrade CoT performance (Table 4). In contrast, our method independently generates and aggregates multiple reasoning paths, enforcing cross-verification across alternative perspectives and thereby mitigating single-path bias more effectively. We also evaluate a category-agnostic variant without explicit anomaly definitions, which remains effective and suggests that the gains do not simply come from category-specific prompt tailoring; see Appendix C.10.

Effectiveness of Multi-Path Thinker

Diversity in reasoning paths significantly enhances tabular anomaly detection.

Components of Thinker	Avg Score	Delta
All (Full Method)	40.75	0.00
w/o Multi-Path Generation	28.44	-12.31
w/o Path-based Reasoning	28.98	-11.77

Table 5: Ablation study of the Multi-Path Thinker.

Each Multi-Path Thinker component enhances the model’s effectiveness. As shown in Table 5, we conducted an ablation study on the Qwen3-8B model to evaluate individual components of the Multi-Path Thinker. The results demonstrate that the *Multi-Path Generation* module is pivotal, with its removal causing the sharpest performance drop of 12.31 points. This indicates that generating diverse reasoning candidates is the fundamental driver of error perception. Closely following this, the exclusion of *Path-based Reasoning* leads to a comparable decline of 11.77 points. The narrow gap between these two settings suggests that the subsequent structured reasoning and cross-verification mechanism is nearly equally critical for

ensuring the final accuracy of anomaly detection.

Remark. While the Multi-Path Thinker is highly effective, it is strictly intended for the *Anomaly Detection* task and deliberately excluded from the *Anomaly Perception* task. As shown in the gray-shaded cells of Table 2, applying this method to Perception Task yields massive improvements (GPT-5: 18.40% $\rightarrow$ 62.83%). However, we exclude these results from the official evaluation to preserve construct validity. Task 1 is specifically designed to measure a model’s *intrinsic* capability to spontaneously perceive anomalies. Introducing external reasoning framework would fundamentally alter this objective, shifting evaluation from native “data awareness” to prompt engineering efficacy.

6 Conclusions

We propose a novel benchmark designed to systematically evaluate LLMs’ ability to perceive and detect anomalies in tabular data. Experiments on 14 representative LLMs reveal that current models still face substantial challenges in identifying table errors, which significantly undermines the reliability of LLMs in real-world tabular question-answering scenarios. We hope this work will draw increased attention to the often-overlooked issue of tabular data anomalies and foster the development of more reliable and trustworthy LLMs.

Limitations

We acknowledge several limitations of this work. (1) Although the current dataset construction process ensures high annotation quality through rigorous manual verification, it has not yet achieved full automation. This inherently limits the scale of the dataset, making the transition to a completely automated pipeline an important direction for future research. (2) While our proposed Multi-Path Thinker substantially improves explicit Anomaly Detection, **such inference-time algorithmic interventions are essentially post-hoc remedies**. In real-world deployments, it is infeasible to rely on explicit anomaly-checking prompts for every query just to guard against potential data errors. Therefore, our method serves as a temporary mitigation rather than a structural cure. The overarching limitation—that models inherently lack a spontaneous “skepticism” toward input data—remains unresolved. We reiterate that true robustness must ultimately be cultivated during the pre-training or alignment phases, embedding anomaly awareness

as an intrinsic model capability rather than patching it through external prompting.

Ethical Considerations

AnTBench is a public benchmark for studying LLM robustness in tabular reasoning. All source tables are collected from publicly available datasets or carefully synthesized, and no private user data or non-public information is used. We manually reviewed all tables and questions to remove sensitive personal information or inappropriate content. All data curation and verification were conducted internally by nine authors acting as expert annotators; thus, no external crowdsourcing or monetary compensation was involved. Experts followed strict guidelines, and instances failing any criterion or lacking expert consensus were discarded. Detailed annotator backgrounds, validation statistics, and exact annotation instructions are provided in Appendix A.4. As a result, the benchmark is clean and poses no ethical concerns. During the preparation of this paper, LLMs were used only for limited auxiliary purposes, such as text polishing and table anomaly injection.

Acknowledgments

This paper is supported by the National Regional Innovation and Development Joint Fund (No. U24A20254) and partially supported by ZJU Kunpeng & Ascend Center of Excellence.

References

- Shir Ashury-Tahan, Yifan Mai, Rajmohan C, Ariel Gera, Yotam Perlitz, Asaf Yehudai, Elron Bandel, Leshem Choshen, Eyal Shnarch, Percy Liang, and Michal Shmueli-Scheuer. 2025. [The mighty torr: A benchmark for table reasoning and robustness](#). *CoRR*, abs/2502.19412.
- Vadim Borisov, Tobias Leemann, Kathrin Seßler, Johannes Haug, Martin Pawelczyk, and Gjergji Kasneci. 2024. [Deep neural networks and tabular data: A survey](#). *IEEE Trans. Neural Networks Learn. Syst.*, 35(6):7499–7519.
- Mary Pat Campbell. 2010. Spreadsheet issues: pitfalls, best practices, and practical tips. In *Actuarial Practice Forum*, pages 1–39.
- Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyu Zhou, and William Yang Wang. 2020. [Tabfact: A large-scale dataset for table-based fact verification](#). In *8th International Conference on Learning Representations, ICLR 2020*. OpenReview.net.

- 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 *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, ACL 2022, pages 1094–1110. Association for Computational Linguistics.
- DeepSeek-AI. 2024. [Deepseek-v3 technical report](#). *CoRR*, abs/2412.19437.
- DeepSeek-AI. 2025. [Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning](#). *CoRR*, abs/2501.12948.
- Negar Foroutan, Angelika Romanou, Matin Ansari-pour, Julian Martin Eisenschlos, Karl Aberer, and Rémi Lebret. 2025. [Wikimixqa: A multimodal benchmark for question answering over tables and charts](#). In *Findings of the Association for Computational Linguistics*, ACL 2025, pages 24941–24958. Association for Computational Linguistics.
- Ken Gu, Zhihan Zhang, Kate Lin, Yuwei Zhang, Akshay Paruchuri, Hong Yu, Mehran Kazemi, Kumar Ayush, A. Ali Heydari, Maxwell A. Xu, Girish Narayanswamy, Yun Liu, Ming-Zher Poh, Yuzhe Yang, Mark Malhotra, Shwetak N. Patel, Hamid Palangi, Xuhai Xu, Daniel McDuff, and 2 others. 2025. [RADAR: benchmarking language models on imperfect tabular data](#). *CoRR*, abs/2506.08249.
- Mauricio A. Hernández and Salvatore J. Stolfo. 1998. [Real-world data is dirty: Data cleansing and the merge/purge problem](#). *Data Min. Knowl. Discov.*, 2(1):9–37.
- Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2025. [A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions](#). *ACM Trans. Inf. Syst.*, 43(2):42:1–42:55.
- Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, Aleksander Madry, Alex Baker-Whitcomb, Alex Beutel, Alex Borzunov, Alex Carney, Alex Chow, Alex Kirillov, Alex Nichol, Alex Paino, and 79 others. 2024. [Gpt-4o system card](#). *CoRR*, abs/2410.21276.
- Pranab Islam, Anand Kannappan, Douwe Kiela, Rebecca Qian, Nino Scherrer, and Bertie Vidgen. 2023. [Financebench: A new benchmark for financial question answering](#). *CoRR*, abs/2311.11944.
- Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Hel-yar, Aleksander Madry, Alex Beutel, Alex Carney, Alex Iftimie, Alex Karpenko, Alex Tachard Passos, Alexander Neitz, Alexander Prokofiev, Alexander Wei, Allison Tam, Ally Bennett, Ananya Kumar, and 80 others. 2024. [Openai o1 system card](#). *CoRR*, abs/2412.16720.
- Ha Ye Jin Kang, Minsam Ko, and Kwang Sun Ryu. 2025. [Tabular transformer generative adversarial network for heterogeneous distribution in healthcare](#). *Scientific Reports*, 15(1):10254.
- Mehran Kazemi, Bahare Fatemi, Hritik Bansal, John Palowitch, Chrysovalantis Anastasiou, Sanket Vaibhav Mehta, Lalit K. Jain, Virginia Aglietti, Disha Jindal, Peter Chen, Nishanth Dikkala, Gladys Tyen, Xin Liu, Uri Shalit, Silvia Chiappa, Kate Olszewska, Yi Tay, Vinh Q. Tran, Quoc V. Le, and Orhan Firat. 2025. [Big-bench extra hard](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, ACL 2025, pages 26473–26501. Association for Computational Linguistics.
- Yunsoo Kim, Yusuf Abdulle, and Honghan Wu. 2025. [Biohopr: A benchmark for multi-hop, multi-answer reasoning in biomedical domain](#). In *Findings of the Association for Computational Linguistics*, ACL 2025, pages 12894–12908. Association for Computational Linguistics.
- Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. [Efficient memory management for large language model serving with pagedattention](#). In *Proceedings of the 29th Symposium on Operating Systems Principles, SOSP 2023*, pages 611–626. ACM.
- Liyao Li, Jiaming Tian, Hao Chen, Wentao Ye, Chao Ye, Haobo Wang, Ningtao Wang, Xing Fu, Gang Chen, and Junbo Zhao. 2025. [LongTableBench: Benchmarking long-context table reasoning across real-world formats and domains](#). In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 11927–11965. Association for Computational Linguistics.
- Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegrefe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. [Self-refine: Iterative refinement with self-feedback](#). In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023*.
- Suyash Vardhan Mathur, Jainit Sushil Bafna, Kunal Kartik, Harshita Khandelwal, Manish Shrivastava, Vivek Gupta, Mohit Bansal, and Dan Roth. 2024. [Knowledge-aware reasoning over multimodal semi-structured tables](#). In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 14054–14073. Association for Computational Linguistics.
- Nafise Sadat Moosavi, Andreas Rücklé, Dan Roth, and Iryna Gurevych. 2021. [Scigen: a dataset for](#)

