

Holmes-SQL: Benchmarking LLM Agents on Evidence-Aware Investigative Reasoning over Structured Databases

Hoang-An Vo¹, Cam-Tu Nguyen^{2,3}

¹University of Information Technology, VNU-HCM, Ho Chi Minh City, Vietnam.

²State Key Laboratory for Novel Software Technology, Nanjing University, China.

³School of Artificial Intelligence, Nanjing University, China.

21520555@gm.uit.edu.vn; ncamtu@nju.edu.cn

Abstract

Existing SQL-based benchmarks usually assume that retrieved records are equally informative. This assumption is limiting for investigative domains, where truthful records may differ substantially in inferential strength. To address this gap, we introduce Holmes-SQL, the first benchmark for urban crime investigation over relational databases. Each case begins with partial cues and requires agents to reconstruct the crime scene, victim, suspect pool, and culprit through database interaction. Holmes-SQL combines state uncertainty, unequal evidential strength, and fallacy-aligned suspect construction to evaluate evidence-aware investigative reasoning. Experiments show that agents achieve over 97% SQL execution success, yet the best solve rate reaches only 49%, highlighting a distinct challenge for evidence-aware investigative reasoning. A further comparison shows that agents consistently underperform when investigating compared with reasoning over all evidence provided upfront, highlighting the added challenge of investigation.

1 Introduction

SQL-based evaluation over structured databases has progressed from text-to-SQL, which maps user questions to executable queries (Yu et al., 2018, 2019b,a), to database question answering, where SQL supports analytical reasoning such as aggregation, time-series analysis, and scenario comparison (Zheng et al., 2024; Maddela et al., 2025), and more recently to agentic SQL benchmarks involving schema exploration, debugging, clarification, and tool-based interaction (Lei et al., 2025; Team, 2025; Li et al., 2026). These benchmarks have substantially improved the realism of database evaluation. Yet they usually assume that relevant records are equally significant.

This assumption is limiting in domains such as fraud analysis and criminal investigation, where

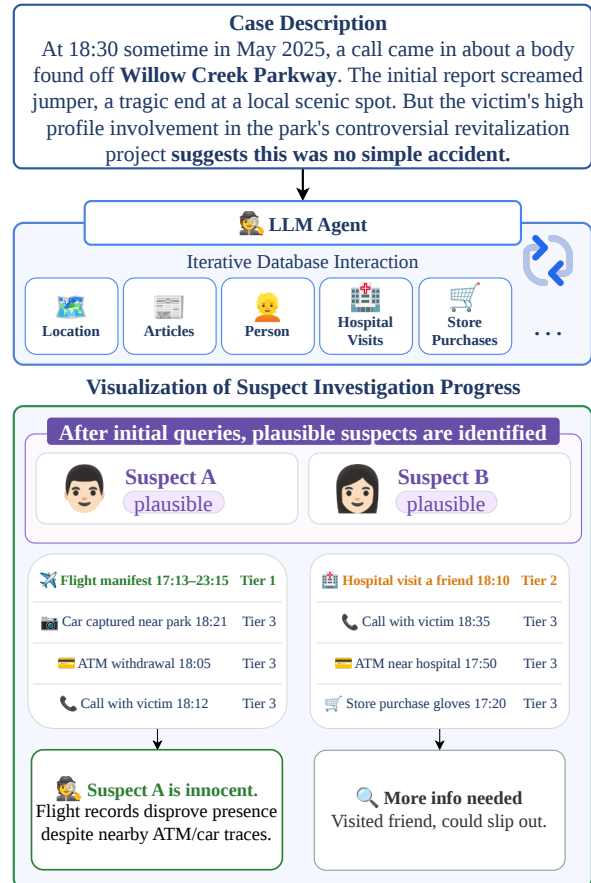


Figure 1: Holmes-SQL investigation under state uncertainty and unequal evidential strength. Partial initial clues require database interaction to recover the investigation state, while records range from decisive alibi evidence to weaker conditional or circumstantial traces.

records may be reliable as observations but ambiguous in interpretation. Adaptive adversaries can shape observable behavior to resemble benign activity, causing detectors to miss malicious cases or make incorrect decisions (Carminati et al., 2020; Lunghi et al., 2023). However, these works mainly study machine-learning detectors rather than agentic systems and therefore leave evidence-aware investigation largely unexplored.

Table 1: Comparison with existing SQL-based and investigation benchmarks. ✓ denotes properties explicitly operationalized by the benchmark design, while ✗ denotes properties outside the benchmark’s core target.

Benchmark	Task Type	Domain	State Uncertainty	Unequal Evidential Strength	Data Provenance
Spider (Yu et al., 2018)	Semantic Parsing	Multi-domain	✗	✗	Human
CoSQL (Yu et al., 2019a)	Conversational Semantic Parsing	Multi-domain	✓	✗	Human
STARQA (Maddela et al., 2025)	Question Answering	Multi-domain	✗	✗	Human
Spider 2.0 (Lei et al., 2025)	Workflow Text-to-SQL	Multi-domain	✓	✗	Human
BIRD-INTERACT (Team, 2025)	Interactive Workflow Text-to-SQL	Multi-domain	✓	✗	Mixed
SWE-SQL (Li et al., 2026)	SQL Debugging Workflow	Multi-domain	✓	✗	Human
ExCyTIn-Bench (Wu et al., 2025)	Investigation	Cybersecurity	✓	✗	Mixed
Holmes-SQL (ours)	Investigation	Urban Crime	✓	✓	Synthetic

To bridge this gap, we propose Holmes-SQL¹ (Fig. 1), a new benchmark for evidence-aware investigation over relational databases. We focus on urban crime because advances in IoT-enabled urban infrastructure create opportunities for AI agents to support investigative workflows over fragmented digital traces (Skipanes et al., 2025; Sharma et al., 2025). Urban investigations also involve heterogeneous evidence sources (Li et al., 2025) that can be naturally structured as relational databases, making them well suited for evaluating database exploration, evidence weighting, hypothesis revision, and suspect disambiguation under uncertainty.

Holmes-SQL is built around two construction properties designed to support evidence-aware investigative reasoning. First, each case begins under **state uncertainty**: the initial narrative provides only partial cues, requiring agents to uncover the crime scene, victim, and suspects through database interaction. Second, evidence has **unequal evidential strength**: different tables impose different levels of constraint, from strong spatiotemporal evidence to weaker circumstantial traces. Together, these properties expose **evidence-aware investigative reasoning**: the ability to conduct investigations by gathering evidence, assessing its inferential strength, and adapting actions under uncertainty. We also distinguish this new investigation-oriented capability from pure reasoning in § 4.2, showing substantial performance drops from the reasoning task to the investigation task.

This paper addresses three research questions. **RQ1.** Does reliable SQL execution translate into successful investigation? **RQ2.** To what extent do agents falsely accuse innocent fallacy-aligned suspects? **RQ3.** How well do agents collect relevant evidence throughout the investigation?

Our experiments reveal three key findings. First, reliable SQL execution does not translate into a successful investigation. Second, even after con-

trolling for victim and suspect recovery, agents still struggle with evidence-aware investigative reasoning. Third, a successful investigation requires not only sufficient evidence acquisition but also the ability to weigh and integrate evidence. The main contributions of this paper are as follows:

- To the best of our knowledge, Holmes-SQL is the first benchmark for evidence-aware urban crime investigation over structured databases.
- We propose a pipeline for constructing synthetic urban databases with spatiotemporal consistency and varying evidential strength.
- We conduct experiments on the benchmark to provide insights for future directions.

2 Related Work

Tab. 1 summarizes how Holmes-SQL compares with existing SQL-based and investigation benchmarks.

LLMs in Text-to-SQL and Database Interaction.

Research on LLMs over relational databases has shifted from single-shot semantic parsing to richer interactive workflows. Early benchmarks such as Spider (Yu et al., 2018) tested generalization to unseen schemas and compositional SQL, while CoSQL (Yu et al., 2019a) introduced multi-turn database interaction. Recent benchmarks further emphasize analytical reasoning, schema exploration, iterative correction, debugging, and tool use in realistic SQL workflows, including STARQA (Maddela et al., 2025), Spider 2.0 (Lei et al., 2025), BIRD-INTERACT (Team, 2025), and SWE-SQL (Li et al., 2026). However, these settings still largely treat database contents as information to retrieve rather than evidence to weigh against competing explanations.

¹<https://github.com/anpc849/Holmes-SQL>

LLMs in Investigation. A separate line of work studies LLMs in investigative settings, particularly cybersecurity and detective-style reasoning. CTIBench (Alam et al., 2024) evaluates cyber threat intelligence reasoning, while ExCyTInBench (Wu et al., 2025) assesses multi-hop cyber investigation over security logs and investigation graphs. Beyond cybersecurity, AR-Bench (Zhou et al., 2025) studies active reasoning under incomplete information, including detective cases where models ask questions to acquire missing evidence, and WhodunitBench (Xie et al., 2024) evaluates culprit identification and multi-step inference in narrative settings. These benchmarks demonstrate growing interest in investigative reasoning, but none explicitly target evidence-aware investigation over relational databases where agents must weigh truthful records with heterogeneous evidential strength against plausible, misleading traces.

3 Holmes-SQL Benchmark

This section introduces the Holmes-SQL benchmark, covering the task definition, dataset construction, evaluation metrics, quality control, data statistics, and environment setup. Additional details are provided in Appendix A.

3.1 Task Overview

We formulate crime investigation as sequential decision-making over a relational database $\mathcal{D}$. Given a case description x and a schema $\mathcal{S}$, an agent issues SQL queries over multiple steps. At step t , it generates a query $q_t \sim \pi_\theta(\cdot \mid x, \mathcal{S}, \mathcal{H}_{t-1})$, receives an observation $o_t = \text{Exec}(q_t, \mathcal{D})$, and updates its history as $\mathcal{H}_t = \mathcal{H}_{t-1} \oplus \{(q_t, o_t)\}$. After T steps, it outputs $\hat{y} \sim \pi_\theta(\cdot \mid x, \mathcal{S}, \mathcal{H}_T)$, where $\hat{y}$ is the predicted murderer and the objective is $\hat{y} = y^*$.

3.2 Task Setting

Evidence Tiers. All records in Holmes-SQL are error-free and truthful at the observation level, but differ systematically in inferential strength. We formalize this as a three-tier evidential hierarchy to ensure a controlled evaluation environment.

- **Tier 1** comprises identity-linked spatiotemporal constraints, instantiated by mobility records that bind individuals to specific travel events and can strongly exclude physical presence elsewhere during the crime window.
- **Tier 2** comprises direct but context-dependent institutional presence records, which establish

recorded location without necessarily implying incapacitation or complete exclusion.

- **Tier 3** comprises indirect, asset-mediated behavioral traces, which provide contextual support, investigative leads, or plausible distractions but should not supersede stronger constraints without corroboration.

Orthogonal demographic or biometric attributes constrain physical feasibility when case mechanics depend on bodily properties such as age or stature.

Fallacy Integration. To diversify the interaction between evidential strength and relational evidence sources across cases, Holmes-SQL combines the evidential hierarchy with logical-fallacy templates. A logical fallacy (Tindale, 2007) is a superficially persuasive but invalid reasoning pattern that can distort inference. In Holmes-SQL, each fallacy template specifies how truthful records can support an initially plausible but ultimately misleading suspect hypothesis. For example, a suspect may appear suspicious because their registered vehicle is captured by a traffic camera near the crime scene, and their bank card is used at a nearby ATM shortly before the murder. These Tier-3 behavioral traces create a plausible “nearby, therefore guilty” inference. However, stronger evidence, such as a Tier-1 flight manifest showing that the suspect was traveling during the crime window, can invalidate that hypothesis. This design increases case diversity by systematically varying the combinations of evidential strength, relational evidence sources, and fallacy-driven misleading hypotheses.

Suspect Design. Each case includes one victim and four independent suspects, exactly one of whom is the perpetrator. All suspects are assigned plausible relationship-based motives, thereby avoiding stranger-perpetrator scenarios and ensuring that suspicion cannot be resolved by motive alone. Suspect profiles are generated under controlled constraints to avoid introducing demographic biases related to race, gender, or religion. The suspects share the case-level fallacy type, but instantiate it through distinct evidential configurations and relational-table interactions. This design enables the evaluation of investigative agents within a controlled environment that systematically varies evidential strength and relational complexity.

Investigation Protocol. Each investigation is organized into three stages. The crime stage requires

the agent to recover the relevant event from narrative cues and extract crime scene information. The victim stage requires linking these descriptions to relational records to identify the victim and derive motive-bearing suspect candidates. The suspect stage requires a comparative assessment of candidates using activity records, alibi evidence, and motive evidence to determine the culprit. Although described in three phases, the task remains an end-to-end investigation rather than a fixed workflow, allowing agents to revisit earlier phases as new evidence changes the interpretation of the case.

3.3 Dataset Construction

Fig. 2 illustrates the two-stage Holmes-SQL construction pipeline, consisting of structured script generation followed by database generation.

Script Generation. Real-world urban investigation data pose significant challenges for direct benchmark construction due to privacy sensitivities, institutional fragmentation, and standardization barriers (Breitinger and Jotterand, 2023). Inspired by the synthetic data generation methodology in AR-Bench (Zhou et al., 2025), we address these limitations by using an instruction-tuned LLM to generate structured scripts. Each script is seeded by a designated crime theme location and a distinct logical fallacy, which jointly guide the core case logic. Conditioned on this seed, the script specifies the crime scene, victim, suspects, motives, alibis, opportunities, and evidential dependencies. These narrative components are subsequently instantiated as relational database records.

