CHAINQABENCH: A BENCHMARK FOR EVIDENCE-GROUNDED QUESTION ANSWERING OVER BLOCKCHAIN DATA

Anonymous authors
Paper under double-blind review

ABSTRACT

Public blockchains make every transaction observable, yet even simple economic questions, such as how much an address gained, require combining many raw records. Doing so involves decoding the records into protocol actions, grouping them into analytical units, and judging what the result can establish. Large language models (LLMs) find these steps hard to perform directly on raw records, where values are hex-encoded and a single action spans several logs and traces. To address this, we introduce ChainQABench, to our knowledge the first evidence-grounded question-answering benchmark for blockchain data. Its 160 expert-curated questions cover asset flows, exchange attribution, value extraction through transaction ordering (MEV), and wash-trading patterns on decentralized exchanges. Five domain experts wrote rulebooks fixing admissible evidence and when a claim must remain indeterminate; an independent verifier reproduces every reference from its raw records. Over 16M transactions and 105M traces, we build an evidence hierarchy that organizes raw records into protocol actions, analytical units, and attribution context while preserving provenance links. We evaluate eleven LLMs as evidence moves from raw records to protocol actions, analytical units, and retrieved evidence. Even the strongest LLM, Gemini-3.8-Flash, achieves a maximum question accuracy of only 53.1% across these evidence conditions. Decoding raw records into protocol actions yields the most reliable gains, exposing typed quantities that LLMs otherwise struggle to reconstruct. We release the benchmark, evidence hierarchy, and independent verifier to support research on how evidence representations shape on-chain reasoning.

1 INTRODUCTION

Public blockchains make transactions and contract executions observable, but their records are rarely ready for economic analysis. A transaction may produce many logs and internal calls, and an individual transfer can be only one part of a larger activity (Schär et al., 2026). Even a simple question, such as how much an address gained over a block interval, therefore cannot be answered by reading records one at a time. Answering it first requires determining which records belong together and what they can establish (Schär et al., 2026).

In practice, on-chain analyses therefore proceed in stages: event logs are first decoded into protocol actions (Park et al., 2024), which are then grouped into analytical units tailored to the question (Qin et al., 2022; Schär et al., 2026). To calculate the net asset change of one or more addresses, for example, transfers within the analyzed set must be distinguished from flows that cross its boundary (Lubbertsen et al., 2025). Answering other questions may require reconstructing the order of several swaps (Wu et al., 2026). In each case, the interpretation depends on how records are related, grouped, and checked against their sources.

Large language models (LLMs) are increasingly applied to such analyses (Guo et al., 2025; Pan et al., 2026), yet whether they can carry out these stages themselves remains unclear. The task is not straightforward: raw values are hex-encoded and a single action can span several logs and traces, so an LLM must reconstruct each action before any economic reasoning can begin, and must also find

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

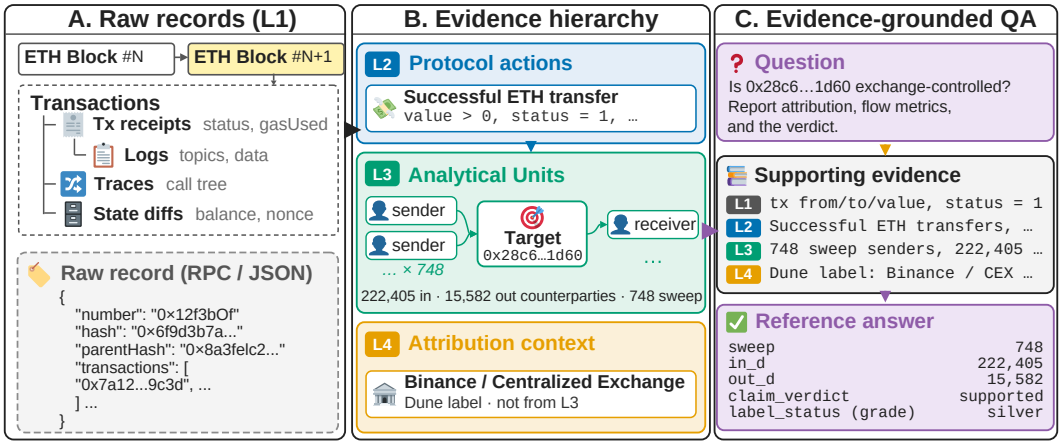


Figure 1: From raw records to evidence-grounded question answering. (A) Ethereum raw records feed (B) an evidence hierarchy of protocol actions, analytical units, and attribution context. (C) An exchange-attribution example connects question-specific evidence to a reference.

the records relevant to the question among many others. It is therefore also unknown which stages an LLM should perform itself and which should be supplied to it in advance.

Existing on-chain studies provide methods for address clustering and behavior-specific measurement (Victor, 2020; Qin et al., 2022; Gan et al., 2024), while recent benchmarks evaluate LLMs on cryptocurrency questions (Guo et al., 2025) and on translating natural-language intents into Ethereum transactions (Pan et al., 2026). Neither line of work can answer these questions, for two reasons. First, existing benchmarks do not provide verifiable references for claims about past activity, because they judge answers by rubric-based LLM grading or execution checks. Whether evidence supports such a claim depends on which sources are admissible, and a label or transaction pattern need not establish real-world ownership or intent (Lubbertsen et al., 2025; European Securities and Markets Authority, 2025); deciding it requires expert-defined rulebooks. Second, existing benchmarks do not provide on-chain evidence at multiple layers of abstraction, such as protocol actions and analytical units linked back to raw records; without such layers, one cannot measure which of them helps an LLM answer correctly.