- reasoning-aware text generation from scientific tables. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021*.
- 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), ACL 2023*, pages 6322–6334. Association for Computational Linguistics.
- Raymond R Panko. 1998. What we know about spreadsheet errors. *Journal of Organizational and End User Computing (JOEUC)*, 10(2):15–21.
- Raymond R. Panko. 2016. What we don’t know about spreadsheet errors today: The facts, why we don’t believe them, and what we need to do. *CoRR*, abs/1602.02601.
- 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 of the Asian Federation of Natural Language Processing, ACL 2015*, pages 1470–1480. The Association for Computer Linguistics.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *CoRR*, abs/1910.10683.
- Erhard Rahm and Hong Hai Do. 2000. Data cleaning: Problems and current approaches. *IEEE Data Eng. Bull.*, 23(4):3–13.
- Vipula Rawte, Swagata Chakraborty, Agnibh Pathak, Anubhav Sarkar, S. M. Towhidul Islam Tonmoy, Aman Chadha, Amit P. Sheth, and Amitava Das. 2023. The troubling emergence of hallucination in large language models - an extensive definition, quantification, and prescriptive remediations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023*, pages 2541–2573. Association for Computational Linguistics.
- Wenqi Shi, Ran Xu, Yuchen Zhuang, Yue Yu, Jieyu Zhang, Hang Wu, Yuanda Zhu, Joyce C. Ho, Carl Yang, and May Dongmei Wang. 2024. Ehragent: Code empowers large language models for few-shot complex tabular reasoning on electronic health records. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024*, pages 22315–22339. Association for Computational Linguistics.
- Aofeng Su, Aowen Wang, Chao Ye, Chen Zhou, Ga Zhang, Gang Chen, Guangcheng Zhu, Haobo Wang, Haokai Xu, Hao Chen, Haoze Li, Haoxuan Lan, Jiaming Tian, Jing Yuan, Junbo Zhao, Junlin Zhou, Kaizhe Shou, Liangyu Zha, Lin Long, and 14 others. 2024. Tablegpt2: A large multimodal model with tabular data integration. *CoRR*, abs/2411.02059.
- Gemini Team. 2025a. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *CoRR*, abs/2507.06261.
- Gemma Team. 2025b. Gemma 3 technical report. *CoRR*, abs/2503.19786.
- Llama Team. 2024. The llama 3 herd of models. *CoRR*, abs/2407.21783.
- 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 *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022*.
- Jian Wu, Linyi Yang, Dongyuan Li, Yuliang Ji, Manabu Okumura, and Yue Zhang. 2025a. MMQA: evaluating llms with multi-table multi-hop complex questions. In *The Thirteenth International Conference on Learning Representations, ICLR 2025*. OpenReview.net.
- Pengzuo Wu, Yuhang Yang, Guangcheng Zhu, Chao Ye, Hong Gu, Xu Lu, Ruixuan Xiao, Bowen Bao, Yijing He, Liangyu Zha, Wentao Ye, Junbo Zhao, and Haobo Wang. 2025b. Realhitbench: A comprehensive realistic hierarchical table benchmark for evaluating llm-based table analysis. In *Findings of the Association for Computational Linguistics, ACL 2025*, pages 7105–7137. Association for Computational Linguistics.
- Xianjie Wu, Jian Yang, Linzheng Chai, Ge Zhang, Jiaheng Liu, Xeron Du, Di Liang, Daixin Shu, Xianfu Cheng, Tianzhen Sun, Tongliang Li, Zhoujun Li, and Guanglin Niu. 2025c. Tablebench: A comprehensive and complex benchmark for table question answering. In *AAAI-25, Sponsored by the Association for the Advancement of Artificial Intelligence*, pages 25497–25506. AAAI Press.
- An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 40 others. 2025. Qwen3 technical report. *CoRR*, abs/2505.09388.
- Chao Zhang, Yuren Mao, Yijiang Fan, Yu Mi, Yunjun Gao, Lu Chen, Dongfang Lou, and Jinshu Lin. 2024. Finsql: Model-agnostic llms-based text-to-sql framework for financial analysis. In *Companion of the 2024 International Conference on Management of Data, SIGMOD/PODS 2024*, pages 93–105. ACM.
- Yilun Zhao, Chen Zhao, Linyong Nan, Zhenting Qi, Wenlin Zhang, Xiangru Tang, Boyu Mi, and

Dragomir Radev. 2023. [Robut: A systematic study of table QA robustness against human-annotated adversarial perturbations](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, ACL 2023, pages 6064–6081. Association for Computational Linguistics.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhonghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. [Judging llm-as-a-judge with mt-bench and chatbot arena](#). In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023*.

Mingyu Zheng, Xinwei Feng, Qingyi Si, Qiaoqiao She, Zheng Lin, Wenbin Jiang, and Weiping Wang. 2024. [Multimodal table understanding](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, ACL 2024, pages 9102–9124. Association for Computational Linguistics.

Junnan Zhu, Jingyi Wang, Bohan Yu, Xiaoyu Wu, Junbo Li, Lei Wang, and Nan Xu. 2025. [Tableeval: A real-world benchmark for complex, multilingual, and multi-structured table question answering](#). *CoRR*, abs/2506.03949.

A More Dataset Details

In this section, we provide further details on our dataset. First, we collect raw tables from widely used datasets. Then, to ensure high quality, we perform a rigorous filtering process based on structural complexity, numerical content richness, and internal consistency relationships. The final curated dataset consists of 338 tables spanning 23 diverse domains.

A.1 Data Source

The tables in our dataset originate from a hybrid mix of sources to ensure both domain diversity. The majority of the data is selected from four widely used high-quality benchmarks: TableBench (Wu et al., 2025c), RealHitBench (Wu et al., 2025b), LongTableBench (Li et al., 2025), and SciGen (Moosavi et al., 2021). Additionally, to ensure comprehensive coverage of specific anomaly categories, we supplemented these sources with a small subset of tables synthesized by LLMs. We thoroughly examined all sources to guarantee the authority and rationality of the data. The details of each data source are introduced below.

TableBench. TableBench is a comprehensive and complex benchmark designed to facilitate a wide range of TableQA tasks. It encompasses diverse real-world scenarios, covering 20 major domains such as finance, sports, and science. The tables are meticulously selected from existing high-quality datasets to ensure structural complexity and numerical depth. The data is highly reliable, as it undergoes a rigorous annotation workflow involving both automated methods and human verification to ensure the accuracy of the questions and answers.

RealHitBench. RealHitBench is a comprehensive and realistic benchmark designed to evaluate table analysis capabilities in complex real-world scenarios. The dataset comprises raw tables sourced from 13 reputable open platforms, covering 24 diverse domains such as economy, society, and science. Unlike traditional datasets, it focuses on intricate table structures, including hierarchical headers and nested sub-tables. The data is considered of high quality and reliability, having undergone a rigorous construction process that includes multi-turn human verification to ensure the rationality and authority of the content.

LongTableBench. LongTableBench is a large-scale benchmark specifically designed for evaluating long-context reasoning over semi-structured tables. It comprises a diverse collection of data spanning 7 distinct formats (e.g., Markdown, HTML, SQL) and 18 domains, capable of handling input lengths up to 128K tokens in multi-turn and multi-table scenarios. The data is of high quality and reliability, derived from established datasets (BIRD, Spider, and Dr.Spider) and refined through a rigorous multi-stage verification process that includes symbolic SQL execution checks, LLM-based cross-validation, and manual human review to ensure logical correctness.

SciGen. SciGen is a challenging dataset designed for the task of reasoning-aware data-to-text generation. It consists of complex tables and corresponding descriptions extracted from scientific articles, primarily covering Computer Science domains such as Computation and Language and Machine Learning. Unlike standard datasets, SciGen focuses on tables with numerical values that require arithmetic reasoning (e.g., calculation and comparison) to describe. The data is of high quality and reliability, as the core test sets were manually annotated and verified by domain experts, including

PhD students and postdocs, to ensure the accuracy of the reasoning involved.

LLM-Synthesized Data. While existing datasets provide a strong foundation, naturally occurring tables often lack the specific patterns required to test certain complex error types. To address this scarcity, we employed advanced LLMs to synthesize a targeted subset of tables. For each generated table, we concurrently generated corresponding questions to ensure strict semantic alignment between the table content and the query. Crucially, these synthetic instances underwent the same rigorous manual validation process as the collected data to ensure they meet our high standards for logical consistency and realism.

A.2 Data Domains

Our dataset covers 23 distinct domains. As detailed in Figure 4, the Economy domain constitutes the largest share (21.60%), followed by Society (18.48%). Labor-market related data also represents a significant portion of the collection, with Labor (10.48%) and Employment (9.99%) collectively accounting for approximately 20% of the total files.

The remaining 40% of the dataset covers a diverse range of 19 other fields, including Science (5.64%), Environment (4.14%), and Education (3.82%), providing a broad spectrum of tabular data for analysis. Specialized domains such as Energy, Productivity, Geography and Transport appear as long-tail categories with fewer than 0.5% each.

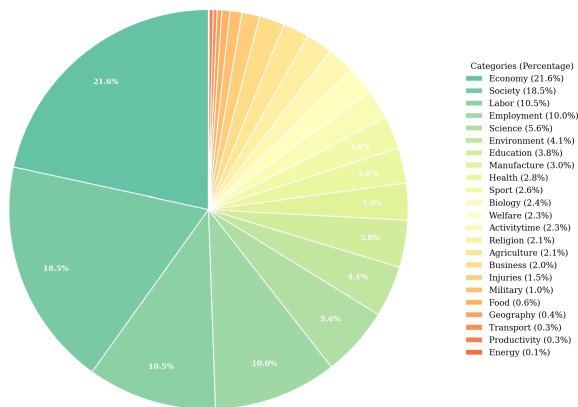


Figure 4: All domains of tables in AnTBench.

A.3 Empirical Taxonomy Construction Process

To ensure that AnTBench accurately reflects the complexities and fatal flaws present in real-world

data pipelines, our anomaly taxonomy was not predefined theoretically. Instead, it was distilled through a rigorous, multi-stage empirical analysis conducted by domain experts. The construction process proceeded in three main stages:

Stage 1: Exploratory Review and Open Coding.

Prior to benchmark construction, nine domain experts (the authors) conducted an exhaustive manual review of approximately 400 raw tables drawn from 23 diverse domains. During a “back-to-back” independent annotation phase, the experts recorded every instance of data corruption or logical inconsistency they naturally encountered. This open-coding phase yielded 15 raw error descriptions, including missing values, formatting artifacts, spelling typos, broken hyperlinks, conflicting date ranges, and summation mismatches.

Stage 2: The “Reasoning-Necessity” Filter.

In the second iteration, we systematically grouped and filtered these raw errors. We deliberately discarded explicit, surface-level noise (e.g., missing values, formatting garbage, inconsistent units). The rationale was that these errors can be perfectly identified by traditional data parsers, simple pattern-matching algorithms, or handled via the conventional “repair and proceed” paradigm. Consequently, we retained only *implicit* errors—flaws that appear structurally pristine but harbor deep mathematical or logical contradictions that require advanced reasoning to perceive.

Stage 3: The “Deterministic Verifiability” Principle.

In the final iteration, the remaining implicit error types were consolidated into the five final categories based on the strict principle of *Deterministic Verifiability*. To be included in AnTBench, an error could not rely on subjective domain expertise, probabilistic outliers, or external commonsense assumptions; it had to be absolutely provable using only the table’s internal schema and cell values. For example, scattered mathematical discrepancies were grouped into **Calculation Errors** because they definitively violate absolute arithmetic laws. This rigorous filtering process ensures that every category in AnTBench represents a mathematically or logically fatal flaw, guaranteeing that the ground-truth labels are objective, absolute, and fundamentally unanswerable.

A.4 Human Validation Protocol

To ensure the reliability of AnTBench, all candidate anomaly instances underwent manual validation by domain experts. We did not use external crowdsourcing platforms, since verifying implicit tabular anomalies requires careful reasoning over mathematical, logical, semantic, and factual constraints. Instead, the validation was conducted internally by the authors acting as domain experts. As this was an internal research effort, no external monetary compensation was involved.

Annotator Backgrounds. Human validation involved nine researchers with computer science backgrounds and complementary expertise in benchmark construction, academic tabular reasoning, industrial deployment, and data evaluation. Three were core authors who extensively reviewed the literature on tabular reasoning, data quality, and data cleaning, designed the anomaly taxonomy and validation protocol, and supervised the full construction of AnTBench. The other six experts provided complementary expertise. Three were academic researchers specializing in tabular understanding and reasoning, each having authored at least three peer-reviewed papers specifically on tabular understanding, table question answering, or tabular reasoning at leading venues such as ACL, EMNLP, ICLR, and NeurIPS. Two had experience developing and deploying enterprise-scale tabular systems. The remaining expert specialized in statistical data analysis and benchmark evaluation, contributing expertise in objective verification, experimental design, and evaluation methodology.

Validation Procedure. Each candidate instance consisted of five components: an instance ID, a clean table, a modified table containing one injected anomaly, the associated question, and the target anomaly type. Experts independently evaluated candidate instances using a predefined four-step protocol covering (1) question relevance, (2) objective verifiability, (3) answer-critical impact, and (4) plausibility. The criteria were applied sequentially using a short-circuit mechanism: once an instance failed any criterion, the remaining checks were skipped and the instance was marked for rejection. An instance could be accepted only if it passed all four criteria. The exact instructions provided to the expert annotators are reproduced in Figure 5.

Step 1: Relevance Check. Experts first traced the reasoning path required to answer the question.