Database Generation. We use a fixed benchmark-wide schema across all scripts to eliminate confounding from schema-specific semantic parsing variation. Once a script s is approved (see Algorithm 1), we instantiate it through GENERATEWORLD, which creates the city-scale entities required by the benchmark, including locations, facilities, person records, and other associated records. Given the generated world W , SIMACTIVITIES simulates daily activity records under mobility constraints. GENTEXTRECORDS then produces case-associated textual records, e.g., news articles. Finally, BUILDDDB materializes (W, A, E) into the relational database d , and COLLECTGTIDS uses d together with (W, A, E) to record the tuples corresponding to the scripted ground truth g , yielding the output pair (d, g) for downstream evaluation tasks.

Algorithm 1 Script-to-database generation.

Require: Case script s

Ensure: Database d and ground-truth IDs g

```

1: # Generate world entities
2:  $W \leftarrow \text{GENERATEWORLD}(s)$ 
3: # Simulate mobility-feasible activities
4:  $A \leftarrow \text{SIMACTIVITIES}(s, W)$ 
5: # Generate textual records
6:  $E \leftarrow \text{GENTEXTRECORDS}(s, W)$ 
7: # Build database
8:  $d \leftarrow \text{BUILDDDB}(W, A, E)$ 
9: # Collect ground-truth IDs
10:  $g \leftarrow \text{COLLECTGTIDS}(d, W, A, E)$ 
11: return  $(d, g)$ 

```

Artifact Generation. After database construction, we create three artifacts per case. (1) The case description serves as the starting point, placing the investigation under state uncertainty by providing partial incident cues while withholding suspect identities and evidential links required for resolution. (2) The checklist defines the evidence items that must be collected to resolve the case. (3) The reference SQL solution path specifies the ground-truth query trajectory for identifying the culprit. This reference path is used only to verify that each generated case has at least one solution; it is not used to evaluate agents, and agents are not penalized for following a different investigation path.

3.4 Evaluation Metrics

Solve Rate. This metric is the proportion of cases in which the agent correctly identifies the murderer ID, serving as the primary end-task metric.

ErrorQ and EmptyQ. ErrorQ measures the fraction of issued queries that fail due to syntax or execution errors, whereas EmptyQ measures the fraction of executed queries that return no rows; together, they separate database-interaction failures from downstream reasoning failures.

Recall@k. The metric is computed over the evidence items in the checklist and measures the fraction of those items that are explicitly or implicitly retrieved from $\mathcal{H}_k$, the history accumulated within the first k investigation steps.

False Accusation Rate (FAR). The metric measures whether an agent falsely accuses an innocent fallacy-aligned suspect in cases where the relevant investigative context has been established. A case

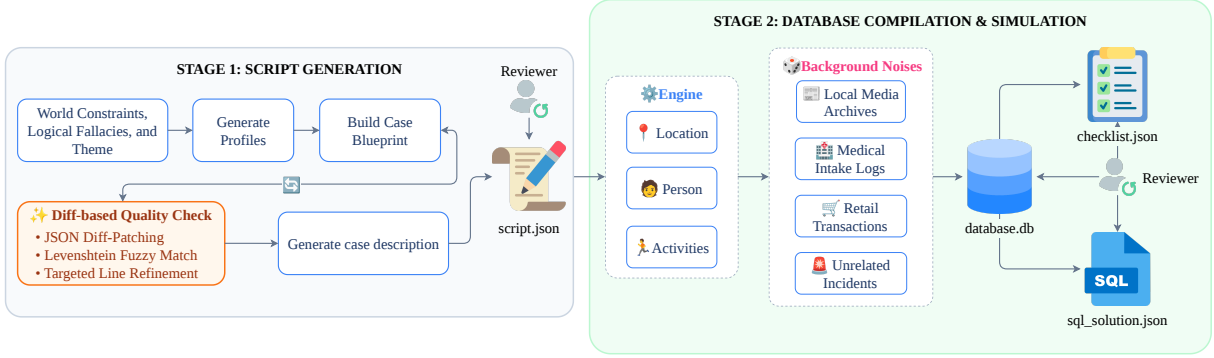


Figure 2: Illustration of the two-stage Holmes-SQL construction pipeline, comprising structured script generation and database generation.

is included in FAR when the agent’s structured output and investigation history $\mathcal{H}$ deterministically identify the correct report, the correct victim, and all four ground-truth suspects. Let C be the set of cases, let $A_c = 1$ indicate that case c is included in FAR and $A_c = 0$ otherwise, let $\hat{m}_c$ be the predicted murderer, and let $\mathcal{I}_c$ be the set of innocent fallacy-aligned suspects. FAR is defined as:

$$\text{FAR} = \frac{\sum_{c \in C} \mathbf{1}[A_c = 1 \wedge \hat{m}_c \in \mathcal{I}_c]}{\sum_{c \in C} \mathbf{1}[A_c = 1]}.$$

Cases that do not satisfy the inclusion criterion are excluded from FAR.

3.5 Quality Control

To ensure dataset quality, we use a two-stage framework that combines automatic refinement with manual review. In the script stage, we first use an LLM-powered diff-editor (Duanis et al., 2025) to revise generated scripts without regenerating them from scratch. A script is marked as valid only if it is successfully refined within a fixed number of attempts. Two guideline-trained fourth-year IT student annotators, who achieved substantial agreement ($\kappa = 0.85$) (Cohen, 1960), review these valid scripts. A valid script is accepted only after passing rigorous validation checks for logical fallacy integration, suspect role suitability, logical consistency, and solvability. Only 15.6% of scripts fail to meet these validation checks and are therefore manually refined by the annotators.

In the database stage, Algorithm 1 enforces logical constraints during database instantiation to preserve the accepted script properties. We then inspect the generated artifacts to ensure that case descriptions provide sufficient but partial cues to initiate the investigation without over-revealing the

culprit. We also verify that the generated checklists preserve all evidence from the accepted scripts, and that the SQL reference solutions execute without syntax errors and provide valid gold reasoning paths. Overall, 8.8% of cases contain artifacts that fail these checks and are manually refined.

3.6 Data Statistics

Holmes-SQL consists of 500 high-quality detective cases spanning five urban themes and ten logical-fallacy settings. Each case uses the same 17-table schema, requiring 14 tables and 15 evidence items on average to solve. Tier 1, Tier 2, and Tier 3 evidence account for 9.00%, 23.16%, and 67.84%, respectively, suggesting that agents cannot rely solely on decisive Tier 1 evidence and instead require strong evidence-aware investigative reasoning.

Evidence is also concentrated in the Suspect Stage (68.22%), compared with the Crime Stage (21.35%) and Victim Stage (10.43%), making suspect reasoning the most demanding phase. In addition, evidence is sparsely embedded in large databases, requiring selective filtering and relational composition. Reference SQL solutions reflect this complexity: JOIN appears in 73.56% of gold queries, while LIKE, IN, arithmetic computation, and BETWEEN appear less frequently.

Holmes-SQL further shows substantial within-case evidence-combination diversity: the average pairwise Jaccard distance (Jaccard, 1901) among the suspects’ evidence-type signatures is 0.747, indicating that suspects within the same case typically rely on different evidence configurations. Across cases, Holmes-SQL exhibits template-level diversity, with an average cross-case Jaccard distance of 0.898 over case-level evidence-combination sets, indicating that different cases need distinct investigative evidence templates.

Table 2: Evaluation of the baseline agent with different backbones. Parentheses in the FAR column report n , the number of cases used as the FAR denominator. **Bold** values denote the best results in each column.

Model	Solve Rate ($\uparrow$)	ErrorQ ($\downarrow$)	EmptyQ ($\downarrow$)	Recall@50 ($\uparrow$)	FAR ($\downarrow$)
<i>Proprietary models</i>					
gemini-3.1-pro-preview	0.4900	0.0033	0.1380	0.5169	0.3964 ($n = 169$)
gpt-5.4-2026-03-05	0.3940	0.0028	0.2776	0.4159	0.5733 ($n = 225$)
glm-5	0.3760	0.0075	0.1642	0.4393	0.5350 ($n = 200$)
gemini-3-flash-preview	0.3220	0.0136	0.1998	0.5166	0.6592 ($n = 179$)
<i>Open-source models</i>					
gemma-4-31b	0.4080	0.0085	0.1792	0.4654	0.5282 ($n = 195$)
deepseek-v3.2	0.3340	0.0142	0.2737	0.5068	0.5545 ($n = 220$)
qwen3-235b-a22b-instruct-2507	0.2080	0.0281	0.3460	0.3056	0.6330 ($n = 109$)
qwen3-coder-480b-a35b-instruct	0.1880	0.0163	0.4122	0.3007	0.6789 ($n = 109$)

3.7 Environment Setup

Following prior database-agent evaluations (Yang et al., 2023; Wu et al., 2025), each case is a standalone SQLite² database. The agent receives only the case description and schema, without example rows. By default, each SQL query returns at most 10 rows, times out after 60 seconds, and each investigation lasts at most 50 steps.

Since agents may transform rows, the returned rows may no longer match the ground truth exactly. We therefore use an LLM-as-a-Judge paradigm (Zheng et al., 2023) to determine how many evidence items have been retrieved at step k . Recall@ k is then computed at the case level as the fraction of evidence items retrieved for that case and averaged across cases. We use Gemini-2.5-Pro as the judge and manually check 100 randomly sampled cases per model, finding strong agreement with human annotations (Cohen’s $\kappa = 0.85 \pm 0.04$ across models). Further details are provided in Appendix B.3.

4 Benchmark Experiments

We conduct all experiments using the Kaggle Benchmarks framework³ for standardized model evaluation and auditability. All code and experiments are run in Kaggle Notebooks⁴ to support reproducibility in a controlled environment. We use deterministic decoding for all LLMs in all experiments by setting the temperature to 0.0. Further details on the experimental setup and additional result analyses are provided in Appendix B.

²<https://www.sqlite.org/>

³<https://github.com/Kaggle/kaggle-benchmarks>

⁴<https://www.kaggle.com/code>

4.1 Baseline Results

We use ReAct (Yao et al., 2023) as our baseline because it is the foundation for many advanced LLM-agent frameworks. We evaluate it with eight backbones across proprietary and open-source models (Appendix B.4), allowing us to assess whether Holmes-SQL remains challenging across diverse model families and design choices. Chain-of-thought (Wei et al., 2022) prompting is used as the default prompting strategy in our experiments. To mitigate long-context degradation (Liu et al., 2024), the agent is instructed every 10 steps to summarize its investigation progress and continue from the updated context. Tab. 2 reports the baseline results.

RQ1: Does reliable SQL execution translate into successful investigation? We examine the relationship between Solve Rate and ErrorQ. The metrics exhibit a statistically significant but small negative case-level Pearson correlation ($r = -0.0341$, $p = 0.0314$). This suggests that SQL execution errors are associated with lower success, but they explain only a negligible portion of case-level variation. At the aggregate level, ErrorQ remains below 3% across all backbones, whereas Solve Rate peaks at only 49% with Gemini-3.1-Pro. These results indicate that executable SQL is necessary but not sufficient for successful investigation.

RQ2: To what extent do agents falsely accuse innocent fallacy-aligned suspects? We compare FAR under two settings: full investigation and suspect-only investigation. In a full investigation, the model must investigate from the beginning, so its results may still be affected by context noise. To control for this effect, the suspect-only setting provides all clues up to the suspect names, so the

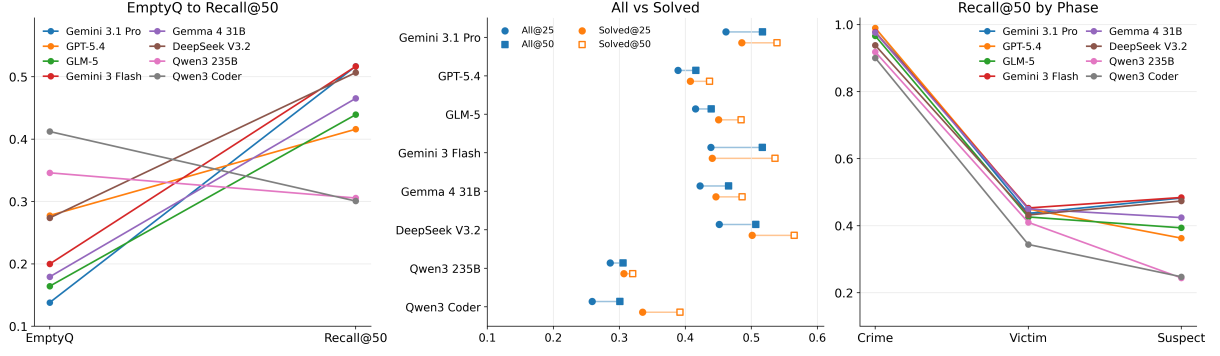


Figure 3: Left: EmptyQ versus Recall@50. Middle: Recall@25/50 for all and solved cases. Right: Recall@50 by evidence phase. Recall is averaged at the case level.

model only needs to collect evidence to accuse one of them. Across both settings (Tab. 3), FAR remains high: it ranges from 39.64% to 67.89% in full investigation, with an average of 56.98%, and from 38.35% to 70.12% in suspect-only investigation, with an average of 56.16%. Although isolating the suspect stage helps some models, the gains are inconsistent, suggesting that evidence-aware investigative reasoning remains a core bottleneck even with the correct suspect pool.

Table 3: FAR comparison between full investigation and suspect-only investigation. Values in parentheses report n , the number of cases used as the FAR denominator in the full-investigation setting. **Bold** values denote the best result in each column.