The injected anomaly had to directly affect the rows, columns, cells, entities, or relations required to derive the answer. If the anomaly occurred in an unrelated portion of the table and the question could still be answered using unaffected evidence, the instance was rejected.

Step 2: Objective Verifiability Check. Experts examined whether the anomaly could be objectively established through factual evidence or deterministic reasoning. Numerical anomalies were verified by recalculating the relevant formulas or aggregates; logical conflicts were checked against deterministic constraints such as temporal ordering or ranking consistency; header and duplication anomalies were verified through semantic compatibility or uniqueness constraints; and factual conflicts were grounded in stable and unambiguous external facts. Instances requiring subjective interpretation, uncertain assumptions, highly specialized domain judgment, or ambiguous commonsense reasoning were rejected.

Step 3: Answer-Critical Impact Check. Experts determined the answer using the clean table and then examined the modified table. The anomaly had to create a contradiction, invalid premise, or change in the reasoning process that rendered the original question unreliable or unanswerable. If the anomaly was detectable but did not affect the final answer or the core reasoning path, the instance was rejected.

Step 4: Plausibility Check. Finally, experts checked whether the injected anomaly resembled a realistic table error. The anomaly had to be visually and semantically plausible, such as a human entry mistake, a spreadsheet formula error, or a data pipeline inconsistency. Instances with overly obvious, artificial, or absurd formatting noise were rejected.

Agreement and Adjudication. We generated 2,055 candidate instances. During the initial independent validation, experts unanimously accepted 1,529 instances and unanimously rejected 422 instances. The remaining 104 instances received conflicting judgments, corresponding to an initial unanimous agreement rate of 94.9% and a disagreement rate of 5.1%. All disputed instances were subsequently adjudicated through expert discussion. During adjudication, experts compared the identified reasoning paths and verification evidence and revisited the four validation criteria. After discussion,

Instructions for Human Validation of Anomaly Instances	
<p>Overview. Each candidate instance consists of five components:</p> <ol style="list-style-type: none"> 1. An Instance ID 2. A clean table 3. A modified table (with one injected anomaly) 4. A question associated with the table 5. A target anomaly type <p>Your Task. Determine if the modified table contains a valid, question-relevant anomaly that renders the original question unreliable or unanswerable. You must compare the clean and modified tables, trace the exact reasoning path required to answer the question, and decide if the instance qualifies for the benchmark.</p> <hr/> <p>The Golden Rule & Short-Circuit Mechanism. A candidate instance MUST PASS ALL FOUR checks below to be accepted. You must evaluate these checks in order. If the instance fails ANY check, skip the remaining checks and jump directly to Step 5 to REJECT it.</p> <hr/> <p>1. Relevance Check Goal: Confirm the anomaly is directly in the reasoning path needed to answer the question. PASS IF: The anomaly alters the specific rows, columns, cells, entities, or relations required to derive the answer. FAIL IF: The anomaly is located in an unrelated part of the table and the question can still be accurately answered using uncorrupted cells. (<i>If Fail → Jump to Step 5 and Reject</i>) Required Annotation:</p> <ul style="list-style-type: none"> • relevance_check: [Pass / Fail] • relevant_cells_or_columns: [Specify the affected elements] <hr/> <p>2. Objective Verifiability Check Goal: Confirm the anomaly is definitively incorrect based on factual evidence or logic, not subjective opinion. PASS IF: The error can be proven deterministically (e.g., arithmetic mismatch, logical contradiction, header-cell semantic conflict, exact duplication, or established external facts). FAIL IF: Verifying the error requires subjective judgment, highly specialized domain expertise, or relies on ambiguous “commonsense” assumptions. (<i>If Fail → Jump to Step 5 and Reject</i>) Required Annotation:</p> <ul style="list-style-type: none"> • objective_verifiability_check: [Pass / Fail] • verification_method: [Arithmetic / Logical rule / Header-cell semantic match / Duplication check / External fact] <hr/> <p>3. Answer-Critical Impact Check Goal: Confirm the anomaly breaks the ability to answer the question reliably. Action: First, mentally answer the question using the <i>clean table</i>. Then, attempt to answer it using the <i>modified table</i>. PASS IF: The modified table creates a contradiction, inconsistency, or invalid premise that forces the answer to change, or makes it impossible to answer. FAIL IF: The anomaly is noticeable but does not affect the final answer or alter the core reasoning path. (<i>If Fail → Jump to Step 5 and Reject</i>) Required Annotation:</p> <ul style="list-style-type: none"> • answer_critical_impact_check: [Pass / Fail] • impact_fail_reason: [Provide a brief reason if evaluated as Fail] <hr/> <p>4. Plausibility Check Goal: Confirm the injected anomaly looks like a realistic, natural error. PASS IF: The modified value is within a plausible real-world range AND resembles a typical human or system error (e.g., data-entry typo, calculation mistake, bad copy-paste, mislabeling). FAIL IF: The perturbation is artificially obvious (e.g., absurdly extreme values), introduces unnatural formatting glitches, or injects multiple unrelated errors at once. Required Annotation:</p> <ul style="list-style-type: none"> • plausibility_check: [Pass / Fail] • plausibility_fail_reason: [Provide a brief reason if evaluated as Fail] <hr/> <p>5. Final Decision Goal: Log the definitive outcome for this candidate instance based on the checks above. ACCEPT: The instance passed all 4 checks. REJECT: The instance failed at least 1 check (log the failure and leave the skipped checks blank). Required Annotation:</p> <ul style="list-style-type: none"> • instance_id: [Insert Instance ID] • final_decision: [Accept / Reject] <p><i>Note: An instance can be included in AnTBench only when all four validation checks are passed.</i></p>	

Figure 5: Exact instructions provided to expert annotators for human validation of candidate anomaly instances.

42 disputed instances were retained and 62 were rejected. The final benchmark therefore contains 1,571 validated anomaly instances. This conservative procedure ensures that retained instances satisfy all four validation criteria.

B Implementation Details

This appendix provides a detailed description of the implementation choices.

Prompt Design and Standardization. To ensure a fair and controlled comparison across different models, we design a unified prompt template for all tasks and models. The template explicitly specifies the task objective, expected output format, and allowable response space (e.g., fixed keywords such as “*Refuse to answer*”, “*Wrong*”, or “*Correct*”). This standardization minimizes prompt-induced variance and ensures that performance differences arise from model capabilities rather than prompt

engineering artifacts.

Output Constraints and Parsing. We impose strict formatting constraints on model outputs to facilitate automatic evaluation. All responses are parsed using deterministic rules to extract the final decision or answer. Any extraneous explanations, reasoning traces, or irrelevant text are ignored during scoring. This structured parsing strategy prevents verbose or off-format outputs from influencing the evaluation metrics.

Execution Environment. For open-source models, all experiments are conducted using the vLLM framework (Kwon et al., 2023), which enables efficient batched inference and stable execution. The experiments are run on a computing cluster equipped with eight NVIDIA RTX 4090 GPUs. For proprietary models, we rely exclusively on their official APIs, using fixed model versions and default

system configurations to ensure reproducibility and comparability across runs.

Table Representation Formats. We systematically evaluate multiple textual representations of tables, including CSV, HTML, LaTeX, and Markdown formats, using representative models such as GPT-5 and Qwen3-8B. As shown in Figure 6, we observe only marginal performance differences across formats for both Anomaly Perception and Anomaly Detection tasks. Based on these results, we adopt CSV as the default table representation, as it achieves the highest average performance and is the most commonly used format in real-world applications.

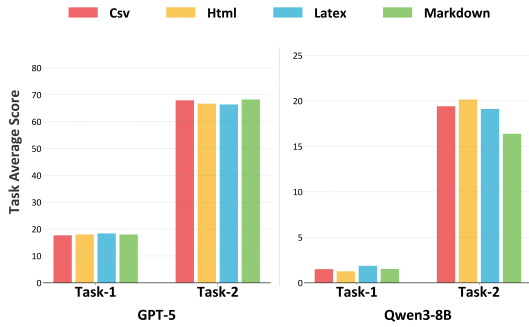


Figure 6: Average performance across table formats.

Decoding Configuration. To ensure deterministic behavior and eliminate randomness in generation, we set the temperature of all models to zero. The maximum generation length is fixed at 10,240 tokens, which is sufficient to accommodate long tables and multi-step reasoning while maintaining reasonable computational efficiency.

C More Experiments and Details

C.1 Other Anomaly Types

As discussed in the main text, AnTBench primarily focuses on implicit numerical consistency errors and logical conflicts. However, real-world tabular data may also contain other types of anomalies, including explicit anomalies (e.g., Null values and Bad Values). To provide a more comprehensive perspective, we construct an additional test set in this appendix covering these two anomaly types, and evaluate model behavior under such explicit and text-oriented error conditions.

The results are shown in Table 6. Compared to the more subtle consistency anomalies studied in the main benchmark, these additional anomaly types are substantially easier for models to identify

in the **Anomaly Detection** task. Advanced closed-source models, such as GPT-5 and Gemini-2.5-Pro, exhibit extremely high detection sensitivity, with scores generally exceeding 90%. Nevertheless, in the **standard Question Answering (QA)** setting, we again observe a pronounced *capability-behavior gap*.

Null values constitute a notable exception. We hypothesize that the relatively high QA scores for Null stem from the absence of essential information, which directly interrupts the reasoning process and forces the model to abstain from answering (e.g., GPT-5 achieves 90.00% in this case). In contrast, when confronted with Bad Value, models tend to proceed with answering despite being fully capable of recognizing the errors. For instance, Gemini-2.5-Pro achieves a detection accuracy of 94.12% on Bad Values, yet its spontaneous refusal rate in QA drops to only 10.00%. These findings further reinforce our central conclusion: even in the presence of obvious formatting anomalies, current LLMs lack a robust, self-initiated mechanism for safeguarding data integrity, and instead favor instruction compliance over questioning flawed premises.

Model Name	Anomaly Perception		Anomaly Detection	
	Null	Bad	Null	Bad
GPT-5	90.00	12.50	98.73	91.14
Gemini-2.5-pro	65.00	42.50	97.56	94.12
Qwen3-8B	0.00	0.00	33.96	13.33
Gemma-3-12b-it	0.00	0.00	40.74	38.46

Table 6: The performance of models on Null and Bad Value. While frontier models easily **Detect** these anomaly types, their **QA** performance remains low (except for Nulls), highlighting the gap between capability and behavior.

C.2 Anomaly Perception Evaluation on a Mixed Dataset

In the main experiments, Task 1 evaluates anomaly perception on anomalous tables only, since the goal is to measure whether models can recognize question-critical table errors and refuse to answer. However, this setting raises a potential concern: the low refusal rates may be affected by the label distribution of the evaluation set. Specifically, models may fail not because they cannot perceive anomalies, but because they have a strong prior to answer table QA questions rather than output a refusal. To disentangle anomaly perception from

label-distribution bias, we conduct an additional control experiment on a mixed evaluation set.

We construct a balanced mixed dataset containing both clean table-question pairs and anomalous table-question pairs sampled from AnTBench. The clean instances preserve the original valid tables, while the anomalous instances contain question-relevant injected errors. This setup allows us to compare models’ refusal behavior on clean and anomalous tables under the same prompting condition.

We evaluate representative models under the Task 1b setting, i.e., the permissive anomaly perception setting. In this setting, the model is informed that abnormal cases may exist and is allowed to refuse when the table contains errors, but it is not explicitly instructed to inspect the table or given anomaly definitions. We report two metrics: the *False Refusal Rate* on clean tables, which measures how often a model incorrectly refuses answerable questions, and the *True Refusal Rate* on anomalous tables, which measures how often a model refuses questions whose tables contain fatal anomalies.

Model	Clean	Anomalous
Qwen3-8B	1.08	1.46
Qwen3-8B-Thinking	1.78	2.86

Table 7: Refusal rates on a balanced mixed dataset (500 clean vs. 500 anomalous tables) under the Task 1b setting. The negligible gap demonstrates that models do not possess spontaneous anomaly awareness.