Model	FAR _{full} ($\downarrow$)	FAR _{suspect-only} ($\downarrow$)
<i>Proprietary models</i>		
gemini-3.1-pro-preview	0.3964 ($n = 169$)	0.3835
gpt-5.4-2026-03-05	0.5733 ($n = 225$)	0.5772
glm-5	0.5350 ($n = 200$)	0.5273
gemini-3-flash-preview	0.6592 ($n = 179$)	0.5526
<i>Open-source models</i>		
gemma-4-31b	0.5282 ($n = 195$)	0.5491
deepseek-v3.2	0.5545 ($n = 220$)	0.5386
qwen3-235b-a22b-instruct-2507	0.6330 ($n = 109$)	0.6635
qwen3-coder-480b-a35b-instruct	0.6789 ($n = 109$)	0.7012

RQ3: How well do agents collect relevant evidence throughout the investigation? We analyze EmptyQ and Recall@k to assess evidence collection. Tab. 2 and Fig. 3 show that EmptyQ alone is insufficient: similar empty-query rates can reflect either failed exploration or broader probing, as DeepSeek-V3.2 attains much higher Recall@50 than GPT-5.4 despite comparable EmptyQ. Evidence recovery efficiency also varies: GPT-5.4 and Qwen3-235B gain little from Recall@25 to Recall@50, indicating that more query budget does not automatically yield more relevant evidence.

Solved cases have higher Recall@k, but recall remains incomplete even when agents are correct, suggesting that final verdicts often rely on partial evidence. Recall@50 is highest in the crime stage and decreases in the victim and suspect stages, suggesting that evidence recovery becomes more difficult as the investigation progresses.

Model	Reasoning	Inv _{suspect-only}	Inv _{full}
<i>Proprietary models</i>			
gemini-3.1-pro-preview	0.7380	0.6150 _{-0.1230}	0.4900 _{-0.2480}
gpt-5.4-2026-03-05	0.6340	0.4220 _{-0.2120}	0.3940 _{-0.2400}
glm-5	0.6480	0.4680 _{-0.1800}	0.3760 _{-0.2720}
gemini-3-flash-preview	0.5960	0.3740 _{-0.2220}	0.3220 _{-0.2740}
<i>Open-source models</i>			
gemma-4-31b	0.6460	0.4220 _{-0.2240}	0.4080 _{-0.2380}
deepseek-v3.2	0.4460	0.4420 _{-0.0040}	0.3340 _{-0.1120}
qwen3-235b-a22b-instruct-2507	0.4540	0.2800 _{-0.1740}	0.2080 _{-0.2460}
qwen3-coder-480b-a35b-instruct	0.4540	0.2940 _{-0.1600}	0.1880 _{-0.2660}

Table 4: Solve Rate for reasoning and investigative tasks. Subscripts denote the performance drop from reasoning. **Bold** indicates the best result in each column.

4.2 From Reasoning to Investigation

We distinguish the reasoning task from the investigation task defined in § 3.1. In the reasoning setting, the model is provided with all necessary information and must identify the murderer in a single inference step. Tab. 4 shows a gap between reasoning and both investigation settings, with full investigation causing the largest drop. This indicates that evidence-aware investigative reasoning introduces additional difficulty beyond pure evidence reasoning, requiring models to discover and update relevant evidence before making the judgment. However, the reasoning task remains challenging: the best model reaches only 0.738, and several stay near 0.450, showing difficulty in integrating unevenly diagnostic records even when all relevant information is provided.

Table 5: Results for comparing different agent architecture on the gemma-4-31b backbone. Parentheses in the FAR column report n , the number of cases used as the FAR denominator. Cost(\$) denotes the average inference cost per case. **Bold** values indicate the best results in each column.

Method	Solve Rate ($\uparrow$)	ErrorQ ($\downarrow$)	EmptyQ ($\downarrow$)	Recall@50 ($\uparrow$)	FAR ($\downarrow$)	Avg. Cost ($\downarrow$)
Base+ReAct	0.3380	0.0042	0.1917	0.3766	0.5204($n = 98$)	0.0183
CoT+ReAct	0.4080	0.0085	0.1792	0.4654	0.5282($n = 195$)	0.0150
CoT+PlanAndExecute	0.4680	0.0117	0.1477	0.4930	0.4894 ($n = 423$)	0.0149
CoT+Reflexion	0.4440	0.0075	0.1911	0.4903	0.4975($n = 394$)	0.0123

4.3 Comparison of Different Methods

To disentangle the effect of agent architecture on investigative performance, we compare four variants using the Gemma-4-31B backbone, chosen for its strong open-source baseline performance in § 4.1. Tab. 5 shows that removing CoT from CoT+ReAct lowers Solve Rate from 0.4080 to 0.3380 and Recall@50 from 0.4654 to 0.3766, despite the lowest ErrorQ. Planning (Wang et al., 2023) and reflexion (Shinn et al., 2023) both improve over CoT+ReAct, with CoT+PlanAndExecute best on Solve Rate, EmptyQ, Recall@50, and FAR. Reflexion ranks second, possibly because it depends more on the model’s internal reasoning quality, whereas planning provides explicit guidance for evidence collection. Overall, these results suggest that advances in agent architectures can improve investigation efficiency while remaining cost-competitive.

4.4 Ablation on DB Scope and SQL Limit

Table 6 studies two factors that affect investigation difficulty.

Database Scope: We retain all ground-truth rows and vary the fraction of sampled background rows, where retention 0 contains only ground-truth rows and larger values add more contextual records. Qwen3-235B and Qwen3-Coder show monotonic drops in Solve Rate as retention increases, suggesting that additional background records can introduce misleading retrieval paths. Gemma-4-31B and DeepSeek-V3.2 instead show non-monotonic trends, which our manual inspection of 200 random investigations attributes to agentic instability: moderate background context can make an incorrect hypothesis appear sufficiently supported and trigger early stopping, while larger databases may expose alternative query paths that recover decisive evidence.

SQL Limit: Increasing the maximum number of returned rows per query generally benefits stronger models, indicating that greater retrieval bandwidth

Table 6: Ablation results for Solve Rate under database scope and SQL limit. **Bold** values indicate the best results in each column.

Model	Retention Ratio			SQL Limit		
	0	0.25	0.5	5	10	20
Gemma-4-31B	0.4580	0.2860	0.3140	0.3580	0.4080	0.4500
DeepSeek-V3.2	0.4240	0.3980	0.4300	0.3400	0.3340	0.4120
Qwen3-235B	0.2700	0.2220	0.1900	0.2120	0.2080	0.2580
Qwen3-Coder	0.2740	0.2500	0.2140	0.1620	0.1880	0.1580

can compensate for imperfect query precision. Weaker models improve less consistently, suggesting that additional evidence helps only when agents can interpret and integrate it effectively. This observation aligns with findings in long-context RAG (Jin et al., 2025), where increasing the number of retrieved passages can initially improve output quality but may later degrade it once excessive context is introduced.

5 Conclusion

In this paper, we introduce Holmes-SQL, the first benchmark for evaluating evidence-aware investigative reasoning over relational urban crime databases. We also present a scalable LLM-assisted pipeline for constructing urban databases with spatiotemporal consistency, as part of an effort to address the practical challenges of collecting real urban data. We evaluate various LLMs and agent systems on the benchmark. Our results show that evidence-aware investigative reasoning poses a new challenge for these systems. We hope Holmes-SQL will support future research toward LLM agents that can assist real-world investigations by gathering evidence, assessing its inferential strength, and adapting actions under uncertainty.

Limitations

While Holmes-SQL provides a new challenge, it is important to acknowledge certain limitations.

First, although the benchmark is designed to reflect realistic investigative structures, it is built from

synthetic cases within a controlled urban crime ontology. As a result, performance on Holmes-SQL may not fully reflect performance in real-world investigative scenarios, and its synthetic evidence hierarchy may not fully capture how humans assess and weigh evidence in practice. Nevertheless, we believe Holmes-SQL provides a useful testbed for evaluating investigative capabilities within a standardized framework.

Second, our benchmark focuses on single-perpetrator cases, which helps improve the yield of high-quality LLM-generated cases but may not fully capture the complexity of practical investigative contexts, such as coordinated crimes, evolving evidence streams, or long-horizon investigations.

Third, although LLMs and agent systems in our experiments generated explanations for their investigative decisions, our current evaluation framework does not assess the quality or faithfulness of these explanations. Future work could extend Holmes-SQL to support the evaluation of explainability in LLM-powered investigative systems.

Finally, Recall@k, like any LLM-based evaluation, has inherent limitations. Its reliability depends on the capabilities and potential biases of the chosen judge LLM, such as position or verbosity bias (Wang et al., 2024). Although our validation shows a strong correlation with human judgment, Recall@k remains a proxy metric rather than a replacement for human evaluation in safety-critical settings. Future work could use ensembles of judge LLMs or fine-tuned open-source evaluators to build a more robust and transparent scoring system.

Ethical Considerations

Holmes-SQL is a fully synthetic benchmark constructed solely for research purposes. All cases, narratives, people, organizations, locations, addresses, institutions, events, and investigation records are fictional, and any resemblance to real persons, places, organizations, incidents, or ongoing investigations is purely coincidental. The benchmark does not use or disclose personally identifiable information, private records, or real criminal case data. In addition, suspect profiles are generated under controlled constraints to reduce the risk of reinforcing discriminatory stereotypes related to race, gender, or religion.

Although Holmes-SQL uses realistic urban investigation scenarios, this realism is intended only to support controlled research evaluation. The

benchmark should not be treated as a source of factual, legal, medical, or law-enforcement guidance, nor as evidence that a system is suitable for real-world criminal investigations, policing decisions, or legal judgments. Crucially, given the high observed False Accusation Rate (FAR) in our evaluations, we strongly caution against deploying such LLM-based agents for real-world suspect triage or automated case screening.

In real-world investigative settings, evidence may be incomplete, contested, legally constrained, or harmful when interpreted without appropriate human oversight. Holmes-SQL should therefore be used only for research on evidence-aware investigative reasoning, with particular attention to false accusations, overconfident conclusions, and unsafe deployment in high-stakes contexts.

Acknowledgments

We thank our reviewers for their thoughtful feedback. Hoang-An Vo is supported by the Kaggle Research Grant Program. Cam-Tu Nguyen is supported by the National Natural Science Foundation of China (Grant No. W2532049). The views and opinions expressed in this material are those of the authors and do not necessarily reflect the views of any organization or agency that provided support for this work.

References

- Md Tanvirul Alam, Le Nguyen, Dipkamal Bhusal, and Nidhi Rastogi. 2024. Ctibench: a benchmark for evaluating llms in cyber threat intelligence. In *Proceedings of the 38th International Conference on Neural Information Processing Systems, NIPS '24*, Red Hook, NY, USA. Curran Associates Inc.
- Andrea Bari and Trevor Robbins. 2013. [Inhibition and impulsivity: Behavioral and neural basis of response control](#). *Progress in neurobiology*, 108.
- Frank Breitinger and Alexandre Jotterand. 2023. [Sharing datasets for digital forensic: A novel taxonomy and legal concerns](#). *Forensic Science International: Digital Investigation*, 45:301562.
- Michele Carminati, Luca Santini, Mario Polino, and Stefano Zanero. 2020. [Evasion attacks against banking fraud detection systems](#). In *23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020)*, pages 285–300, San Sebastian. USENIX Association.
- Jacob Cohen. 1960. [A coefficient of agreement for nominal scales](#). *Educational and Psychological Measurement*, 20:37 – 46.

- Sarel Duanis, Asnat Greenstein-Messica, and Eliya Habba. 2025. [JSON whisperer: Efficient JSON editing with LLMs](#). In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 1265–1274, Suzhou (China). Association for Computational Linguistics.
- Paul Jaccard. 1901. [Etude de la distribution florale dans une portion des alpes et du jura](#). *Bulletin de la Societe Vaudoise des Sciences Naturelles*, 37:547–579.
- Bowen Jin, Jinsung Yoon, Jiawei Han, and Sercan Arik. 2025. Long-context llms meet rag: Overcoming challenges for long inputs in rag. In *International Conference on Learning Representations*, volume 2025, pages 37784–37822.
- Fangyu Lei, Jixuan Chen, Yuxiao Ye, Ruisheng Cao, Dongchan Shin, Hongjin Su, Zhaoqing Suo, Hongcheng Gao, Wenjing Hu, Pengcheng Yin, and 1 others. 2025. Spider 2.0: Evaluating language models on real-world enterprise text-to-sql workflows. In *International Conference on Learning Representations*, volume 2025, pages 28691–28735.
- Jinyang Li, Xiaolong Li, Ge Qu, Per Jacobsson, Bowen Qin, Binyuan Hui, Shuzheng Si, Nan Huo, Xiaohan Xu, Yue Zhang, and 1 others. 2026. Swe-sql: Illuminating llm pathways to solve user sql issues in real-world applications. *Advances in Neural Information Processing Systems*, 38:97085–97120.
- Zhonghang Li, Lianghao Xia, Xubin Ren, Jiabin Tang, Tianyi Chen, Yong Xu, and Chao Huang. 2025. [Urban computing in the era of large language models](#). *ACM Trans. Intell. Syst. Technol.*, 16(6).
- Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. [Lost in the middle: How language models use long contexts](#). *Transactions of the Association for Computational Linguistics*, 12:157–173.
- Daniele Lunghi, Alkis Simitsis, Olivier Caelen, and Gianluca Bontempi. 2023. [Adversarial learning in real-world fraud detection: Challenges and perspectives](#). In *Proceedings of the Second ACM Data Economy Workshop*, DEC '23, page 27–33, New York, NY, USA. Association for Computing Machinery.
- Mounica Maddela, Lingjue Xie, Daniel Preotiuc-Pietro, and Mausam. 2025. [STARQA: A question answering dataset for complex analytical reasoning over structured databases](#). In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 34487–34499, Suzhou, China. Association for Computational Linguistics.
- Akira Miyake, Naomi P. Friedman, Michael J. Emerson, Alexander H. Witzki, Amy Howerter, and Tor D. Wager. 2000. [The unity and diversity of executive functions and their contributions to complex “frontal lobe” tasks: A latent variable analysis](#). *Cognitive Psychology*, 41(1):49–100.
- Binaya Sharma, James Ghawaly, Kyle McCleary, Andrew M. Webb, and Ibrahim Baggili. 2025. [Forensic-llm: A local large language model for digital forensics](#). *Forensic Science International: Digital Investigation*, 52:301872. DFRWS EU 2025 - Selected Papers from the 12th Annual Digital Forensics Research Conference Europe.
- Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. [Reflection: language agents with verbal reinforcement learning](#). In *Advances in Neural Information Processing Systems*, volume 36, pages 8634–8652. Curran Associates, Inc.
- Mads Skipanes, Tollef Emil Jørgensen, Kyle Porter, Gianluca Demartini, and Sule Yildirim Yayilgan. 2025. [Enhancing criminal investigation analysis with summarization and memory-based retrieval-augmented generation: A comprehensive evaluation of real case data](#). In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 4993–5010, Abu Dhabi, UAE. Association for Computational Linguistics.
- BIRD Team. 2025. Bird-interact: Re-imagine text-to-sql evaluation via lens of dynamic interactions. <https://github.com/bird-bench/BIRD-Interact>. Accessed: 2025-06-01.
- Christopher W. Tindale. 2007. *Fallacies and Argument Appraisal*. Critical Reasoning and Argumentation. Cambridge University Press.
- Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. 2023. [Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2609–2634, Toronto, Canada. Association for Computational Linguistics.
- Peiyi Wang, Lei Li, Liang Chen, Zefan Cai, Dawei Zhu, Binghuai Lin, Yunbo Cao, Lingpeng Kong, Qi Liu, Tianyu Liu, and Zhifang Sui. 2024. [Large language models are not fair evaluators](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9440–9450, Bangkok, Thailand. Association for Computational Linguistics.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed 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*, volume 35, pages 24824–24837. Curran Associates, Inc.
- Yiran Wu, Mauricio Velazco, Andrew Zhao, Manuel Raúl Meléndez Luján, Srisuma Movva, Yogesh K Roy, Quang Nguyen, Roberto Rodriguez, Qingyun Wu, Michael Albada, and 1 others. 2025. Excytin-bench: Evaluating llm agents on cyber threat investigation. *arXiv preprint arXiv:2507.14201*.

- Junlin Xie, Ruifei Zhang, Zhihong Chen, Xiang Wan, and Guanbin Li. 2024. [Whodunitbench: Evaluating large multimodal agents via murder mystery games](#). In *Advances in Neural Information Processing Systems*, volume 37, pages 86655–86687. Curran Associates, Inc.
- John Yang, Akshara Prabhakar, Karthik Narasimhan, and Shunyu Yao. 2023. [Intercode: Standardizing and benchmarking interactive coding with execution feedback](#). In *Advances in Neural Information Processing Systems*, volume 36, pages 23826–23854. Curran Associates, Inc.
- Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.
- Tao Yu, Rui Zhang, Heyang Er, Suyi Li, Eric Xue, Bo Pang, Xi Victoria Lin, Yi Chern Tan, Tianze Shi, Zihan Li, Youxuan Jiang, Michihiro Yasunaga, Sungrok Shim, Tao Chen, Alexander Fabbri, Zifan Li, Luyao Chen, Yuwen Zhang, Shreya Dixit, and 5 others. 2019a. [CoSQL: A conversational text-to-SQL challenge towards cross-domain natural language interfaces to databases](#). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1962–1979, Hong Kong, China. Association for Computational Linguistics.
- Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. 2018. [Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task](#). In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3911–3921, Brussels, Belgium. Association for Computational Linguistics.
- Tao Yu, Rui Zhang, Michihiro Yasunaga, Yi Chern Tan, Xi Victoria Lin, Suyi Li, Heyang Er, Irene Li, Bo Pang, Tao Chen, Emily Ji, Shreya Dixit, David Proctor, Sungrok Shim, Jonathan Kraft, Vincent Zhang, Caiming Xiong, Richard Socher, and Dragomir Radev. 2019b. [SPaRC: Cross-domain semantic parsing in context](#). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4511–4523, Florence, Italy. Association for Computational Linguistics.
- Danna Zheng, Mirella Lapata, and Jeff Pan. 2024. [Archer: A human-labeled text-to-SQL dataset with arithmetic, commonsense and hypothetical reasoning](#). In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 94–111, St. Julian’s, Malta. Association for Computational Linguistics.
- Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhaghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph Gonzalez, and Ion Stoica. 2023. [Judging llm-as-a-judge with mt-bench and chatbot arena](#). In *Advances in Neural Information Processing Systems*, volume 36, pages 46595–46623. Curran Associates, Inc.
- Zhanke Zhou, Xiao Feng, Zhaocheng Zhu, Jiangchao Yao, Sanmi Koyejo, and Bo Han. 2025. From passive to active reasoning: Can large language models ask the right questions under incomplete information? In *ICML*.

A Additional Details on the Dataset

A.1 Database Schema

This section summarizes the database schema used in Holmes-SQL, as shown in Tab. 7. The schema is organized around four types of information: locations, background context, people, and behavioral activities. Location tables define the setting of each case, including stores, companies, airports, hospitals, schools, parks, and ATM ports. Person-related tables describe suspects and other individuals through demographic, financial, employment, licensing, and vehicle records. Activity tables capture time-stamped behavioral evidence such as phone calls, purchases, hospital visits, flights, ATM transactions, and traffic camera logs.

A.2 Logical Fallacy

Tab. 8 summarizes the ten logical fallacies used in Holmes-SQL and explains how each is operationalized as an SQL-solvable reasoning pattern.

A.3 Dataset Sample

Each Holmes-SQL sample consists of four components: a case description, a relational database, a checklist, and a gold SQL investigation path. Fig. 4 shows an example from Holmes-SQL. The left contains the case description and checklist, while the right side shows the corresponding gold SQL path.

A.4 Evidence Strength Hierarchy

Holmes-SQL organizes evidence by inferential strength rather than treating all retrieved records as equally decisive. This hierarchy is used to diversify evidence combinations across cases and to instantiate logical fallacies in a controlled way. Records in higher tiers provide stronger constraints on a suspect’s physical presence or capability, while lower-tier records mainly provide indirect traces that can support, weaken, or mislead an investigation.

Table 7: Database schema of Holmes-SQL, including each table’s benchmark role, columns, and a representative example record.

Table	Description	Columns	Example
location	Stores physical places used as crime scenes, background locations, or activity sites. The type field can be store, company, airport, ATM Port, hospital, school, or park.	location_id, name, street, type, latitude, longitude	(1, 'Gonzalez-Thomas Food Mart', '1711 Johnson Corners Apt. 054', 'store', 2.5215, 1.2836)
crime_scene_report	Records the reported incident, including the time, narrative description, and associated location.	report_id, timestamp, description, location_id	(1, '2024-05-20 22:15:00', 'Reported burglary at a retail store.', 1)
company	Links company entities to their physical locations.	company_id, company_name, location_id	(1, 'Cook LLC', 18)
airport	Stores airport metadata and links each airport to a location.	airport_id, abbreviation, full_name, location_id	(1, 'LCLemonCity1', 'North Johnville International Airport', 85)
article	Provides background context, public reports, or narrative evidence that may contain motives, misleading claims, or case-relevant facts.	article_id, timestamp, content	(41, '2024-12-05 11:20:00', 'Following its hostile acquisition by Newman-Deleon, ...')
person	Stores core identity information for suspects, victims, witnesses, and other relevant individuals.	person_id, name, phone_number, passport_number, blood_type, date_of_birth	(1, 'Barry Roberts Jr.', '554.989.1669', 'T14175562', 'A+', '1982-11-22')
income_record	Records employment and salary information, enabling reasoning over workplaces, job roles, financial motives, and company affiliations.	income_id, person_id, company_id, job_title, annual_salary, year	(1, 1, 1, 'Software Engineer', 61834.64, 2026)
bank_account	Connects people to bank accounts used in financial transactions.	account_number, person_id, creation_timestamp	('4637835542', 1, '2022-02-18T16:28:38.629598')
driver_license	Stores physical descriptors and license information that can be matched against witness reports or scene evidence.	driver_license_id, person_id, height_cm, eye_color, hair_color, gender, expiration_date	('DLy000001', 1, 169, 'Green', 'Black', 'Male', '2027-04-23')
vehicle	Links vehicles and license plates to their owners.	vehicle_id, owner_person_id, license_plate, car_make, car_model	(4, 4, 'LCLemonCity-MNL9980', 'Kia', 'Soul')
atm_transaction	Captures cash withdrawals, deposits, and other ATM activity at specific locations and times.	atm_transaction_id, account_number, location_id, timestamp, transaction_type, amount	(1, '4637835542', 53, '2025-01-04 14:41:17', 'withdrawal', 387.26)
flight	Stores flight schedules and airport routes, supporting alibi and travel reasoning.	flight_id, flight_number, departure_airport_id, arrival_airport_id, departure_timestamp, arrival_timestamp	(1, 'LCLemonCity10001', 1, 1, '2025-01-01 12:15:00', '2025-01-01 13:58:00')
passenger	Links people to flights and seat numbers.	flight_id, person_id, seat_number	(1, 52, '3D')
phone_call	Records call activity between phone numbers, enabling communication-based evidence and timeline reconstruction.	call_id, caller_phone_number, receiver_phone_number, timestamp, duration_seconds	(1, '554.989.1669', '(282)578-0830x3403', '2025-01-07 17:52:37', 50)
hospital_visit	Records medical visits, including admission and discharge times, which may support alibis or connect injuries to the crime. The reason_for_visit field follows the format <VISIT_TYPE>: <MEDICAL_CONDITION> [on <BODY_PART>] [after <CAUSE/EVENT>].	visit_id, person_id, location_id, admission_time, discharge_time, reason_for_visit	(20, 52, 105, '2024-05-21 01:00:00', '2024-05-21 04:30:00', 'Emergency Treatment: Laceration on right hand after kitchen accident')
purchase_invoice	Records purchases at stores, including the account used, time, amount, and item descriptions. The items_description field is a comma-separated list in the form <quantity>x <ITEM_NAME>.	invoice_id, account_number, location_id, timestamp, amount, items_description	(101, '4637835542', 1, '2024-05-20 21:30:00', 45.50, '2x DUCT TAPE, 1x FLASHLIGHT, 3x GLOVES')
traffic_cam_log	Records vehicles captured by cameras placed at specific locations rather than along streets.	log_id, location_id, license_plate, timestamp	(1, 1, 'LCLemonCity-MNL9980', '2024-05-20 09:15:00')

Table 8: Logical fallacies in Holmes-SQL and their SQL-solvable operational patterns. Each fallacy is instantiated as a reasoning trap requiring cross-table evidence verification.

Fallacy	Operational pattern in Holmes-SQL
False Cause	Assuming that event A caused event B simply because they occurred close together in time. For example, a suspect called the victim shortly before the murder, which may suggest that the suspect lured the victim to the scene. However, cross-referencing hospital admission records shows that the call was made while the suspect was already admitted to the hospital.
Hasty Generalization	Drawing a conclusion from superficial or insufficient evidence. For example, the crime scene report states that the killer had red hair and green eyes, so the solver immediately searches the <code>driver_license</code> table for a person with these exact traits, while ignoring evidence that the killer purchased hair dye and colored contact lenses.
Red Herring	Introducing prominent but irrelevant evidence that distracts from decisive clues. For example, a large cash withdrawal at a distant ATM or a speeding car on the other side of town may appear suspicious, but the actual solution depends on small, mundane clues left at the crime scene.
Anchoring	Relying too heavily on the most visible or abundant data. For example, a solver may focus on a suspect with extensive transaction and movement records, while overlooking the real culprit, whose sparse digital footprint makes them less noticeable in the database.
Confirmation Bias	Selecting only evidence that supports an initial hypothesis while ignoring contradictory context. For example, a suspect’s bank account was used to buy the murder weapon, but flight records show that the suspect was on an airplane at the exact time the card was used.
Appeal to Probability	Treating a plausible explanation as if it were proven. For example, finding a person’s passport at the scene may suggest that they dropped it while committing the crime, but the solver must check the <code>crime_scene_report</code> table to discover that the passport had been reported stolen earlier.
Affirming the Consequent	Inferring guilt because a clue matches what would be expected if the suspect were guilty. For example, if the killer used the victim’s stolen credit card, the card would appear in transaction logs. The card does appear in transaction logs tied to a suspect, but the solver must still check whether the card was stolen, planted, or used by someone else.
Argument from Ignorance	Concluding that a claim is false because there is no direct evidence for it. For example, a suspect may seem innocent because their license plate was not captured in the <code>traffic_cam_log</code> near the scene, but they may have walked, parked on another street, or taken public transportation.
Guilt by Association	Accusing someone solely because they are connected to a suspicious person or entity. For example, a suspect may have many <code>phone_call</code> records with a recently terminated employee, but the solver must determine whether this relationship is actually relevant to the crime.
Blind Loyalty	Trusting an apparently credible statement without verifying it against structured evidence. For example, a news article may report that someone publicly confessed to the crime, but the solver must still check whether the confessor’s physical attributes in the <code>driver_license</code> table match the evidence from the scene.