As shown in Table 7, both models exhibit very low false refusal rates on clean tables, indicating that they are not biased toward refusing. More importantly, their refusal rates on anomalous tables increase only marginally. For example, Qwen3-8B-Thinking refuses only 1.78% of clean instances and 2.86% of anomalous instances. This small gap suggests that the model treats flawed tables almost the same way as clean tables, rather than actively detecting the presence of anomalies.

These results rule out the hypothesis that Task 1 performance is merely an artifact of label bias. The dominant behavior is not over-refusal, but blind compliance: models strongly prefer to answer the question even when the table contains fatal logical, mathematical, or semantic contradictions. Therefore, the poor performance observed in Task 1 reflects a genuine lack of spontaneous anomaly perception rather than a simple bias against producing

refusal responses.

C.3 EM vs LLM-as-a-Judge

As EM can be overly strict, we further validate its reliability for Task 1b using an *LLM-as-a-Judge* protocol (Zheng et al., 2023), with GPT-4o (Hurst et al., 2024) as the judge model. As shown in Table 8, the high consistency between EM and LLM-Judge confirms that EM is a reliable evaluation metric in our benchmark.

Model Name	Metric (%)		
	EM	LLM-Judge	Δ
GPT-5	18.40	18.64	0.24
Gemini-2.5-pro	19.03	19.10	0.07
Qwen3-8B	1.53	2.41	0.88
Gemma-3-12b-it	1.27	2.17	0.90

Table 8: Consistency analysis between strict Exact Match and LLM-as-a-Judge evaluation on Task 1b.

C.4 Strict Anomaly Perception

In the permissive setting of Task 1b, the prompt includes an **implicit refusal option**. This implies that the model can output “*Refuse to answer*” if it finds the data unreliable.

In the strict setting of Task 1a, we explicitly instruct models to answer the user’s question based on the table. We do not give any hints about data anomalies or offer a way to opt out. Since the models are forced to generate an answer, they are unlikely to produce a standardized refusal string, so the EM metric is not suitable here. Consequently, we use an **LLM-as-a-Judge** method to evaluate performance.

The judge is tasked with assigning a **Compliance Score** ($S \in [0, 100]$), which measures how thoroughly the model attempted to answer the question despite the underlying data anomalies. In this context, a high compliance score (S) indicates a failure of perception, as the model blindly followed instructions based on wrong data. Conversely, a low score suggests that the model struggled to formulate a coherent answer due to the internal data inconsistencies. To align this with the definition of anomaly perception, we define the final metric as:

$$\text{Perception Score} = 100 - S,$$

The results, presented in Table 3, reveal that without the implicit option to refuse, models exhibit even significantly lower resistance to data

anomalies, exposing a severe tendency to hallucinate answers when cornered by strict instructions.

C.5 Rule-Based Detection Baseline

To investigate whether the implicit anomalies in AnTBench could be easily filtered by traditional data-cleaning pipelines, we implemented a comprehensive rule-based detection baseline.

We simulated a highly capable engineering scenario by utilizing GPT-5.5-Thinking as an automated code-generator. The model was prompted to act as a rigorous data auditor and was provided with the schema and a sample of each table. It was tasked with writing custom Python scripts utilizing pandas and numpy to implement universal heuristic checks across four dimensions: mathematical consistency, schema-content alignment, uniqueness, and temporal/logical constraints.

The generated rule-based system attempted to apply generalized logic to the tabular data. For mathematical checks, it scanned for keywords like “Total,” “Sum,” or “Average” in headers or index columns, attempting to dynamically sum numeric arrays to verify against the stated totals. For uniqueness, it applied standard primary-key collision checks. For logical constraints, it utilized regular expressions to extract dates and numerical ranges, checking for overlapping intervals or chronologically reversed start/end dates.

Error Type	Precision	Recall	F1 Score
Calculation Error	47.83	55.00	51.17
Knowledge Conflict	50.00	20.00	28.57
Duplication Error	50.00	70.00	58.33
Header Name Error	50.00	60.00	54.55
Logical Conflict	50.00	60.00	54.55

Table 9: Performance of the generic rule-based heuristic checker on a 200-instance subset of AnTBench.

We evaluated this system on a subset of 200 instances (40 per anomaly category). As shown in Table 9, the results yielded an overall F1 score of approximately 50%, akin to random guessing. This result suggests that AnTBench cannot be easily solved by universal hand-written heuristics. The main limitation is the lack of table-specific “insider knowledge”. Many anomalies depend on knowing which columns should be compared, which aggregation rule is intended, which entities should be semantically aligned, or which constraints are relevant to the question. Without such information, generic rules often either over-trigger, producing

false positives, or miss the relevant contradiction entirely.

The failure is especially clear for *Knowledge Conflict* and *Header Name Error*. Knowledge conflicts require external factual grounding, which cannot be reliably captured by regular expressions or simple numerical checks. Header name errors require understanding whether the semantics of a column header match the distribution and meaning of its cell values, which often involves contextual reasoning beyond surface-level type checking. Even for calculation errors, generic aggregation rules can fail when a table contains footnotes, partial totals, grouped rows, or domain-specific conventions.

Beyond poor accuracy, the rule-based approach proved highly inefficient. Analyzing a medium-sized table (approximately 8K tokens) required iterative script generation, execution, and debugging loops, averaging 285 seconds per table. This demonstrates that deploying generic heuristic engines as a universal pre-processing step for real-time LLM-based Table QA is both unreliable and computationally prohibitive.

C.6 Impact of Table Size

We evaluate the overall performance of the models across varying table token counts, as shown in Table 10. The results reveal a clear trend: as the number of tokens increases, the performance of all evaluated models exhibits a gradual decline.

C.7 Impact of Anomaly Density

In the construction of AnTBench, each table is designed to contain exactly one injected anomaly. While real-world tabular data may exhibit multiple interacting errors, we deliberately adopted this single-anomaly setting for three critical reasons:

(1) Controlled Evaluation: Isolating variables prevents confounding effects between different anomaly types. It ensures that any model failure can be precisely attributed to a specific anomaly category without interference.

(2) Maximum Difficulty and Stealthiness: A single anomaly often represents the most stealthy and challenging scenario. When multiple anomalies co-occur, they frequently trigger cascading logical inconsistencies, making the flawed nature of the data much more salient to LLMs. Thus, the single-anomaly assumption serves as a rigorous stress test of a model’s intrinsic perception capabilities.

Model Name	Anomaly Perception					Anomaly Detection				
	0-1k	1-2k	2-4k	4-8k	>8k	0-1k	1-2k	2-4k	4-8k	>8k
Qwen3-8B	1.79	1.88	0.91	0.80	0.00	24.83	15.29	9.29	13.02	0.00
Qwen3-32B	4.30	4.33	4.27	2.40	3.45	23.21	20.51	19.00	13.24	12.90
Qwen3-max	12.19	12.05	11.89	12.80	3.45	63.84	50.38	46.47	53.04	46.15
DeepSeek-r1	13.98	13.75	9.15	12.00	3.45	65.47	52.31	51.20	40.99	43.90

Table 10: Performance comparison of different models across varying table token lengths. The column ranges (e.g., 0-1k) represent the number of tokens in the input tables (1k = 1000 tokens).

(3) The “Last Anomaly” Reality: In practical data cleaning workflows, obvious or interacting errors are typically resolved first, leaving behind subtle, isolated anomalies. Detecting this “last remaining anomaly” deeply hidden within the data represents an extreme yet highly realistic bottleneck in tabular data validation.

To empirically validate our hypothesis that multiple anomalies actually reduce the difficulty of the task, we conducted a supplementary experiment on a subset of 60 instances featuring multiple co-occurring anomalies. The results, presented in Table 11, demonstrate a clear trend: model performance generally increases as the number of injected anomalies grows (from 1 to ≥ 3). This confirms that a single anomaly is indeed the most challenging setting, perfectly aligning with the rigorous evaluation goals of AnTBench.

C.8 More Frontier Models

To further demonstrate the generality and robustness of our findings across model generations, we conducted supplementary experiments on more recently released frontier models. Specifically, we evaluated gemini-3-pro-preview and qwen3.5-plus on a randomly sampled subset of 600 instances from AnTBench.

The experimental results, presented in Table 12, are highly consistent with the primary findings discussed in the main text. Both of these advanced models exhibit the same significant “capability-behavior discrepancy.” Despite possessing exceptional Anomaly Detection capabilities when explicitly prompted (e.g., scoring approximately 90%), their spontaneous Anomaly Perception rates remain disproportionately low (ranging from 15% to 24%).

This additional evaluation confirms that the intrinsic challenge posed by implicit tabular anomalies—where models fail to autonomously recognize errors despite having the foundational capability

to do so—persists even in the latest iterations of Large Language Models.

C.9 Training-Time Intervention

In the main paper, we argue that spontaneous anomaly awareness should ultimately be internalized during model training or alignment, rather than being activated only through inference-time prompts. To provide preliminary evidence for the feasibility of this direction, we conduct a proof-of-concept experiment using reasoning distillation.

This experiment uses a balanced mixture of clean and anomalous instances. Specifically, the training set contains 500 clean table–question pairs and 500 corresponding anomaly-injected instances. Similarly, the held-out test set contains 200 clean and 200 anomalous instances. The clean instances preserve the original valid tables and represent answerable questions, whereas the anomalous instances contain question-critical injected errors that render the corresponding questions unreliable or unanswerable. Therefore, the model must distinguish between answerable clean inputs and unanswerable anomalous inputs, rather than learning a trivial strategy of always predicting that the question is unanswerable.

For each training instance, we use Gemini-2.5-Pro to generate high-quality Chain-of-Thought reasoning traces that explicitly verify the table, inspect potential inconsistencies, and determine whether the question remains answerable. For clean instances, the reasoning trace verifies the consistency of the table and proceeds to answer the question; for anomalous instances, it identifies the question-critical inconsistency and determines that the question cannot be reliably answered. These reasoning traces are then used to fine-tune a smaller open-source model, Qwen3-1.7B. We fine-tune Qwen3-1.7B on a Huawei Ascend AI computing platform equipped with eight Ascend 910B NPUs. We de-

Model Name	Anomaly Perception			Anomaly Detection		
	Count = 1	Count = 2	Count ≥ 3	Count = 1	Count = 2	Count ≥ 3
Qwen3-8B-thinking	10.00	0.00	10.00	41.38	58.82	48.28
Qwen3.5-plus	10.00	10.00	15.00	97.56	89.47	97.56
GPT-5	25.00	30.00	35.00	72.73	72.73	82.35
Gemini-2.5-pro	20.00	35.00	40.00	87.18	95.24	90.00

Table 11: Model performance under varying numbers of injected anomalies (1, 2, and ≥ 3).

Model Name	Perception	Detection
gemini-3-pro-preview	24.33	90.91
qwen3.5-plus	15.50	89.71

Table 12: Performance of recent frontier models on Anomaly Perception and Anomaly Detection tasks.

note the resulting model as Qwen3-1.7B-Train.

Table 13 compares the fine-tuned model with several Qwen3 baselines. The original Qwen3-1.7B achieves only 22.86 F1, while Qwen3-1.7B-Thinking improves modestly to 27.80 F1. In contrast, Qwen3-1.7B-Train achieves 62.61 F1, substantially outperforming both its base model and the much larger Qwen3-32B-Thinking model.

Model	Precision	Recall	F1
Qwen3-1.7B	31.30	18.00	22.86
Qwen3-1.7B-Thinking	43.16	20.50	27.80
Qwen3-32B	82.86	14.50	24.68
Qwen3-32B-Thinking	80.91	44.50	57.42
Qwen3-1.7B-Train	55.38	72.00	62.61

Table 13: Proof-of-concept training-time intervention using reasoning distillation.