CASE DESCRIPTION Dispatch log, March 2025. Call came in for a fall from a tower in the Bishop Skyway financial district. The victim is a high-level finance director, and initial reports suggest this was no accident. We're treating this as a homicide.	SQL SOLUTION <u>Crime:</u> T1: The initial dispatch mentioned a homicide in the Bishop Skyway financial district. I need to find the official crime scene report to get the exact location, time, and any initial details recorded by the first responders. Q1: <code>SELECT * FROM crime_scene_report WHERE description LIKE '%Bishop Skyway%';</code> <u>Victim:</u> T1: A key fob with the license plate 'LemonCity-NEC2118' was found on the decedent. I will use this license plate to query the vehicle and person tables to identify the owner, who is likely our victim. Q1: <code>SELECT p.* FROM person p JOIN vehicle v ON p.person_id = v.owner_person_id WHERE v.license_plate = 'LemonCity-NEC2118';</code> T2: I have identified the victim as Kathy Boyd (person_id 43). The initial report described her as a 'high-level finance director'. I will now verify her employment history to understand her professional background, which may be relevant to the motive. Q2: <code>SELECT c.company_name, ir.job_title, ir.annual_salary, ir.year FROM income_record ir JOIN company c ON ir.company_id = c.company_id WHERE ir.person_id = 43 ORDER BY ir.year DESC;</code> <u>Suspect:</u> T1: The victim, Kathy Boyd, was the Director of Finance at Foster Group. To find individuals with a potential motive, I must search the article table for news reports mentioning her or her company, as per investigation protocol. Q1: <code>SELECT content FROM article WHERE content LIKE '%Kathy Boyd%' OR content LIKE '%Foster Group%';</code> T2: The articles revealed four individuals who were professionally and financially harmed by Kathy Boyd's audits: Kayla Mitchell, Scott Reed, Misty Moore, and Nathaniel Long. These are my primary suspects. I will retrieve their personal details to begin investigating their whereabouts on the night of the murder. Q2: <code>SELECT * FROM person WHERE name IN ('Kayla Mitchell', 'Scott Reed', 'Misty Moore', 'Nathaniel Long');</code> T3: I have my suspects' IDs (1275, 1349, 764, 508). Now I need to check for any evidence placing them near the crime scene (lat: 1.51, lon: -2.58) around the time of the murder (21:30). I'll start by checking traffic camera logs for their vehicles on the day of the crime. Q3: <code>SELECT p.name, v.license_plate, tcl.timestamp FROM person p JOIN vehicle v ON p.person_id = v.owner_person_id JOIN traffic_cam_log tcl ON v.license_plate = tcl.license_plate WHERE p.person_id IN (1275, 1349, 764, 508) AND DATE(tcl.timestamp) = '2025-03-12';</code> T4: Traffic cameras placed Kayla Mitchell's and Nathaniel Long's cars near the scene. Now I'll check for other activities, like ATM withdrawals or store purchases, made by any of the four suspects within a 5km radius of the crime scene on the night of the murder. Q4: <code>SELECT p.name, 'ATM' as activity_type, atm.timestamp, l.name as location_name, SQRT(POWER((l.latitude - 1.51, 2) + POWER((l.longitude - -2.58, 2)) * 10 AS distance_km FROM person p JOIN bank_account ba ON p.person_id = ba.person_id JOIN atm_transaction atm ON ba.account_number = atm.account_number JOIN location l ON atm.location_id = l.location_id WHERE p.person_id IN (1275, 1349, 764, 508) AND atm.timestamp BETWEEN '2025-03-12 19:00:00' AND '2025-03-12 22:00:00' AND distance_km < 5 UNION ALL SELECT p.name, 'Purchase' as activity_type, pi.timestamp, l.name as location_name, SQRT(POWER((l.latitude - 1.51, 2) + POWER((l.longitude - -2.58, 2)) * 10 AS distance_km FROM person p JOIN bank_account ba ON p.person_id = ba.person_id JOIN purchase_invoice pi ON ba.account_number = pi.account_number JOIN location l ON pi.location_id = l.location_id WHERE p.person_id IN (1275, 1349, 764, 508) AND pi.timestamp BETWEEN '2025-03-12 19:00:00' AND '2025-03-12 22:00:00' AND distance_km < 5;</code>
CHECKLIST <u>Crime:</u> - The crime occurred at Apex Executive Suites at 2025-03-12 21:30:00, and a key fob for an Acura with license plate LemonCity-NEC2118 was recovered from the decedent's pocket. <u>Victim:</u> - The victim is Kathy Boyd, and the cause of death is massive trauma consistent with a fall from height. <u>Suspect:</u> A. Nathaniel Long - Nathaniel Long was a Risk Management Consultant at Innovate Forward whose contract was terminated in 2025 following an audit by Kathy Boyd, with his salary dropping from \$182,500 to \$45,000. - Nathaniel Long's vehicle, license plate LemonCity-FMH5041, was recorded by a camera at Christian Union Security Camera on 2025-03-12 at 21:25:00. B. Kayla Mitchell - Kayla Mitchell was the Chief Financial Officer at Synergy Solutions Inc. and was ousted from her position in 2025 following an audit by Kathy Boyd. - Kayla Mitchell withdrew \$10,000 from Prosperity Bank ATM on 2025-03-11 at 18:00:00. - Kayla Mitchell was admitted to Harper Clinic from 2025-03-12 20:45:00 to 2025-03-12 23:55:00 for 'Emergency Treatment: Acute stress reaction and syncope after job termination'. - Kayla Mitchell's vehicle, license plate LemonCity-KGB8914, was recorded by a camera at Cassandra River CCTV Node on 2025-03-12 at 21:10:00. C. Scott Reed - Scott Reed was the VP of Logistics at Nexus Innovations and was terminated in 2025 after an audit by Kathy Boyd revealed fraudulent overbilling. - A withdrawal of \$5,000 was made from Scott Reed's bank account at Federal Bank ATM on 2025-03-12 at 21:05:00. - Scott Reed was on flight LCLemonCity404, which departed at 2025-03-12 20:30:00 and arrived at 2025-03-12 22:15:00. D. Misty Moore - Misty Moore was demoted from Data Science Lead to Junior Analyst at QuantumLeap Analytics in 2025 following a reorganization led by Kathy Boyd, with her salary decreasing from \$145,000 to \$65,000. - A purchase of '1X CROWBAR, 2X HEAVY DUTY ZIP TIES' for \$75.50 was made using Misty Moore's bank account at Parker-Hamilton Food Mart on 2025-03-12 at 21:20:00. - Misty Moore was admitted to Howard Memorial Hospital from 2025-03-12 19:00:00 to 2025-03-13 10:00:00 for 'Follow-up: Post-injury evaluation for ankle; complications required overnight observation'. - Misty Moore was treated at Howard Memorial Hospital from 2025-03-05 14:00:00 to 2025-03-05 16:30:00 for 'Treatment: Sprain on right ankle after sports injury'.	...

Figure 4: Example of a Holmes-SQL dataset sample. Each sample includes a narrative case description, a relational database, a checklist of evidence objectives, and a gold SQL investigation path that recovers the relevant crime, victim, and suspect evidence.

Tier 1: High-Certainty Mobility Evidence.

Flight and passenger records provide the strongest form of evidence in Holmes-SQL. If a suspect is listed as a passenger on a flight that overlaps with the crime time, the record establishes a strong alibi because the suspect is assumed to be physically on the plane. However, flights after the crime are not treated as alibis. Instead, they may indicate a possible escape route and must be interpreted in relation to the crime timeline.

Tier 2: Conditional Medical Evidence. Hospital visit records provide evidence of presence at a medical facility, but their inferential strength depends on the visit reason, admission time, and discharge time. A minor injury, routine checkup, or visit to another patient does not necessarily prevent a suspect from committing the crime. In contrast, severe conditions such as unconsciousness, emergency surgery, or intensive treatment can establish incapacitation during the relevant time window. Past hospital visits may also impose physical constraints, such as a recent fractured wrist limiting the suspect's ability to use a heavy weapon.

Tier 3: Indirect Behavioral Evidence. ATM transactions, phone calls, store purchases, and traffic camera logs are treated as weak or circumstantial evidence because they record the use of an asset rather than the direct presence of a person. A bank card, phone, vehicle, or account may be shared, borrowed, stolen, or manipulated. Therefore, these records are useful for constructing misleading traces and fallacy-inducing evidence patterns, but they cannot by themselves prove that the asset owner was present at the crime scene or responsible for the crime.

Deductive Physical Constraints. In addition to the tiered evidence sources, Holmes-SQL uses driver's license attributes as deductive constraints. Fields such as height, gender, eye color, hair color, and other physical descriptors can eliminate suspects when they conflict with crime-scene requirements. For example, if the crime requires reaching a high window or matching a specific physical description, the agent must cross-check these constraints against suspect profiles rather than relying only on activity traces.

A.5 Suspect Role Definition

Each case contains a fixed set of four ground-truth suspects, each assigned one of the following roles:

- **Innocent Suspect with Standard Fallacy:** A suspect who appears guilty due to evidence that perfectly instantiates the target logical fallacy. The reasoning pattern matches the exact structure of the fallacy being tested, creating misleading but logically invalid implications of guilt. This suspect acted alone in their documented activities and has no connection to any accomplice.
- **Innocent Suspect with Transposed Fallacy:** A suspect who appears guilty due to the same fallacious reasoning structure as Role 1, but applied to a different evidential dimension. The logical error remains identical but manifests through different data features. This suspect operated independently in all documented activities without accomplices.
- **Innocent Suspect with Contradicted Fallacy:** A suspect whose evidence contains the target fallacy's reasoning pattern but also includes explicit logical constraints that directly negate the fallacious conclusion. A hard contradiction exists that should override any fallacious inference. This suspect's activities were conducted solely by themselves without any accomplice.
- **True Murderer (Solo Perpetrator):** The single individual who committed the murder, acting entirely alone without any accomplice or co-conspirator. This suspect's guilt is strategically concealed by weaponizing the same fallacious reasoning pattern used to exonerate innocent suspects. Their incriminating evidence is deliberately embedded within apparently routine patterns that superficially resemble the innocent explanations of other suspects, requiring investigators to apply deeper contextual analysis to distinguish genuinely exculpatory patterns from those that merely camouflage guilt.

A.6 Investigation Stages

Holmes-SQL simulates a standard crime investigation that starts from a short narrative case description and proceeds through three stages: crime scene, victim, and suspect. In the first stage, the agent uses narrative cues such as location, time, and incident description to recover the official crime scene report. In the second stage, the agent

identifies the victim and reconstructs the victim-centered context, including recovered objects, relationships, ownership records, workplace links, and other clues that reveal potential suspects. In the third stage, the agent reasons about these suspects by comparing motives, activities, alibis, physical constraints, and evidence of varying inferential strength.

Fig. 5 illustrates this process. The example shows how the agent moves from the narrative story to the crime scene, uses victim-linked evidence to recover the victim and candidate suspects, and then revises its suspect hypotheses after recognizing that asset-based logs such as ATM records, phone logs, and camera traces can be misleading unless verified against stronger evidence such as flight records and hospital incapacitation.

A.7 World Constraints

Holmes-SQL uses a simplified urban world to maintain spatial and temporal coherence across generated cases. Each city is represented as a two-dimensional coordinate system centered at $(0,0)$, where one coordinate unit corresponds to 10 kilometers. All location coordinates are bounded between -50.0 and 50.0 , and Euclidean distance is used to estimate travel distance between locations.

To support physical-feasibility checks, we assume approximate travel speeds of 30 km/h for city travel and 500 km/h for flights, with an additional 30-minute overhead for airport procedures. These assumptions are treated as flexible guidance rather than strict rules. They are mainly used to avoid clearly impossible timelines while preserving the ambiguity needed for open-ended investigation.

All of these world constraints are provided to agents as part of the initial context before the investigation begins.

A.8 Quality Control Analysis

This section provides additional details on the quality-control process used in Holmes-SQL. Our validation focuses on two levels: script-level validity and artifact-level executability. The script-level checks ensure that each case has a coherent investigative structure before database instantiation, while the artifact-level checks ensure that the generated database, checklist, and SQL reference path faithfully preserve the accepted script. Tab. 9 summarizes the main validation dimensions and the corresponding refinement actions.

At the script level, annotators review each refined case according to four criteria. First, logical fallacy integration checks whether the intended fallacy is reflected in the evidence pattern rather than inserted as an artificial distractor. Second, suspect role suitability verifies that each suspect has a plausible relationship-based motive and role in the case. Third, logical consistency checks whether the timeline, locations, injuries, ownership links, and alibis are mutually compatible. Finally, solvability ensures that the true murderer can be uniquely identified by combining the available evidence. Scripts that fail any of these checks are manually refined rather than directly used for database construction.

At the artifact level, we inspect the case description, checklist, database records, and gold SQL investigation path. The case description must provide enough information to start the investigation, but it must not directly reveal the culprit. The checklist must preserve all evidence objectives from the accepted script and organize them according to the crime, victim, and suspect stages. The database must contain recoverable tuples for each checklist item, and the SQL reference solution must execute successfully and retrieve the intended evidence. We also verify that the final evidence set eliminates the innocent suspects and leaves exactly one murderer under the single-perpetrator assumption.