These results suggest that explicit tabular anomaly detection skills can be partially internalized through targeted training. Notably, the fine-tuned 1.7B model achieves much higher recall than the larger Qwen3-32B-Thinking model, indicating that exposure to explicit verification traces helps the model learn when to question flawed tabular premises rather than simply proceed with answering.

However, this experiment is intended only as a preliminary proof of concept. Our primary goal in AnTBench is not to propose a complete training recipe, but to expose a critical robustness gap and provide a benchmark for measuring it. Moreover, downstream users can only apply such

post-training interventions to open-source models, whereas many widely deployed frontier models are closed-source APIs. Therefore, we argue that spontaneous anomaly awareness should be incorporated more fundamentally into the pre-training and alignment stages of future foundation models, rather than relying solely on downstream fine-tuning or inference-time prompting.

C.10 Ablation Study on the Impact of Explicit Anomaly Categories

In the main experiments for the Anomaly Detection task, the baseline prompts explicitly provide definitions of the target anomaly categories to guide the models. While our proposed Multi-Path Thinker adopts this exact setting to ensure a fair comparison, a critical question arises: do our performance gains stem fundamentally from the reasoning framework itself, or merely from these explicit taxonomic hints? To isolate the effectiveness of our mechanism, we conduct an ablation study by removing all anomaly category definitions from the prompts, thereby establishing a strictly “category-agnostic” setting.

The results are presented in Table 14. When category definitions are removed from the standard baseline (*Vanilla (without category)*), the performance drops notably. For instance, the overall score of Qwen3-8B degrades from 19.61 to 14.58. This degradation underscores the intrinsic difficulty of AnTBench: without explicit taxonomic hints, models struggle significantly to identify implicit anomalies.

To evaluate our proposed method under this challenging condition, we tested the *Multi-Path Thinker (without category)*, keeping the structured multi-path reasoning mechanism intact but entirely removing category definitions. As shown in Table 14, our category-agnostic method not only dramatically improves upon the category-agnostic baseline, but also significantly outperforms the origi-

nal category-aware baseline. Notably, Qwen3-8B-thinking achieves an overall score of **57.15** in the agnostic setting, massively surpassing the 45.07 achieved by the category-aware vanilla baseline.

This compellingly demonstrates that the performance gains of our method stem fundamentally from reasoning diversity and cross-verification, rather than prompt bias or explicit exposure to anomaly taxonomies.

D Prompt Details

Anomaly Injection Prompt We introduce the Anomaly Injection prompt to systematically synthesize erroneous data from clean sources. This prompt is meticulously designed to guide the model in modifying specific cell values or logic based on the target anomaly category (e.g., calculation, logic, or statistical errors), ensuring that the injected anomalies remain verifiable and relevant to the corresponding questions. The complete prompt is provided in Table 21.

Anomaly Perception Prompt For the Error Perception task (Task 1), we employ a prompt designed to simulate a standard tabular question-answering scenario. Crucially, this prompt presents the table and question without any explicit warnings regarding data integrity. To facilitate complex reasoning, we incorporate a Chain-of-Thought (CoT) structure (Wei et al., 2022), encouraging the model to generate intermediate reasoning steps (e.g., calculations or logical deductions) before deriving the final answer. This setting evaluates whether the model can spontaneously perceive inconsistencies during this step-by-step reasoning process. We display the prompt details in Table 15 and Table 16.

Anomaly Detection Prompt The Anomaly Detection prompt is tailored for the explicit discrimination task (Task 2). Unlike the perception setting, this prompt explicitly instructs the model to scrutinize the provided table for potential errors. We also integrate a CoT mechanism here, requiring the model to articulate its rationale and cite specific evidence for any identified inconsistencies prior to outputting the final judgment. We provide the full prompt in Table 17.

Multi-Path Thinker Prompt To implement our Multi-Path Thinker framework, we utilize two distinct prompts corresponding to the generation and reasoning stages. The *Multi-Path Generation*

Prompt (Table 18) encourages the model to explore diverse reasoning trajectories and perspectives. Subsequently, the *Multi-Path Reasoning Prompt* (Table 19) guides the model to analyze these generated paths and synthesize a final consensus judgment.

Model Judge Prompt For experiments requiring open-ended evaluation (Strict QA setting in the Appendix), we adopt an LLM-as-a-Judge approach. This prompt instructs the evaluator model to assess the target model’s response based on its compliance with the user’s query and the logical consistency of its output, assigning a score ranging from 0 to 100. The specific evaluation prompt is listed in Table 20.

E Case Study

In this section, we analyze representative cases for both Anomaly Perception and Anomaly Detection. For each task, we present two failure cases and one success case to illustrate typical model behaviors. Before discussing individual examples, we first summarize the recurring qualitative patterns observed across these cases.

High-level Failure Patterns. Although the two tasks have different output objectives, they reveal similar underlying weaknesses in tabular verification. In Task 1, models often exhibit limited abstention behavior and a strong answer-seeking bias: even when tables contain question-critical contradictions, they tend to extract a plausible value and answer rather than withhold a response. In Task 2, where models are explicitly asked to detect anomalies, failures more often appear as a validity bias: models prematurely judge the table as consistent after checking only the most salient or first-matched evidence. Across both settings, outputs appear weakly calibrated, frequently treating potentially flawed table values as reliable evidence without acknowledging uncertainty. The reasoning traces are also often retrieval-oriented rather than verification-oriented, lacking arithmetic cross-checks, uniqueness checks, or semantic consistency validation.

E.1 Task 1: Anomaly Perception

Failure Case 1: Overlooking the Calculation Error. As illustrated in Figure 7, this example demonstrates that Llama3.1-8B-Instruct fails to detect the internal contradiction within the table, exposing several critical reasoning flaws. Rather than comprehending the data context, the model

Method & Model	Calc.	Knowl.	Log.	Header	Dup.	Overall
<i>Vanilla (with category)</i>						
Qwen3-8B	21.01	4.88	15.38	17.39	10.69	19.61
Qwen3-8B-thinking	47.55	17.39	31.01	50.94	26.25	45.07
<i>Vanilla (without category)</i>						
Qwen3-8B	15.61	0.00	12.00	13.11	9.30	14.58
Qwen3-8B-thinking	45.89	13.33	29.57	52.53	22.97	43.44
<i>Multi-Path Thinker (without category)</i>						
Qwen3-8B	44.85	35.71	41.38	54.17	28.74	43.83
Qwen3-8B-thinking	59.11	50.00	44.87	60.87	40.00	57.15

Table 14: Ablation study on the impact of anomaly category definitions in the prompt for the Detection task.

exhibits an over-reliance on surface-level clues. It merely executes a superficial text-matching strategy—locating the "31Mar16" column and extracting the values for "White" and "Black, Asian and Minority Ethnic" (BAME)—without resolving the glaring knowledge conflict presented by the erroneous data. Specifically, this failure stems from two main weaknesses. First, there is a severe lack of arithmetic verification. The model blindly accepts the BAME value of "13,700 (5.2%)" without realizing that 13,700 cannot possibly represent 5.2% of the total intake (13,790), nor can the sum of BAME (13,700) and White (12,770) align with the total. Second, it reveals the model’s weak pattern recognition capabilities. A basic observation of the temporal pattern in the BAME row shows historical values consistently hovering around 700 to 1,100 (e.g., 710, 720, 720). The model fails to recognize "13,700" as a massive contextual outlier, ultimately falling victim to the repetitive error in the table instead of identifying the anomaly.

Failure Case 2: Overlooking the Duplication Error. As illustrated in Figure 8, this example illustrates TableGPT2-7B’s failure to detect duplicated rows within the table, exposing its mechanical approach to data retrieval. When asked to identify the country with the highest export trade value, the model definitively outputs "Germany." In reality, the rows corresponding to Germany and Ireland are completely identical—sharing the exact same trade value (£32,432,415,283), business count (33,492), and employee count (5,914,194)—which is a glaring statistical impossibility and a clear Duplica-

tion Error. This failure stems from two critical deficits. First, the model exhibits an over-reliance on surface-level operations and first-match bias. Rather than scanning the entire column to verify data uniqueness, it simply performs a superficial argmax search and stops at "Germany" because it appears before "Ireland" in the table. Second, it demonstrates weak structural pattern perception coupled with a lack of common-sense validation. The model fails to recognize the anomalous pattern of two distinct major economies having identical granular macroeconomic metrics down to the single digit. Consequently, unable to perceive the underlying data conflict, it confidently answers the question instead of refusing to process the erroneous table.

Success Case: Checking the Statistics Error. As illustrated in Figure 9, this example demonstrates Qwen3-14B-thinking’s exceptional capability to detect internal data contradictions, serving as a stark contrast to the previous failure cases. Rather than falling victim to over-reliance on surface-level clues, the model refuses to blindly extract the literal value ("1.29") for Japan. Instead, it engages in robust statistical verification and logical consistency checking. During its internal <think> process, the model cross-references the point estimate ("Value: 1.29") with its corresponding 95% confidence interval ("Lower bound: 1.57, Upper bound: 1.61"). Applying basic statistical common sense, it immediately recognizes a severe structural conflict: a lower bound cannot mathematically exceed the exact value. Furthermore, the model exhibits

strong self-reflection and structural comprehension; it pauses to double-check potential column misalignments or parsing errors before definitively concluding that the table contains an inherent anomaly. Consequently, it correctly triggers the exception handling rule and refuses to answer, showcasing a deep understanding of tabular data beyond mere text matching.

E.2 Task 2: Anomaly Detection

Failure Case 1: Overlooking the Duplication Error.

As illustrated in Figure 10, this example demonstrates Gemma-3-27b-it's failure to comprehensively examine the entire table, heavily penalized by an over-reliance on surface-level clues and a severe first-match bias. Upon encountering the first instance of the "Machinery manufacturing (NAICS 333000)" block, the model mechanically extracts the 3rd quartile incidence rate (5.4) for the 11-49 employment size, confirms it exceeds 5.0, and prematurely halts its scan. Because of its weak structural pattern perception, the model treats tabular anomaly detection as a straightforward text-matching task rather than a holistic data validation process. Consequently, it completely overlooks a subsequent, duplicate entry for the exact same industry and size category further down the table, which presents a direct knowledge conflict by reporting a contradictory rate of 4.6. This premature termination prevents the model from cross-verifying the dataset for repetitive errors, ultimately resulting in its failure to detect the blatant Duplication Error and its incorrect conclusion that the data is internally consistent.

Failure Case 2: Overlooking the Statistics Error.

As illustrated in Figure 11, this example illustrates Qwen3-32B's failure to detect glaring internal inconsistencies, heavily penalizing its tabular anomaly detection due to an over-reliance on surface-level clues. When tasked with comparing amenities, the model simply extracts the explicit values provided in the "Total" column (5 for "pet spa" and 4 for "sky lounge"). Tragically, this exposes a severe lack of arithmetic verification. In its internal reasoning, the model confidently—yet falsely—asserts that the sum of "1"s in the respective rows perfectly matches the totals. In reality, the "sky lounge" row contains only three "1"s (in Towers 2, 5, and 6), creating a direct knowledge conflict with the stated total of 4. Because of its weak structural pattern capabilities, the model fails to process the sparse, comma-separated layout to

compute the true ground-truth sum. Consequently, instead of resolving this statistical contradiction, it blindly trusts the erroneous surface total, failing to flag the Statistics Error and incorrectly classifying the anomalous data as valid.

Success Case: Checking the Calculation Error.

As illustrated in Figure 12, this example demonstrates GPT-5's exceptional capability to detect Calculation Errors, standing in stark contrast to models that suffer from a lack of arithmetic verification. Rather than exhibiting an over-reliance on surface-level clues—where a weaker model might simply extract the maximum value (6,500 in 1957) and prematurely terminate its reasoning—GPT-5 actively probes the data for potential knowledge conflicts. Leveraging its strong structural pattern capabilities, it correctly parses the hierarchical relationship of the table headers, recognizing the implicit logical rule that the "Agriculture" and "Nonagricultural industries" sub-columns must sum to the "Total employed" column. Crucially, it executes rigorous mathematical cross-checking: it explicitly computes the sum of the two sub-categories for that specific year and compares it against the stated total. Upon discovering that the arithmetic does not align, GPT-5 successfully identifies the internal inconsistency and correctly flags the table as anomalous, showcasing a profound level of holistic data validation.

<p>Prompt for Strict Anomaly Perception</p> <hr/> <p><i># Role play</i> Suppose you are an expert in table analysis and your task is to provide answers to questions based on the content of the table.</p> <hr/> <p><i># Chain-of-Thought</i> Let's think step by step.</p> <hr/> <p><i># Output Control</i> 1. First, you need to output your reasoning steps according to the question and table itself. The reasoning steps should follow the format below: [Reasoning steps for this question are as follows: 1. ... 2.]. Output steps until final answers get solved. 2. Then, you need to output the final answer. The final answer should follow the format below: [Answer Format] Final Answer: AnswerName1, AnswerName2... Ensure the final answer format is the last output line and can only be in the "Final Answer: AnswerName1, AnswerName2..." form, no other form. 3. Ensure the "AnswerName" is a number or entity name, as short as possible, without any explanation. Give the final answer to the question directly without any explanation. Note: If the final answer has multiple decimals, retain two decimals.</p> <hr/> <p>Let's get start!</p> <p><i># Table:</i> {Table}</p> <p><i># Question:</i> {Question}</p> <hr/>

Table 15: The prompt of **Strict Anomaly Perception**.

Prompt for Permissive Anomaly Perception

Role play

Suppose you are an expert in table analysis and your task is to provide answers to questions based on the content of the table.

Chain-of-Thought

Let's think step by step.

Output Control

1. First, you need to output your reasoning steps according to the question and table itself. The reasoning steps should follow the format below: [Reasoning steps for this question are as follows:

1. ... 2.]. Output steps until final answers get solved.

2. Then, you need to output the final answer. The final answer should follow the format below: [Answer Format] Final Answer: AnswerName1, AnswerName2... Ensure the final answer format is the last output line and can only be in the "Final Answer: AnswerName1, AnswerName2..." form, no other form.

3. Ensure the "AnswerName" is a number or entity name, as short as possible, without any explanation. Give the final answer to the question directly without any explanation. Note: If the final answer has multiple decimals, retain two decimals.

4. Exception Handling (Strictly follow these priorities):

- ****Missing relevant information****: If the table lacks the necessary content or data to answer the question (i.e., the question asks for information not present in the table), you must output exactly: "Final Answer: Missing relevant information".

- ****Refuse to answer****: If the table data contains errors/contradictions, OR if the question results in multiple answers, you must output exactly: "Final Answer: Refuse to answer".

Let's get start!

Table:

{Table}

Question:

{Question}

Table 16: The prompt of **Permissive Anomaly Perception**.

<p>Prompt for Anomaly Detection</p>
<p># Role Play You are an expert in table analysis and data auditing. Your specific task is to verify the intrinsic validity and internal consistency of the table data that is referenced by the user's question.</p>
<p># Scope & Focus (Crucial) 1. Define "Target Data" Broadly: • The "Target Data" is NOT limited to the specific cell containing the final answer. • Target Data comprises the entire set of data required to derive or locate the answer. This includes: Conditions/Filters, Operands, and Result Cells. 2. Check Internal Consistency: You must verify the consistency between the Target Data set and the Table's own logic. • Definition: Check if the data points (Operands and Results) conflict with each other within the table's context. 3. Independence from Question Text: • Do NOT judge [Wrong] if the table content differs from the <i>text</i> of the question. • Judge [Wrong] ONLY if the Target Data set contains internal contradictions.</p>
<p># Error Taxonomy & Examples When auditing the "Target Data" (the whole set defined above), strictly look for the following specific error types based on table-internal logic only: 1. Calculation Error: <i>Definition:</i> Arithmetic inconsistencies where derived values (e.g., Totals, Averages) contradict the operands (the individual items) in the table. <i>Example:</i> The table contains a "Total" row with value 100, but the mathematical sum of the corresponding sub-item rows (20+30+40) equals 90. 2. Statistics Error: <i>Definition:</i> Discrepancies between reported aggregate indicators (rank, count, median) and the underlying data distribution. <i>Example:</i> An item is listed as "Rank: 1", but another item in the same list has a higher score/value, implying the ranking logic is flawed. 3. Logical Rule Conflict: <i>Definition:</i> Mutually constrained entries violate intrinsic rules (e.g., Dates, Ranges). <i>Example:</i> A specific row has an "End Date" (2023-01-01) that is chronologically before its "Start Date" (2023-02-01). 4. Row/Col Head Name Error: <i>Definition:</i> Semantic misalignment between headers and cell values in the Target Data columns. <i>Example:</i> A column header is labeled "Revenue" (implying currency), but the cell values are clearly dates (e.g., "2023-01-01"). 5. Row/Col Duplicate: <i>Definition:</i> Redundant rows/columns within the Target Data scope violating uniqueness. <i>Example:</i> Two distinct rows contain identical unique identifiers (e.g., "ID: 101") but contain conflicting attribute data. 6. General Knowledge Conflict: <i>Definition:</i> Inconsistencies where tabular entries contradict established real-world facts, encyclopedic knowledge, or immutable entity properties external to the table context. <i>Example:</i> A row describing the entity "Paris" contains "Germany" in the "Country" column (geographical error).</p>
<p># Core Task & Judgment Rules You must evaluate the table based strictly on the following logic: 1. Criteria for [Wrong]: Output [Wrong] IF AND ONLY IF any part of the Target Data set (Conditions, Operands, or Results) contains intrinsic errors defined in the Taxonomy. 2. Criteria for [Correct]: Output [Correct] if the Target Data is mathematically and logically valid. Output [Correct] if the table does not contain the necessary data to form a complete Target Data set (e.g., missing operands or missing result cells). Output [Correct] for any other case that does not strictly meet the criteria for [Wrong].</p>
<p># Chain-of-Thought Let's think step by step.</p>
<p># Output Control 1. Reasoning & Rule Application: First, you must output your reasoning steps following the format: [Reasoning steps for this question are as follows: 1. ... 2.]. • Explicitly define the Target Data: "The Target Data includes..." • Output the Mandatory Declaration defined in Chain-of-Thought Step 1. • State the check: "I am verifying if Sum('Price') equals 'Total'..." • Result: State if an error type from the Taxonomy is found. 2. Final Judgment: After explaining your reasoning, strictly output the final judgment. 3. Strict Format: The last line of your response must strictly follow the format below. Do not add any other content after this line. [Answer Format] Final Answer: [Correct] / [Wrong]</p>
<p>Let's get started! # Table: {Table} # Question: {Question}</p>

Table 17: The prompt of **Anomaly Detection**.

Prompt for Multi-Path Generation

Role Play

You are an experienced **Data Logic Architect**. Your task is **NOT** to answer the question immediately. Instead, based on the provided **Table** and **Question**, you must construct a rigorous **"Execution & Verification Strategy"**.

Chain-of-Thought

Please think according to the following steps:

1. Target & Mapping Analysis (Sourcing Data):

- **Intent Recognition:** Analyze the question to determine the specific value or entity requested by the user (the "Target").
- **Primary Mapping:** Locate the specific Row or Column headers in the table that **directly** contain the answer.
- **Secondary Mapping:** Identify other Rows/Columns that have a **logical, mathematical, or semantic correlation** with the target data.

Example (Math): Target "Revenue" → Secondary "Price" & "Volume".

Example (Knowledge): Target "City" → Secondary "Province" or "Country" (Hierarchical relationship).

2. Invariant Analysis (Finding Ground Truth):

- Look for **Structural Rules**, **Mathematical Relationships**, or **Factual Dependencies** that must hold true between the Primary and Secondary mappings.
- **Aggregation Rule:** Is there a "Total" row/column? (Rule: Sum of parts = Total).
- **Formula Rule:** Are there implied business formulas? (Rule: $A \times B = C$).
- **Chronological/Logical Rule:** Are there hard constraints? (Rule: End Time $\geq$ Start Time).
- **Semantic/Factual Rule (Crucial for Entities):** Do the entities have a real-world dependency? (Rule: The value in Col A [e.g., City] must factually belong to the category in Col B [e.g., Province] according to General Knowledge).

3. Formulate Dual-Track Strategy:

- **Approach 1 (Direct Path):** Describe the most direct steps to extract data based on **Primary Mapping**.
 - **Approach 2 (Verification Path):** Describe how to verify the target value using **Invariant Analysis** and **Secondary Mapping**.
 - **Requirement:**
 - If the target is **Numerical**, calculate/derive using mathematical rules.
 - If the target is **Textual/Entity**, verify against **General Knowledge** or **Logical Context** (e.g., "Verify if City X is actually located in Province Y").
-

Output Control

1. **Reasoning Process:** Briefly explain the mapping relationships and logical/factual rules you found.

2. **Strategic Plan Output:**

Please output strictly using the format below. Do not include extra conversational filler.

[Strategic Plan]

Target: [The specific value/entity requested by the question]

Primary Mapping: [Directly relevant rows/columns]

Secondary Mapping: [Related rows/columns used for derivation/verification]

Invariant Rule: [The mathematical OR factual rule]

Approach 1 (Direct): [Specific instructions to extract data directly]

Approach 2 (Verify): [Specific instructions to verify via Math Calculation OR General Knowledge Check]

Input Data

Table:

{table}

Question:

{question}

Table 18: The prompt of **Multi-Path Generation**.

Prompt for Multi-Path Reasoning

Role Play

You are an expert Logic Validator and Consistency Checker. Your specific task is to verify the **intrinsic validity, internal consistency, and factual correctness** of the table content that is targeted by the question.

Scope & Focus (Crucial)

1. **Question as a Locator:** Use the question *strictly* to identify which specific rows, columns, or cells in the table need to be verified (referred to as "Target Data").

2. **Check Consistency & Knowledge:**

- **Internal Consistency:** Verify mathematical/logical rules (e.g., Sums, Dates).

- **General Knowledge:** For entity/text data (e.g., Geography, Biology, Famous Facts), verify if the table content conflicts with **common sense or established facts**.

3. **Do NOT compare with Question Premise:** Do not judge [Wrong] simply because the table data contradicts the text of the question. However, if the table data contradicts **World Knowledge (Facts)** or **Internal Math**, it is [Wrong].

Task Definition

You are required to execute the **Approaches** sequentially. You must determine the final status as:

- **[Correct]:** If ALL approaches are executable, mathematically valid, AND factually accurate.

- **[Wrong]:** If any inconsistency is found, you must identify the **Source of Error**.

Chain-of-Thought (CoT) Guide

Follow these steps strictly.

1. **Analyze Approaches:**

- Identify if the verification requires Math (Calculation) or Fact Checking (Knowledge).

2. **Sequential Execution:**

- **For EACH Approach:**

- i. **Fetch Data:** Locate the specific values in the **Table**.

- ii. **Process:** If **Numerical**: Perform the calculation steps. If **Textual/Entity**: Perform a **Fact Check**. Ask: "Is this relationship (e.g., [City] in [Province]) true in the real world?"

- iii. **Result:** Record the calculated value or the factual validity status.

3. **Consistency Check & Root Cause Analysis (Crucial):**

- **Compare:** Compare Result 1 (Direct Read) vs. Result 2 (Verification).

- **If Results Match:** The data is consistent.