A.9 Additional Dataset Analysis

Database Size. To further contextualize Holmes-SQL relative to existing SQL benchmarks, Tab. 10 reports the average database size of Spider, ExCyTIn-Bench, and Holmes-SQL. Compared with Spider, Holmes-SQL is substantially larger in both the number of tables and the number of rows, reflecting its focus on multi-table evidence gathering and distractor-rich investigative reasoning rather than standard text-to-SQL parsing alone. Although ExCyTIn-Bench has larger databases on average, the more moderate scale of Holmes-SQL makes it easier to deploy, reproduce, and extend in future research.

Ground-truth row density. Ground-truth rows are extremely sparse relative to the full database. Each case contains only a small set of planted clues, while behavior-centric tables such as phone calls, purchases, passenger records, and financial transactions contain thousands to tens of thousands of rows. This design reflects a realistic data imbalance, where routine activities such as purchases oc-

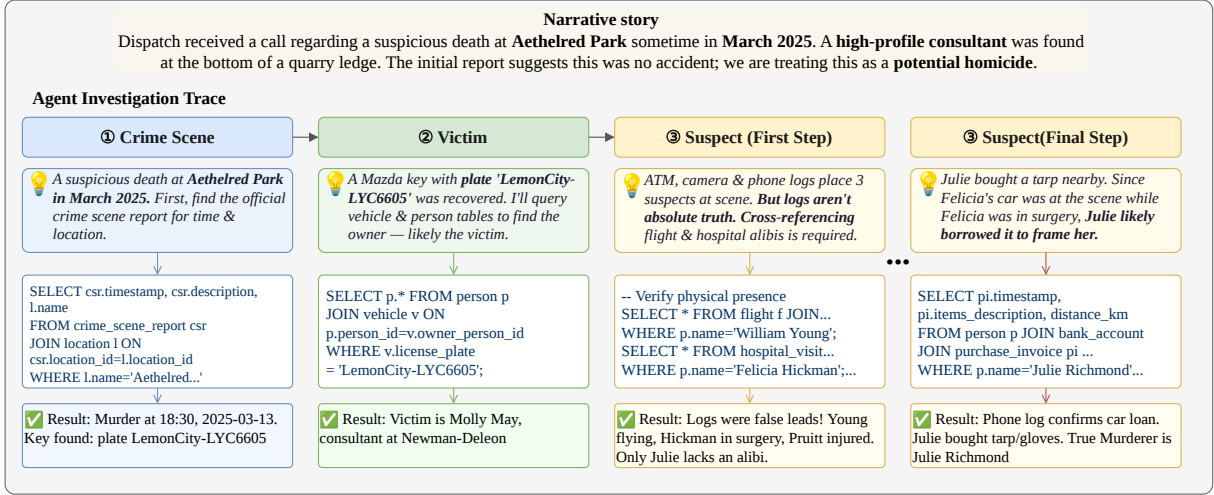


Figure 5: Illustration of an investigation trace in Holmes-SQL. Starting from a narrative case description, the agent identifies the crime scene, reconstructs the victim-centered context to recover candidate suspects, and iteratively reasons over suspects by cross-checking weak asset-based traces against stronger alibis, medical constraints, and physical evidence.

cur far more frequently than hospital visits or flight records. It also creates a challenge for supervised learning methods, since positive ground-truth rows are greatly outnumbered by non-critical rows, and prevents cases from being solved through simple lookup or surface-level pattern matching. Tab. 11 reports the average number of ground-truth rows, average table size, and ground-truth coverage percentage for each table.

SQL Solution Step Distribution. We analyze the number of steps in the reference SQL solution for each case, where each step consists of executing one SQL query against the database and retrieving its results. As shown in Fig. 6, most cases require 10 to 15 steps, with the distribution peaking at 12 steps. The full range spans 7 to 18 steps, indicating that Holmes-SQL cases generally require multi-step investigative reasoning rather than single-query retrieval.

A.10 Evidence-Combination Diversity

We compute evidence-combination diversity at two levels: within each case and across cases. For within-case diversity, we represent each suspect by an evidence-type signature. The signature records which activity evidence types are associated with the suspect, including flight, hospital, phone, atm, store, and camera. For example, a suspect with hospital and purchase evidence is represented as {hospital, store}.

Given two suspects s_i and s_j in the same case, we compute their evidence-type difference using

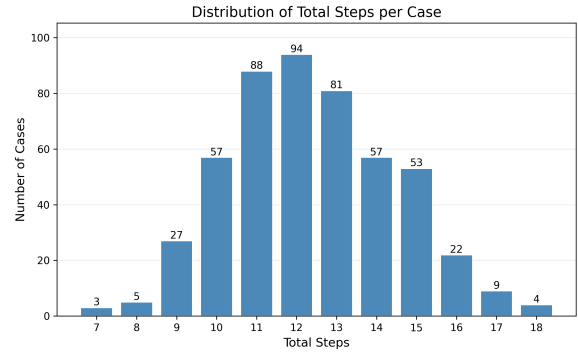


Figure 6: Distribution of the number of SQL reasoning steps in the reference ground-truth solutions. Each bar reports how many cases require a given number of SQL solution steps.

Jaccard distance:

$$d(s_i, s_j) = 1 - \frac{|E_i \cap E_j|}{|E_i \cup E_j|},$$

where E_i and E_j are the evidence-type signatures of the two suspects. Since each case contains four suspects, we compute this distance over the six unordered suspect pairs and average them:

$$D_{\text{case}} = \frac{1}{6} \sum_{1 \leq i < j \leq 4} d(s_i, s_j).$$

A value close to 0 means that the suspects share very similar evidence patterns, while a value close to 1 means that the suspects are associated with different types of evidence. We report the average of D_{case} over all cases as the within-case evidence-combination diversity.

Table 9: Quality-control dimensions used in Holmes-SQL and their corresponding refinement actions.

Check	Refinement Action
<i>Script-level validation</i>	
Logical fallacy integration	Revise evidence patterns so that misleading clues arise from the intended fallacy rather than arbitrary noise.
Suspect role suitability	Adjust suspect profiles, motives, or relationships to ensure that all suspects are plausible candidates.
Logical consistency	Repair inconsistent timelines, impossible movements, conflicting injuries, or incompatible ownership links.
Solvability	Add, remove, or revise evidence so that the true murderer is uniquely identifiable.
<i>Artifact-level validation</i>	
Case description quality	Remove over-revealing details or add partial cues when the narrative is too vague.
Checklist coverage	Ensure that all required crime, victim, and suspect evidence is preserved as recoverable objectives.
Database grounding	Verify that each checklist item is supported by concrete database tuples.
SQL executability	Correct SQL reference paths that fail to execute or fail to retrieve the intended evidence.
Final uniqueness	Confirm that evidence eliminates innocent suspects and leaves exactly one true murderer.

Table 10: Average database size comparison across SQL benchmarks. Average rows are reported in thousands, and average database size is reported in gigabytes.

Benchmark	Avg. tables	Avg. rows (K)	Avg. DB size (GB)
Spider	3.5	8.6	0.030
ExCyTIn-Bench	45.0	364.7	0.447
Holmes-SQL (ours)	17.0	104.59	0.062

For cross-case diversity, we construct a case-level evidence template for each case. Each template is the order-invariant multiset of the four suspect signatures after sorting the evidence types within each signature. For example, a case with suspect signatures $\{\text{hospital}, \text{store}\}$, $\{\text{flight}\}$, $\{\text{camera}, \text{phone}\}$, and $\{\text{hospital}, \text{atm}\}$ is represented as:

[hospital+store, flight,
camera+phone, hospital+atm]

This representation ignores suspect ordering, so two cases are considered to have the same template if they contain the same multiset of suspect-level evidence signatures.

We then compute the Jaccard distance between case-level templates. Because templates are multisets, we use the multiset version of Jaccard distance:

$$d(C_a, C_b) = 1 - \frac{\sum_x \min(c_a(x), c_b(x))}{\sum_x \max(c_a(x), c_b(x))},$$

Table 11: Average ground-truth row density by table in Holmes-SQL. The coverage percentage is computed as the average number of ground-truth row identifiers divided by the average total number of rows in the corresponding table.

Table	GT IDs	Rows	Cov. %
airport	0.00	4.69	0.0000
article	4.00	182.52	2.1905
atm_transaction	1.07	7374.83	0.0145
bank_account	5.30	1758.01	0.3014
company	0.00	35.20	0.0000
crime_scene_report	1.00	98.26	1.0177
driver_license	5.30	1672.38	0.3168
flight	0.85	263.73	0.3238
hospital_visit	2.20	370.63	0.5930
income_record	7.33	5265.48	0.1393
location	12.70	213.66	5.9430
passenger	0.85	19243.12	0.0044
person	5.30	1758.01	0.3014
phone_call	1.30	13598.20	0.0096
purchase_invoice	1.92	47940.12	0.0040
traffic_cam_log	2.11	3254.41	0.0650
vehicle	5.28	1556.38	0.3392

where $c_a(x)$ and $c_b(x)$ denote the counts of suspect-level evidence signature x in cases C_a and C_b , respectively. We compute this distance over all unordered pairs of cases and report the average as cross-case evidence-combination diversity. A higher value indicates that different cases require distinct investigative evidence templates rather than reusing the same suspect-evidence configurations.

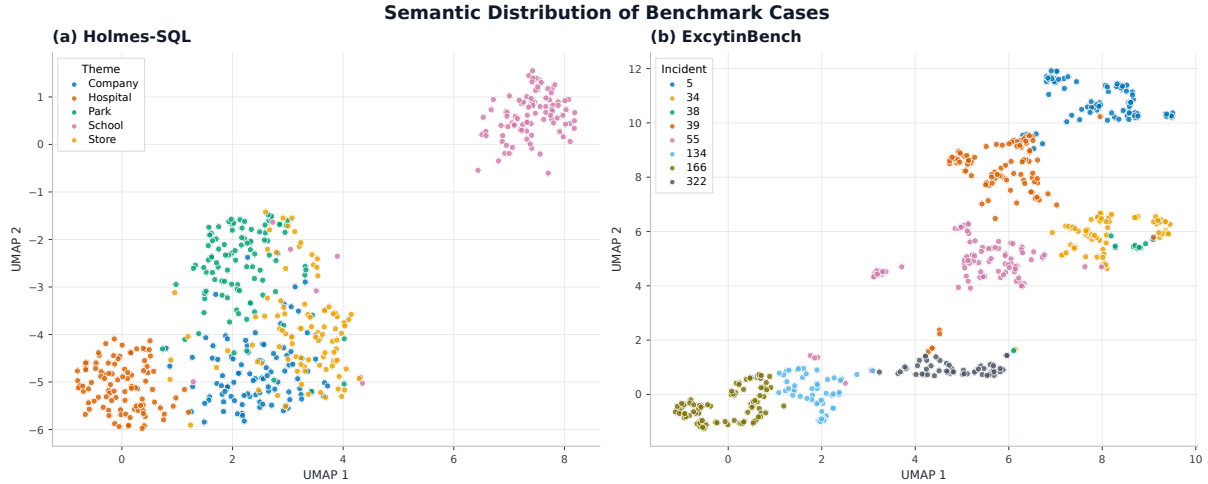


Figure 7: UMAP visualization of case-level semantic embeddings for Holmes-SQL and ExCytIn-Bench. Each point represents one Markdown-serialized case embedded with embeddinggemma-300m.

A.11 Semantic Distribution Visualization

The evidence-combination diversity above measures only evidence-type configurations and does not capture the semantic content of cases. As a complementary qualitative diagnostic, we convert each case into a Markdown document before embedding. For Holmes-SQL, we serialize each script into a Markdown document with sections for the crime scene, victim details, suspect profiles, motives, alibis, and activity records, including flights, ATM transactions, phone calls, hospital visits, store purchases, and camera logs. For ExCytIn-Bench, we serialize each case using its context, question, shortest alert path, and solution text.

We then encode each Markdown document with embeddinggemma-300m⁵ and project the resulting case-level embeddings into two dimensions with UMAP. Fig. 7 visualizes Holmes-SQL cases by urban theme and ExCytIn-Bench cases by incident identifier. This visualization is not used as a formal diversity metric, since UMAP can vary with embedding models and projection settings. Instead, it provides a qualitative view showing that both benchmarks contain structured semantic groupings, while Holmes-SQL also shows visible variation within and across its urban themes.

Table 12: Performance comparison on a sample of 100 cases between human evaluators and the top LLM agent baseline.

Method	Solve Rate (%)	Duration (min)
Humans (avg.)	87.5	23.97
CoT+ReAct (Gemini 3.1 Pro)	52.0	4.84

B Additional Details on Benchmark Experiments

B.1 Human Performance

Human performance provides a useful reference baseline for our benchmark. Evaluated on a representative sample of 100 cases (Tab. 12), human participants achieve an average solve rate of 87.5%, substantially outperforming our strongest agent configuration (CoT+ReAct with Gemini 3.1 Pro at 52.0%), though requiring nearly five times longer per mystery (23.97 vs. 4.84 minutes). To establish even more robust baselines, we plan to scale evaluations across a broader demographic of skill levels and deploy an interactive web-based interface. This platform will enable participants to play directly alongside or against agents, yielding rich trajectory data to study reasoning gaps, decision-making behaviors, and collaboration effectiveness.

B.2 Conditional Analysis of Query Errors

To understand the failure modes of SQL-based agents, we analyze the conditional distribution of SQL errors across different investigation stages and

⁵<https://huggingface.co/google/embeddinggemma-300m>

error categories. As detailed in Tab. 13, decision-critical SQL failures are heavily concentrated in the later investigation phases: only 3.60% of errors occur during crime identification, whereas 21.25% occur during victim identification, and 75.15% occur during suspect assessment. In total, 96.40% of malformed queries arise in the final two stages.