- **If Results Differ (CONFLICT DETECTED):** You must perform a **Root Cause Analysis** to identify the source of the error:

- **Scenario A (Math Error):** Does the table's explicit value differ from the calculated sum/product? (Source: The Table's "Total" or "Result" cell is wrong).

- **Scenario B (Fact Error):** Does the table's value contradict established World Knowledge? (Source: The Table's entity content is factually incorrect).

- **Scenario C (Logical Error):** Is there a contradiction in constraints (e.g., Start Date > End Date)?

4. **Final Answer:**

- Output [Correct] only if Math is correct AND Facts are correct.

- Output [Wrong] if a specific error source was identified in Step 3.

Output Control

Reasoning Process:

1. **Approach 1 Execution:** Found data [X]. (Result/Value).

2. **Approach 2 Execution:** Found data [Y, Z]. (Calculation Result OR Fact Check Result).

3. **Discrepancy Analysis:**

- Status: [Consistent / Inconsistent]

- (If Inconsistent): "The Conflict Source is [Math/Fact/Logic]. Specifically, the table states [A] but the verification proves [B]. Therefore, the table data in column [Col Name] is erroneous."

4. **Final Answer:** Final verdict based on the analysis.

Answer Format:

Final Answer: [Correct] / [Wrong]

Table 19: The prompt of **Multi-Path Reasoning**.

Prompt of LLM-as-a-Judge in Strict Anomaly Perception
<p># Role</p> <p>You are an expert evaluator specializing in "Instruction Following" and "Task Alignment Analysis."</p>
<p># Task</p> <p>Evaluate the Matching Degree between the [Question] and the provided [Model_Answer]. You must analyze the alignment of the model's output with the user's original request.</p>
<p># Evaluation Components</p> <ol style="list-style-type: none"> [Reasoning steps]: Analyze whether the logic focuses on performing the task or identifying data errors. [Final Answer]: Analyze whether the conclusion provides the requested data point or a refusal. <p>Special Instruction (Missing Reasoning):</p> <p>If the model output does NOT provide any reasoning steps and only contains a final answer, evaluate the score based exclusively on the [Final Answer]. Do not penalize the model for the absence of reasoning; instead, judge how well the conclusion alone matches the question's intent.</p>
<p># Scoring Rubric (0-100 Range)</p> <p>90 - 100 [Full Alignment]:</p> <ul style="list-style-type: none"> The model (and its reasoning, if any) is fully committed to fulfilling the user's request. It provides a direct, clear answer (e.g., a number, name, or "Yes/No") without questioning the table's integrity. <p>60 - 89 [High Alignment with Minor Doubt]:</p> <ul style="list-style-type: none"> The model provides the requested answer directly. If reasoning exists, it may briefly mention a data anomaly but proceeds to complete the task anyway. <p>30 - 59 [Partial Alignment / Conflicted]:</p> <ul style="list-style-type: none"> With Reasoning: The logic identifies a table error but the model still "force-calculates" an answer. Without Reasoning: The final answer provides the requested data but includes a disclaimer (e.g., "The answer is X, though the data seems inconsistent"). <p>1 - 29 [Weak Alignment]:</p> <ul style="list-style-type: none"> The model provides very little relevant information. Most of the output (if reasoning is present) focuses on why the task is difficult. The final answer is vague or suggests "missing information" without explicitly flagging a specific table error. <p>0 [Explicit Mismatch / Corrective Refusal]:</p> <ul style="list-style-type: none"> The model explicitly identifies that the table contains errors, contradictions, or logical flaws. It refuses to provide the requested data in order to maintain factual integrity.
<p># Output:</p> <p>Return structured scores only. No explanations.</p> <p># Question: {Question}</p> <p># Model output: {Model output}</p>

Table 20: The prompt of **LLM-as-a-Judge** in Strict Anomaly Perception.

Prompt for Anomaly Injection

Role play

You are an expert in table analysis and data manipulation. Your role is to act as an "adversary" to strategically introduce plausible inconsistencies into a dataset.

Core Task

You will be provided with an Original Table and a Relevant Question. Your task is to create a new, modified version of the table by injecting inconsistencies that are directly relevant to the questions. The goal is to ensure that a system answering these questions based on the modified table would produce incorrect or contradictory results.

Generation Guidelines & Constraints

1. Minimal & Impactful Change:

- **Modify the minimum number of data points necessary.** The ideal modification is a single change that triggers one or more inconsistencies.
- **Detectable Inconsistency:** The introduced error must create a conflict that can be detected through **consistency checks** (e.g., a value contradicts another value in the table, a summary statistic doesn't match its components, a logical rule is broken).

2. Error Types: The single inconsistency you introduce must be from the following categories:

- Calculation: Errors in formulas or derived values (e.g., sum, ratio).
- Statistics: Incorrect statistical summaries (e.g., rank, count, max/min value).
- Logical Conflict: Data that violates a stated or implicit business logical rule.
- Row/Col Head Name Error: Mistakes in row or column headers.
- Row/Col Duplicate: Unjustified duplication of rows or columns.
- Knowledge Conflict: Contrary to common sense and laws of reality.

3. Focus-Driven Modification: The modification must be concentrated on the specific rows, columns, and data points referenced by the Relevant Question.

4. Plausibility: The modified data must remain within a realistic and reasonable range. Avoid obviously absurd values.

Output Control

You must output your modification exclusively in the following structured format. Typically, only one set of changes should be listed.

[ModifiedValue]: [The new value you are inserting into the cell],

[Location]: [Row X, Col Y(X and Y are integers starting from 1,X=1 refers to the first row(including its header))],

[ErrorType]: [One of: Calculation, Statistics, Rule Conflict, NULL, Row/Col Head Name Error, Row/Col Duplicate].

Let's get start!

Original Table

{Table}

Question

{Question}

Table 21: The prompt of **Anomaly Injection**.

Anomaly Perception Failure Example: Llama3.1-8B-Instruct — Table with Calculation Error

User:
{{ Anomaly_Perception_Prompt }}
Table:

MILITARY-TABLE07-1

UK Regular Forces Intake
Table 7 Intake to UK Regular Forces by Ethnic Origin
Source: Defence Statistics (Tri Service)

	31Mar12	30Sep12	31Mar13	30Sep13	31Mar14	30Sep14	31Mar15	30Sep15	31Mar16	30Sep16	
12 Months Ending:	Number	%	Number	%	Number	%	Number	%	Number	%	Number
All Services	14 790	14 520	14 370	13 390	11 880	12 050	12 980	13 580	13 790	13 640	
"Black, Asian and Minority Ethnic"	1 040	7.3	1 100	7.8	1 110	7.9	980	7.5	800	7 710	6 720 5.4 13 700 5.2 720 5.4
of which											
Mixed	235	230	230	190	170	205	245	250	255	260	
Asian	230	280	295	280	245	265	325	340	310	310	
Black	540	570	550	480	370	210	130	100	100	110	
Other	35	25	35	25	25	20	30	35	30		
White	13 290	92.7	13 110	92.2	13 000	92.1	12 000	92.5	10 650	93 11 070 94 12 040 94.3 12 600 94.6 12 770 94.8 12 580 94.6	
Unknown	460	*	310	*	260	*	410	*	270	*	220 260 320 340 *
Officers	1 060	960	1 060	1 050	1 060	1 100	1 130	1 210	1 210	1 350	
"Black, Asian and Minority Ethnic"	20	2.1	30	3.3	30	3.1	30	2.6	30	2.9 30 2.8 30 2.9 40 3.8 40 3.9 40 3.4	

Question:

Compare the total intake of “White” personnel and “Black, Asian and Minority Ethnic” personnel in March 2016. Which group had a higher intake?

Assistant:

Reasoning steps for this question are as follows:

1. Identify the total intake of “White” personnel in March 2016 from the table.
2. Identify the total intake of “Black, Asian and Minority Ethnic” personnel in March 2016 from the table.
3. Compare the two values to determine which group had a higher intake.

Final Answer: White, Black, Asian and Minority Ethnic

Figure 7: Anomaly Perception Failure Example 1 — Overlooking the Calculation Error.

Anomaly Perception Failure Example: TableGPT2-7B — Table with Duplication Error

User:

{{ Anomaly_Perception_Prompt }}
Table:

BUSINESS-TABLE05-2

1. Total
"Source: UK overseas trade in goods statistics from HM Revenue & Customs, and Inter-Departmental Business Register from the Office for National Statistics"
This worksheet contains one table.
Flow,Year,Country,Zone,Business size (No. of employees),Age (Years),Industry group,Value (),Business count,Employee count
...
Import,2023,United States,Non-EU,All,All,All,63309062535,45060,7807691
Import,2023,Vietnam,Non-EU,All,All,All,5102243738,5453,2011118
Export,2023,Belgium,EU,All,All,All,20295080962,16161,3134804
Export,2023,Czechia,EU,All,All,All,2201088454,9820,2492345
Export,2023,Denmark,EU,All,All,All,2671466023,12795,2834630
Export,2023,France,EU,All,All,All,23872217810,30504,5392871
Export,2023,Germany,EU,All,All,All,32432415283,33492,5914194
Export,2023,Ireland,EU,All,All,All,32432415283,33492,5914194
Export,2023,Italy,EU,All,All,All,9845745493,21117,4277109
Export,2023,Netherlands,EU,All,All,All,30035139626,30229,5956498
Export,2023,Poland,EU,All,All,All,6666197818,15060,3446051
Export,2023,Spain,EU,All,All,All,9480216468,18894,4437263
Export,2023,Sweden,EU,All,All,All,4848027866,13817,2929468
...

Question:

Which country ranks highest in trade value (£) for exports in 2023 within the EU zone?

Assistant:

Reasoning steps for this question are as follows:

1. Identify the relevant data for exports in 2023 within the EU zone.
2. Compare the trade values (£) for each country within the EU zone.
3. Determine the country with the highest trade value (£) for exports.

[Answer Format] Final Answer: Germany

Figure 8: Anomaly Perception Failure Example 2 — Overlooking the Duplication Error.

Anomaly Perception Success Example: Qwen3-14B-thinking — Table with Statistics Error

 **User:**

{{ Anomaly_Perception_Prompt }}

Table:

SOCIETY-TABLE78-02

"Table 2. The pooled age-standardized musculoskeletal disorders mortality rates (95% CI) per million person-years and women to men rate ratio (95% CI), 1986-2011"

Region	Men	Women	Women to men rate ratio
	rate	Lower bound	Upper bound
Asia	15.98	15.85	16.12
Hong Kong SAR	8.86	8.25	9.47
Israel	19.8	18.81	20.78
Japan	13.56	13.42	13.71
Kazakhstan	9.51	9.02	10.13
Kyrgyzstan	18.61	17.02	20.2
Republic of Korea	35.3	34.65	35.95
Singapore	7.34	6.53	8.15
Eastern Europe	8.75	8.65	8.84
Bulgaria	3.65	3.31	3.99
Czech Republic	3.34	3.06	3.62
Hungary	15.32	14.72	15.92
Poland	8.75	8.5	9.01

Question:

What is the women to men rate ratio for Japan?

 **Assistant:**

<think>

Okay, let's see. The user is asking for the women to men rate ratio for Japan from the given table. First, I need to locate the row corresponding to Japan in the table.

Looking at the table structure, the regions are listed under the "Region" column. Scrolling through the entries, I find "Japan" listed with the following data:

Under the "Women to men rate ratio" section, the Value is 1.29, with lower bound 1.57 and upper bound 1.61. Wait, that doesn't make sense. The rate ratio should be the value, and the lower and upper bounds are the confidence intervals. But the value here is 1.29, and the lower and upper bounds are 1.57 and 1.61? That seems contradictory because the lower bound should be less than the upper bound, but 1.57 is higher than 1.29. Wait, maybe I misread the columns.

Wait, looking back at the table headers: the columns for Women to men rate ratio are "Value", "Lower bound", "Upper bound". So for Japan, the Value is 1.29, Lower bound is 1.57, Upper bound is 1.61. Wait, that can't be right. The lower bound should be less than the value, and the upper bound higher. But here, the lower bound is higher than the value. That's a contradiction. Maybe there's a typo in the table? Or perhaps I'm misaligning the columns.

Wait, let me check the headers again. The table has columns for "Women to men rate ratio" with sub-columns: Value, Lower bound, Upper bound. So for each region, the women to men rate ratio's value is in the "Value" column, and the confidence interval bounds are in the next two columns.

Looking at Japan's row: the "Women to men rate ratio" Value is 1.29, Lower bound 1.57, Upper bound 1.61. That's impossible because the lower bound should be less than the value. This inconsistency suggests there might be an error in the table data. However, the user's instruction says if the table data contains errors or contradictions, I must output "Refuse to answer".

But wait, maybe I made a mistake in reading the numbers. Let me check again. The Japan row under "Women to men rate ratio" has Value 1.29, Lower bound 1.57, Upper bound 1.61. That's definitely a contradiction because the lower bound is higher than the value. This is an error in the table. Therefore, according to the exception handling rules, if the table data has errors, I should refuse to answer.

But maybe the user expects me to proceed despite the error? However, the strict rule says if there's an error, refuse.

Alternatively, maybe the numbers are in the wrong columns. For example, maybe the lower and upper bounds are swapped? If the lower bound was 1.25 and upper 1.33, that would make sense. But as per the given table, the numbers are as stated.

Therefore, since the data is contradictory, the correct response is to refuse to answer.

</think>

Reasoning steps for this question are as follows:

1. Locate the row for Japan in the table.
2. Identify the "Women to men rate ratio" section.
3. Check the "Value" column for Japan's rate ratio.
4. Notice that the "Lower bound" (1.57) is greater than the "Value" (1.29), which is a contradiction.
5. Determine that this inconsistency indicates an error in the table data.
6. Apply exception handling rules for data errors.

Final Answer: Refuse to answer

Figure 9: Anomaly Perception Success Example — Checking the Statistics Error.

Anomaly Detection Failure Example: Gemma-3-27b-it — Table with Duplication Error

User:

{{ Anomaly_Detection_Prompt }}

Table:

INJURIES-TABLE10-3

TABLE Q1. Incidence rates¹ of total recordable cases of nonfatal occupational injuries and illnesses by quartile distribution and employment size, 2019

Industry, NAICS code, 2 and establishment employment size, Average incidence rates for all establishments: , One-quarter of the establishments had a rate lower than or equal to: , One-half of the establishments had a rate lower than or equal to: , Three-fourths of the establishments had a rate lower than or equal to: , (mean), (1st quartile), (median), (3rd quartile)

All industries including state and local government³

...

Agriculture, forestry, fishing and hunting³

Total all sizes, 5.2, (-8-), (-8-), 3.4

1 - 10, 2.0, (-8-), (-8-), (-8-)

11 - 49, 4.5, (-8-), (-8-), 6.3

50 - 249, 6.0, 1.4, 3.9, 8.6

250 - 999, 5.6, 2.5, 5.2, 8.8

1,000+, 5.2, 3.1, 5.0, 6.5

Crop production (NAICS 111000)³

Total all sizes, 5.3, (-8-), 1.7, 7.6

11 - 49, 4.5, (-8-), (-8-), 7.4

50 - 249, 5.7, 1.1, 4.0, 8.3

250 - 999, 5.6, 2.5, 4.4, 7.4

1,000+, 5.8, -, -, -

Machinery manufacturing (NAICS 333000)

Total all sizes, 3.0, (-8-), (-8-), 3.0

11 - 49, 3.3, (-8-), (-8-), 5.4

50 - 249, 3.7, 1.2, 2.9, 5.2

250 - 999, 2.6, 1.0, 2.2, 3.9

1,000+, 2.2, 0.7, 1.4, 3.3

Sports teams and clubs (NAICS 711211)

1 - 10, -, -, -, -

50 - 249, 9.1, (-8-), 1.7, 6.7

250 - 999, 25.3, 2.1, 6.6, 28.0

1,000+, 5.2, -, -, -

Public administration

Total all sizes, 6.2, (-8-), (-8-), 3.7

11 - 49, 4.0, (-8-), (-8-), 5.6

50 - 249, 5.7, 1.3, 4.1, 8.4

250 - 999, 6.5, 3.1, 5.5, 9.3

1,000+, 7.7, 4.1, 6.3, 8.5

Machinery manufacturing (NAICS 333000)

Total all sizes, 6.2, (-8-), (-8-), 3.7

11 - 49, 4.0, (-8-), (-8-), 4.6

50 - 249, 5.7, 1.3, 4.1, 8.4

250 - 999, 6.5, 3.1, 5.5, 9.3

1,000+, 7.7, 4.1, 6.3, 8.5

...

Question:

Within the Machinery Manufacturing (NAICS 333000) industry, identify if the 3rd quartile incidence rate for the 11-49 employment size category exceeds 5.0. The answer is of type Yes/No.

Assistant:

Reasoning steps for this question are as follows:

1. The Target Data consists of the row corresponding to "Machinery Manufacturing (NAICS 333000)" and the column corresponding to the "11 - 49" employment size category, specifically the "3rd quartile" value.
2. The Target Data is: Industry = "Machinery Manufacturing (NAICS 333000)", Employment Size = "11 - 49", 3rd Quartile = "5.4".
3. I am verifying if the 3rd quartile value (5.4) is greater than 5.0.
4. Since $5.4 > 5.0$, the condition is met.
5. There are no apparent calculation, statistical, or logical errors within this Target Data set. The data appears internally consistent.

[Answer Format] Final Answer: [Correct]

Figure 10: Anomaly Detection Failure Example 1 — Overlooking the Duplication Error.

Anomaly Detection Failure Example: Qwen3-32B — Table with Statistics Error

User:
{{ Anomaly_Detection_Prompt }}
Table:

ECONOMY-TABLE50-1

Amenity Type	Tower 1	Tower 2	Tower 3	Tower 4	Tower 5	Tower 6	Tower 7	Tower 8	Tower 9	Total	% of Total
Luxury Amenities											
Indoor Pool										0	0%
golf simulator	1	1	1	1						4	44%
pet spa	1	1	1	1	1					5	56%
sky lounge	1	1	1							4	33%
Outdoor BBQ areas	1	1	1	1	1	1	1			7	78%
Outdoor Pool	1	1	1	1						3	33%
firepit	1	1	1							3	33%
dog park	1		1	1	1	1				5	56%
sport court	1									1	11%
...											

Question:

Compare the availability of the "pet spa" and the "sky lounge" in towers. Which has more towers equipped with it, and by how many?

Assistant:

Reasoning steps for this question are as follows:

- Define the Target Data:
 - The Target Data includes the "pet spa" and "sky lounge" rows, specifically the values in the "Total" column for each.
 - The "Total" column represents the total number of towers equipped with each amenity.
 - The question asks to compare the availability of these two amenities and determine the difference in the number of towers equipped with them.
- Verify Internal Consistency:
 - For "pet spa", the "Total" is 5.0.
 - For "sky lounge", the "Total" is 4.0.
 - The difference is $5.0 - 4.0 = 1.0$.
 - The "pet spa" is available in 1 more tower than the "sky lounge".
- Check for Error Types:
 - Calculation Error:** The "Total" values for both amenities are consistent with the sum of the 1.0s in the respective rows.
 - Statistics Error:** No statistics are being misreported.
 - Logical Rule Conflict:** No logical inconsistencies are present.
 - Row/Col Head Name Error:** The column headers and data types are consistent.
 - Row/Col Duplicate:** No duplicate rows or columns are present in the Target Data.
- Conclusion:
 - The Target Data is internally consistent and valid.
 - The "pet spa" is available in 1 more tower than the "sky lounge".
 - No error types from the Taxonomy are found.

Final Answer: [Correct]

Figure 11: Anomaly Detection Failure Example 2 — Overlooking the Statistics Error.

Anomaly Detection Success Example: GPT-5 — Table with Calculation Error

 **User:**

{{ Anomaly_Detection_Prompt }}
Table:

```
EMPLOYMENT-TABLE01-1

HOUSEHOLD DATA ANNUAL AVERAGES 1. Employment status of the civilian noninstitutional population, 1953 to date
[Numbers in thousands]
Year,Civiliannoninstitu-tional population,Civilian labor force,,,,,Not in labor force
,,Total,Percent of population,Employed,,,,Unemployed,,
,,,,Total,Percent of population,Agriculture,Nonagricultural industries,Number,Percent of labor force
Persons 16 years of age and over
1953,107056,63015,58.9,61179,57.1,6260,54919,1834,2.9,44041
1954,108321,63643,58.8,60109,55.5,6205,53904,3532,5.5,44678
1955,109683,65023,59.3,62170,56.7,6450,55722,2852,4.4,44660
1956,110954,66552,60.6,63799,57.5,6283,57514,2750,4.1,44402
1957,112265,66929,59.6,64071,57.1,6500,58123,2859,4.3,45336
1958,113727,67639,59.5,63036,55.4,5586,57450,4602,6.8,46088
1959,115329,68369,59.3,64630,56.5,5565,59065,3740,5.5,46960
1960,117245,69628,59.4,65778,56.1,5458,60318,3852,5.5,47617
1961,118771,70459,59.3,65746,55.4,5200,60546,4714,6.7,48312
1962,120153,70614,58.8,66702,55.5,4944,61759,3911,5.5,49539
1963,122416,71833,58.7,67762,55.4,4687,63076,4070,5.7,50583
1964,124485,73091,58.7,69305,55.7,4523,64782,3786,5.2,51394
...
```

Question:

Match the year where the employment in agriculture was the highest and state its corresponding total employed population in the year.

 **Assistant:**

Reasoning steps for this question are as follows:

1. Define the Target Data.
2. Parse the table structure and identify the relevant columns.
3. Determine the year with the maximum value in the “Employed: Agriculture” column.
4. Check internal consistency for that row and, more broadly, validate the intrinsic arithmetic relationship in the Employed breakdown (Agriculture + Nonagricultural industries should equal Employed Total).
5. Conclude whether any Taxonomy error exists within the Target Data.

The Target Data includes:

1. Conditions/Filters: All years’ “Employed: Agriculture” values to locate the maximum.
2. Operands: The “Employed: Agriculture,” “Employed: Nonagricultural industries,” and “Employed: Total” values for the relevant rows (particularly the row with the maximum agriculture value).
3. Result Cells: The “Employed: Total” for the year with the maximum agricultural employment.

I am verifying if: The year with the maximum “Employed: Agriculture” is correctly identified.
Employed Total equals Agriculture + Nonagricultural industries for the involved rows, especially the max-agriculture year.

Findings: The maximum “Employed: Agriculture” appears as 6500 (thousands) in 1957. For 1957, Employed Total is 64071, Agriculture is 6500, and Nonagricultural industries is 58123.

Check: $6500 + 58123 = 64623$, which does not equal the Employed Total of 64071. The discrepancy is 552 (thousands), far beyond minor rounding and thus an intrinsic Calculation Error.

By contrast, many nearby years either match exactly or show only small rounding differences (e.g., 1953: $6260 + 54919 = 61179$ matches Employed Total; 1956 differs by 2, which could be rounding). The 1957 mismatch is large and inconsistent with the table’s logic.

Result: A Calculation Error is present in the Target Data (Employed breakdown for 1957), which is part of the dataset required to determine the maximum agriculture year and its corresponding total employed.

[Answer Format] Final Answer: [Wrong]

Figure 12: Anomaly Detection Success Example — Checking the Calculation Error.