Table 13: Conditional distribution of SQL errors across investigation stages.

Model	Crime	Victim	Suspect
gemini-3.1-pro-preview	0.0000	0.1429	0.8571
gpt-5.4-2026-03-05	0.0000	0.2000	0.8000
glm-5	0.0345	0.0690	<u>0.8966</u>
gemini-3-flash-preview	0.0289	0.0289	0.9422
gemma-4-31b	0.1591	0.2045	0.6364
deepseek-v3.2	0.0417	0.2361	0.7222
qwen3-235b-a22b-instruct-2507	<u>0.0423</u>	0.3492	0.6085
qwen3-coder-480b-a35b-instruct	0.0132	<u>0.2500</u>	0.7368

Categorizing these failures (Tab. 14) reveals that they are predominantly schema-linking errors. The missing column error (no_such_column) constitutes the largest share at 65.67%, followed by general syntax_error (18.01%) and ambiguous_column_name (9.84%). This concentration of failures in downstream stages stems from clue fragmentation: as the investigation progresses, evidence becomes increasingly dispersed across multiple relational tables, significantly elevating schema-grounding difficulty and compounding reasoning overhead for the agents.

B.3 Backbone-Level Analysis of LLM-Judge Agreement

To ensure that our evaluation judge (Gemini-2.5-Pro) does not exhibit intra-family bias toward models from the same provider (e.g., Gemini-3.1-Pro and Gemini-3-Flash), we compute Cohen’s Kappa (κ) between human annotators and the LLM judge disaggregated by backbone model. As reported in Tab. 15, the judge maintains high and consistent agreement with human annotators across all evaluated backbones, achieving an overall $\kappa = 0.85 \pm 0.04$. Notably, the agreement scores for same-family models (0.8072 for Gemini-3-Flash and 0.8163 for Gemini-3.1-Pro) are comparable to those for external backbones, confirming that the automated evaluation does not suffer from systematic family-level favoritism.

B.4 Baseline Model Specifications

To improve reproducibility, Tab. 16 lists the exact model identifiers used throughout our experiments.

B.5 Prompts

To save space, we report only the difference between the Base and CoT prompts, along with the recall evaluator prompt.

CoT workflow instruction

To solve this case, strictly adhere to the following investigation workflow:

1. Query the ‘crime_scene_report’ table using specific details from the narrative to locate the correct report.
2. Utilize the ‘person’ table to identify the victim and extract their relevant background information.
3. Query the ‘article’ table for records linking the victim and their company to build a comprehensive list of potential suspects.
4. Investigate each suspect’s activities. You must critically weigh the reliability of the evidence and validate alibis by applying strict spatial and temporal constraints.
5. Synthesize the collected evidence logically to deduce the true murderer.

Recall evaluator instruction

You are a precise evidence evaluator for a murder-mystery detective benchmark.

You will be given:

1. A detective’s investigation log (their SQL queries and findings up to a certain step)
2. A numbered list of ground-truth objectives (facts that should have been discovered)

Your task:

For EACH objective, decide whether the key fact was clearly present or directly implied somewhere in the investigation log.

Output format – return ONLY a JSON array, one object per objective, in this exact shape:

```
[
  {
    "objective": "<exact objective text>",
    "score": 0
  },
  {
    "objective": "<exact objective text>",
    "score": 1
  },
  ...
]
```

Scoring rules:

- score = 1 if the objective’s core fact is present or unambiguously implied in the log
- score = 0 if the fact is absent, unclear, or only partially mentioned
- Do NOT infer facts not shown in the log
- Reproduce the objective text EXACTLY as given – do not paraphrase

B.6 Solve Rate by Logical Fallacy and Theme

We analyze model performance across logical fallacy types and urban themes. Fig. 8 reports solve rates for all evaluated models under each fallacy and theme setting. Performance varies across both dimensions, showing that Holmes-SQL difficulty is shaped not only by the targeted reasoning pattern but also by the narrative context and available evidence types. Some fallacies, such as guilt by association and hasty generalization, are solved more often by stronger models, whereas anchoring and argument from ignorance remain challenging. Across themes, company, park, school, and store cases achieve higher solve rates than hospital cases,

Table 14: Conditional distribution of SQL errors by error type.

Model	no_such_column	syntax_error	ambiguous_column_name	other
gemini-3.1-pro-preview	0.7143	0.2857	0.0000	0.0000
gpt-5.4-2026-03-05	1.0000	0.0000	0.0000	0.0000
glm-5	0.8276	0.0690	0.0345	0.0690
gemini-3-flash-preview	<u>0.8902</u>	0.0289	0.0000	<u>0.0809</u>
gemma-4-31b	0.5000	<u>0.3182</u>	0.1364	0.0455
deepseek-v3.2	0.7500	0.0833	<u>0.1389</u>	0.0278
qwen3-235b-a22b-instruct-2507	0.4127	0.4603	0.0212	0.1058
qwen3-coder-480b-a35b-instruct	0.6184	0.1053	0.2237	0.0526

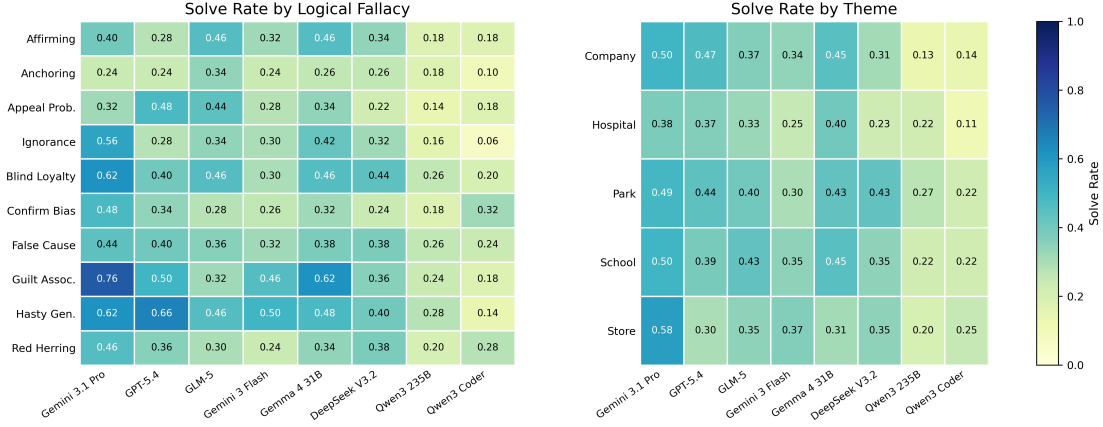


Figure 8: Solve Rate breakdown by logical fallacy and location theme. The left heatmap reports model solve rates across logical fallacy categories, while the right heatmap reports model solve rates across theme settings. Darker colors indicate higher solve rates.

Table 15: Disaggregated Cohen’s Kappa (κ) agreement between human evaluators and the LLM judge across different backbone models.

Model	Cohen’s Kappa (κ)
gemini-3.1-pro-preview	0.8163
gpt-5.4-2026-03-05	0.8837
glm-5	0.9021
gemini-3-flash-preview	0.8072
gemma-4-31b	0.8479
deepseek-v3.2	0.8352
qwen3-235b-a22b-instruct-2507	0.8888
qwen3-coder-480b-a35b-instruct	0.8305
Overall	0.85 $\pm$ 0.04

suggesting that domain-specific evidence, such as medical visits and incapacitation constraints, can make SQL-based investigation harder.

B.7 False Accusation Target Analysis

We analyze which suspect roles models select when they make incorrect accusations. A case is included only if the model returns a valid `murder_id` and the

predicted identifier differs from the ground-truth murderer. As shown in Fig. 10, false accusations are often distributed across the three designed innocent suspect roles: standard fallacy, transposed fallacy, and contradicted fallacy. This indicates that fallacy-based distractors can attract model predictions when the model follows an invalid reasoning path.

B.8 Investigation Length and Solve Rate

We analyze the number of steps taken by each agent backbone during an investigation and relate it to the Solve Rate. Fig. 9 groups investigation lengths into step-count bins for readability and overlays the Solve Rate within each bin. The results show that different LLM backbones exhibit distinct investigation-length patterns: some concentrate most cases in shorter bins, while others more often produce longer investigations. The solve-rate curves further show that longer investigations do not necessarily yield higher accuracy, suggesting

Table 16: Baseline model specifications used in the experiments.

Model Identifier	Official Reference
gemini-3.1-pro-preview	https://ai.google.dev/gemini-api/docs/models/gemini-3.1-pro-preview
gpt-5.4-2026-03-05	https://developers.openai.com/api/docs/models/gpt-5.4
glm-5	https://huggingface.co/zai-org/GLM-5
gemini-3-flash-preview	https://ai.google.dev/gemini-api/docs/models/gemini-3-flash-preview
gemma-4-31b	https://huggingface.co/google/gemma-4-31b
deepseek-v3.2	https://huggingface.co/deepseek-ai/DeepSeek-V3.2
qwen3-235b-a22b-instruct-2507	https://huggingface.co/Qwen/Qwen3-235B-A22B-Instruct-2507
qwen3-coder-480b-a35b-instruct	https://huggingface.co/Qwen/Qwen3-Coder-480B-A35B-Instruct
gemini-2.5-pro	https://ai.google.dev/gemini-api/docs/models/gemini-2.5-pro

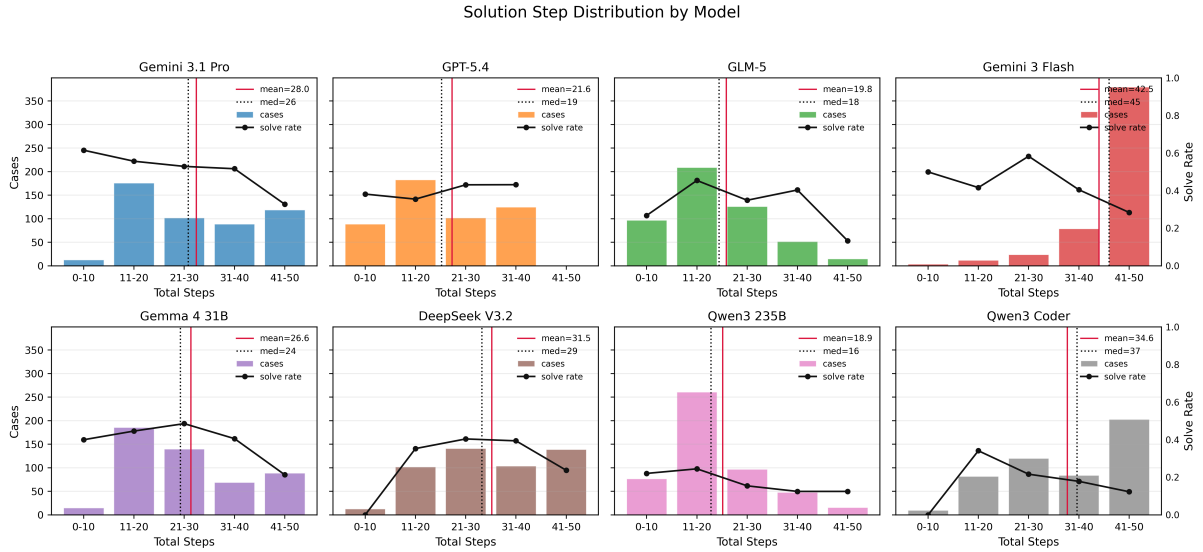


Figure 9: Distribution of total investigation steps and Solve Rate by agent. Bars show the number of cases in each step-count bin, black lines show the Solve Rate within each bin, and vertical lines indicate the mean and median number of investigation steps.

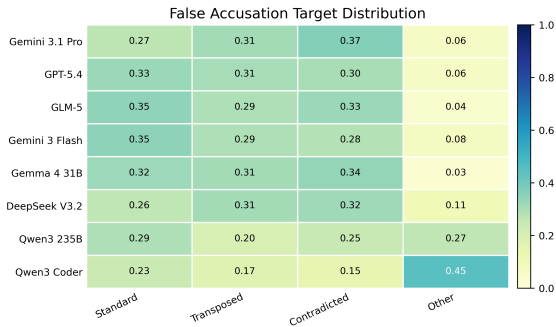


Figure 10: Distribution of false accusation targets by suspect role. Other denotes incorrect predictions outside the predefined suspect roles.

that success depends not only on the number of steps taken, but also on whether those steps retrieve, verify, and integrate the right evidence.

B.9 Cost and Resource Consumption

We report the resource consumption of each agent on Holmes-SQL in Tab. 17, including solve rate, average input tokens, average output tokens, average monetary cost, and average runtime. A higher solve rate indicates better task performance, while lower values are preferred for all resource metrics. The results show that no model dominates across all dimensions. For example, gemini-3.1-pro-preview achieves the highest solve rate but also has the highest average cost, while gemma-4-31b provides a stronger cost-performance trade-off but requires longer runtime. In contrast, qwen3-coder-480b-a35b-instruct is the fastest model but has the lowest solve rate. These results highlight the practical trade-off between investigative accuracy, cost, and latency when deploying SQL-based investigation agents.

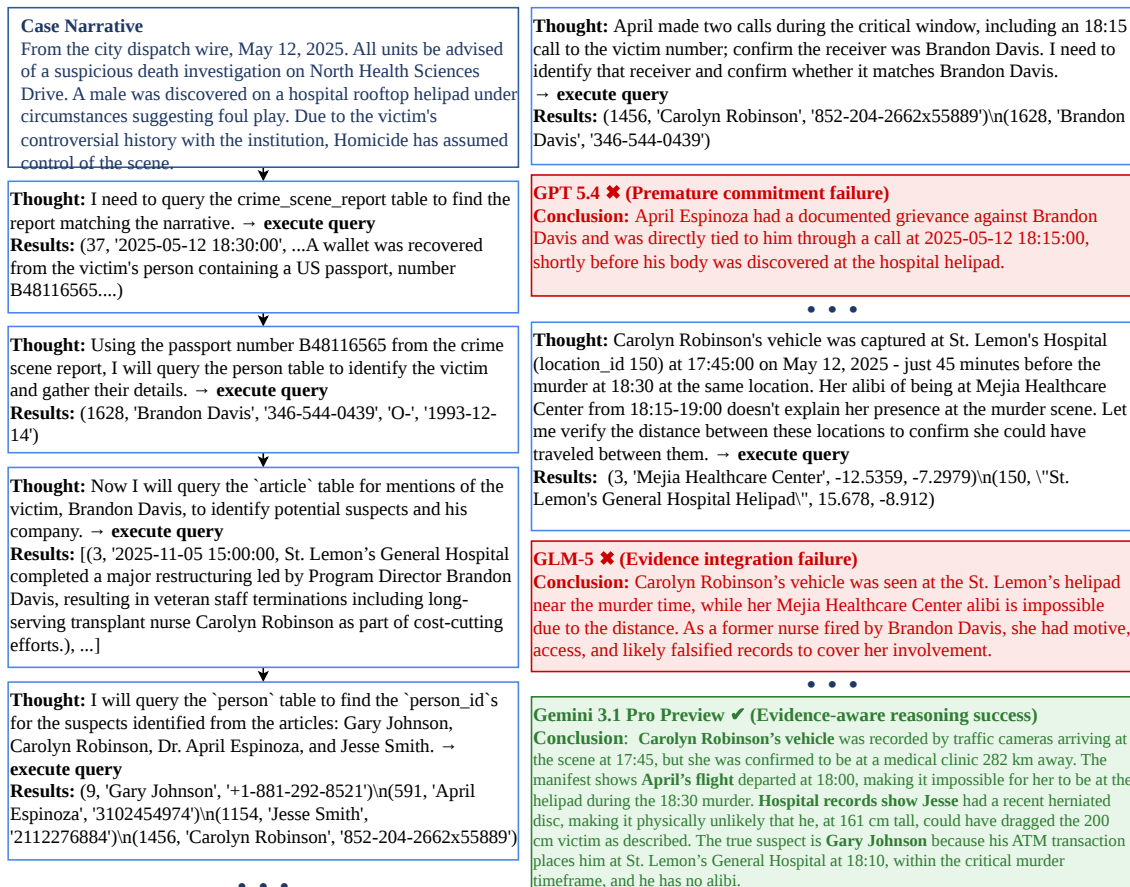


Figure 11: Case study of model failures and successful evidence-aware investigation in Holmes-SQL. GPT-5.4 commits prematurely to a suspect from weak phone-call evidence, while GLM-5 overweights a vehicle trace despite stronger contradictory evidence. Gemini-3.1-Pro-Preview succeeds by maintaining competing hypotheses, cross-checking evidence strength, and identifying the true murderer through stronger spatiotemporal evidence.

Table 17: Resource consumption results on Holmes-SQL. **Bold** values indicate the best results in each column across all models.

Model	Solve Rate ($\uparrow$)	Avg. Input Tokens ($\downarrow$)	Avg. Output Tokens ($\downarrow$)	Avg. Cost (\$) ($\downarrow$)	Avg. Runtime (min) ($\downarrow$)
<i>Proprietary models</i>					
gemini-3.1-pro-preview	0.4900	135544.15	5620.71	0.6237	4.7331
gpt-5.4-2026-03-05	0.3940	109586.14	5904.31	0.3429	1.5603
glm-5	0.3760	85187.75	9753.93	0.1122	2.8048
gemini-3-flash-preview	0.3220	212890.55	7558.87	0.1785	3.0810
<i>Open-source models</i>					
gemma-4-31b	0.4080	119755.97	5385.28	0.0150	9.6134
deepseek-v3.2	0.3340	146918.45	7630.28	0.0914	6.7505
qwen3-235b-a22b-instruct-2507	0.2080	100728.11	5128.60	0.0267	1.0911
qwen3-coder-480b-a35b-instruct	0.1880	141139.06	6767.49	0.0387	0.8248

Model	Crime	Victim	Suspect Set	Murderer
<i>Proprietary models</i>				
gemini-3.1-pro-preview	0.9459	0.9880	0.3387	0.4910
gpt-5.4-2026-03-05	0.9880	0.9679	0.4509	0.3948
glm-5	0.9359	0.9499	0.4028	0.3808
gemini-3-flash-preview	0.7615	0.9780	0.3667	0.3327
<i>Open-source models</i>				
gemma-4-31b	0.9279	0.9860	0.4128	0.4349
deepseek-v3.2	0.7114	0.9379	0.4409	0.3347
qwen3-235b-a22b-instruct-2507	0.6052	0.8798	0.2846	0.2685
qwen3-coder-480b-a35b-instruct	0.4048	0.7375	0.2184	0.1884

Table 18: Stage-level performance on Holmes-SQL. *Suspect Set* measures recovery of all four ground-truth suspects, and *Murderer* measures correct murder_id prediction. **Bold** indicates the best result in each column.

B.10 Stage-level Performance Analysis

We analyze model performance across investigation stages. As shown in Tab. 18, most models achieve high performance on crime and victim identification, but performance drops sharply when recovering the complete suspect set and identifying the murderer. This pattern reflects real-world investigations, where suspect-level reasoning is typically more difficult than the earlier stages of identifying the crime scene and victim.

B.11 Error Analysis

We analyze failed cases and report three observed error categories. Invalid final outputs account for 10.87% of failures, including missing or unparseable responses. In 58.85% of failed cases, the agent does not identify the complete ground-truth suspect set. The remaining 30.28% occur after the full suspect set has been recovered, indicating incorrect final selection among plausible suspects. This final category corresponds to the failure regime captured by FAR, where the agent has recovered the relevant candidates but still selects an innocent fallacy-aligned suspect.

Fig. 11 illustrates this pattern. GPT-5.4 prematurely commits to April Espinoza based on motive and a weak phone-call trace, reflecting

an inhibitory control failure (Bari and Robbins, 2013; Miyake et al., 2000), defined as the ability to change, withhold, or suppress learned or habitual responses in favor of more controlled, goal-appropriate ones. GLM-5 misweights Carolyn Robinson’s vehicle record against stronger clinical and spatial evidence. In contrast, Gemini-3.1-Pro succeeds by maintaining hypotheses, discounting weaker traces when contradicted by stronger constraints, and identifying Gary Johnson through spatiotemporal evidence near the murder scene.

C Dataset Documentation

C.1 Motivation

For what purpose was the dataset created? Collecting real investigative data is difficult because of privacy, legal, ethical, and institutional constraints. At the same time, LLM agents are increasingly being applied to complex database-driven tasks that require evidence retrieval, hypothesis tracking, and decision-making under uncertainty. The dataset was created as a testbed for studying evidence-aware investigation over structured relational databases in a controlled and reproducible setting.

C.2 Composition

What do the instances that comprise the dataset represent? Each instance represents a synthetic urban crime investigation encoded as a relational database. It includes a case description, fictional victim and suspect records, and investigation-related tables. Each instance tests whether an agent can retrieve relevant records, compare competing suspects, and identify the true murderer.

Does the dataset contain all possible instances or is it a sample (not necessarily random) of instances from a larger set? No, it does not contain all possible instances. The dataset is a

constructed sample of synthetic investigation cases generated from a controlled design space.

What data does each instance consist of? Each instance consists of a generated relational database and its artifacts. The artifacts include the case description, evidence checklist, reference SQL solution path, and ground-truth tuple identifiers used for evaluation.

Is there a label or target associated with each instance? Yes. Each instance includes ground-truth labels for the key investigation targets, including the crime report, victim, candidate suspects, and true murderer.

Is any information missing from individual instances? No, there is no information missing from individual instances.

Are relationships between individual instances made explicit (e.g., users' movie ratings, social network links)? No, relationships between individual instances are not explicitly defined. Each instance is treated as an independent investigation case without direct links to other instances.

Are there recommended data splits (e.g., training, development/validation, testing)? No, there are no recommended data splits. Users of the dataset can define their own splits based on their specific needs and goals.

Are there any errors, sources of noise, or redundancies in the dataset? Yes, there may be minor sources of noise or redundancy in the dataset. Although the dataset is synthetically generated and verified through script-level and database-level quality checks, some cases may still contain inconsistencies or ambiguities that can introduce noise despite our best efforts to minimize them.

Is the dataset self-contained, or does it link to or otherwise rely on external resources (e.g., websites, tweets, other datasets)? The dataset is self-contained and does not link to or rely on external resources such as websites, social media posts, or other datasets.

Does the dataset contain data that might be considered confidential (e.g., data that is protected by legal privilege or by doctor–patient confidentiality, data that includes the content of individuals' non-public communications)? No, the dataset does not contain any data that might be considered confidential. The dataset is fully synthetic,

and all people, locations, organizations, communications, medical records, and investigation records are fictional and created specifically for this dataset.

Does the dataset contain data that, if viewed directly, might be offensive, insulting, threatening, or might otherwise cause anxiety? The dataset does not intentionally contain any data that might be offensive, insulting, threatening, or anxiety-inducing. However, because the dataset is built around synthetic crime investigation scenarios, some instances may include crime-related or sensitive investigative themes that could be sensitive to some users. We have made efforts to review and filter the content to minimize potential issues. Additionally, if users identify any content they find problematic, we encourage them to report it to us, and we will take steps to replace or remove such content as necessary.

Does the dataset identify any subpopulations (e.g., by age, gender)? No

Is it possible to identify individuals (i.e., one or more natural persons), either directly or indirectly (i.e., in combination with other data) from the dataset? No, it is not possible to identify individuals, either directly or indirectly, from the dataset. The dataset consists entirely of fictional characters, locations, organizations, and synthetic investigation records, and does not include any real personal information that could be used to identify natural persons.

Does the dataset contain data that might be considered sensitive in any way (e.g., data that reveals race or ethnic origins, sexual orientations, religious beliefs, political opinions or union memberships, or locations; financial or health data; biometric or genetic data; forms of government identification, such as social security numbers; criminal history)? No. The dataset contains only fictional characters and synthetic crime investigation scenarios, without any real personal information or real criminal history. Suspect profiles are generated under controlled constraints to avoid introducing demographic biases related to race, gender, or religion.

C.3 Collection Process

How was the data associated with each instance acquired? The data for each instance was acquired through synthetic generation and verification. We first generated structured scripts from

controlled design templates covering urban themes, suspect profiles, evidence plans, timelines, and logical-fallacy patterns. Each approved script was then instantiated as a relational database, with corresponding ground-truth identifiers, evidence checklists, and reference SQL solution paths for evaluation.

What mechanisms or procedures were used to collect the data (e.g., hardware apparatuses or sensors, manual human curation, software programs, software APIs)? The data was created through software-based synthetic generation and manual human curation. We used generation scripts and LLM-assisted tools to produce raw data, while human reviewers checked, refined, and validated the outputs.

If the dataset is a sample from a larger set, what was the sampling strategy (e.g., deterministic, probabilistic with specific sampling probabilities)? The dataset is not sampled from a pre-existing larger population. Instead, it is constructed from a controlled synthetic design space. We use a systematic design strategy that combines predefined urban themes, logical-fallacy templates, suspect configurations, and evidence types to generate diverse candidate cases.

Who was involved in the data collection process (e.g., students, crowdworkers, contractors) and how were they compensated (e.g., how much were crowdworkers paid)? The dataset was constructed by the authors, who designed and generated the synthetic data artifacts. Two guideline-trained fourth-year IT student annotators were involved in script review and manual refinement during quality control. No crowdworkers or contractors were employed; the student annotators were compensated for their review work by the project team.

Over what timeframe was the data collected? The final dataset was constructed from February 2026 to April 2026.

Were any ethical review processes conducted (e.g., by an institutional review board)? No.

Did you collect the data from the individuals in question directly, or obtain it via third parties or other sources (e.g., websites)? We did not collect data from individuals directly, nor did we obtain it from third parties, websites, or external sources.

Were the individuals in question notified about the data collection? N/A.

Did the individuals in question consent to the collection and use of their data? N/A.

If consent was obtained, were the consenting individuals provided with a mechanism to revoke their consent in the future or for certain uses? N/A.

Has an analysis of the potential impact of the dataset and its use on data subjects (e.g., a data protection impact analysis) been conducted? No.

C.4 Preprocessing, Cleaning, and Labeling

Was any preprocessing/cleaning/labeling of the data done (e.g., discretization or bucketing, tokenization, part-of-speech tagging, SIFT feature extraction, removal of instances, processing of missing values)? Yes. We applied script-level and database-level quality-control checks to identify and refine problematic cases.

Is the software that was used to preprocess/clean/label the data available? No. The preprocessing, cleaning, and labeling scripts are not released with the current version of the dataset.

C.5 Uses and Distribution

Will the dataset be distributed to third parties? Yes. The dataset will be made publicly available upon publication of this research to encourage further study and reproducibility. The dataset will be released under the Creative Commons Attribution-NonCommercial 4.0 International (CC BY-NC 4.0) license.

D Use of AI Assistants

This work used large language models (LLMs) to assist in generating the synthetic dataset and to polish the original text written by human authors. All ideas, analyses, and conclusions were developed by the human authors, who take full responsibility for the content of the manuscript.