To address both limitations, we introduce ChainQABench, to our knowledge the first evidence-grounded question-answering benchmark for blockchain data. It contains 160 questions over Ethereum mainnet data, covering asset flows, exchange attribution, value extraction through transaction ordering (MEV), and wash-trading patterns on decentralized exchanges. Our work makes two contributions, one for each limitation. **(1) An expert-built benchmark.** Five domain experts wrote a rulebook for each task that fixes admissible evidence, decision criteria, and when a claim must remain indeterminate, then curated every item from real on-chain cases and reviewed it against these rulebooks. An independent verifier reproduces all 160 references. **(2) An evidence hierarchy with provenance links.** From 16.34M transactions, 39.19M logs, and 105.23M traces, we decode raw records into protocol actions and group them into analytical units, each linked to the raw records needed to check it (Cheney et al., 2009). Holding questions and references fixed, we evaluate LLMs' ability to reason over on-chain evidence across different degrees of abstraction and assess how retrieving question-relevant evidence affects answer correctness.

We evaluate eleven LLMs, including four closed models and seven open-weight models, on ChainQABench under four evidence conditions. Even the strongest LLM, Gemini-3.8-Flash, achieves a maximum question accuracy of only 53.1%, highlighting the difficulty of answering on-chain questions correctly across all required fields. Providing protocol actions and analytical units can improve question accuracy, but the benefits vary across LLMs and tasks, and retrieval does not consistently improve performance. ChainQABench reveals where evidence preparation helps and where LLMs continue to struggle.

Table 1: Comparison with blockchain datasets and LLM benchmarks. QA: LLMs answer natural-language questions; Historical analysis: questions analyze recorded on-chain activity; Verification: how references are checked; Layers: evidence at multiple layers of abstraction; Links: answers are linked to supporting raw records. ✓: yes; △: partial; ✗: no.

	Task		References	Evidence	
Work	QA	Historical analysis	Verification	Layers	Links
On-chain datasets and analyses					
Graphs of Graphs (Luo et al., 2024)	✗	✓	Explorer tags	✓	✗
BlockScan (Yu et al., 2025)	✗	✓	Vendor labels	✗	✗
Ghost Clusters (Lubbertsen et al., 2025)	✗	✓	Seized records	✗	△
LLM benchmarks on crypto and on-chain data					
CryptoBench (Guo et al., 2025)	✓	△	Expert cross-validation	✗	△
EVM-QuestBench (Yang et al., 2026)	✓	✗	Fork execution	✗	✗
Intent2Tx (Pan et al., 2026)	✓	✗	Fork execution	✗	✗
Gebreab et al. (2026)	✓	△	Reference SQL	✗	△
ChainQABench (ours)	✓	✓	**Expert rulebooks**	✓	✓

2 On-Chain Data and Evaluation Requirements

2.1 Characteristics of On-Chain Data

On-chain analysis must link distributed raw records, construct analytical units above them, and choose a representation that exposes the relationships relevant to a question.

Distributed evidence across raw records. Unlike document or table-centric datasets, on-chain activity is not a single self-contained record: one transaction can involve nested contract calls, multiple logs, and asset transfers among several addresses. Analyzing the activity therefore requires first linking transactions, receipts, logs, and traces through their execution order and structural relations.

From raw records to analytical units. Analytical units organize related records into quantities, relationships, or patterns that no single record contains. Constructing them is not purely mechanical: it depends on domain-specific definitions and explicit analytical boundaries, and each unit must remain traceable to the raw records from which it is constructed.

Alternative representations of the same evidence. The same evidence can be organized into different representations, each exposing different relationships, aggregations, or interaction structures. Because a representation that highlights one structure may obscure information needed for another, which representation best exposes the relevant evidence depends on the question.

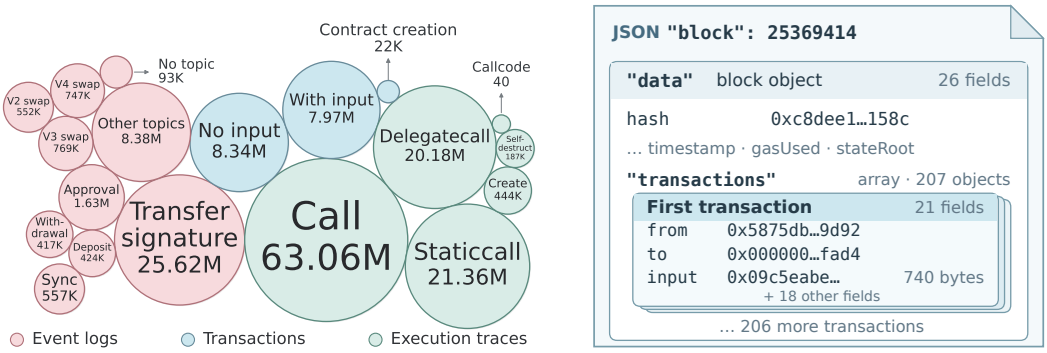
2.2 Related Benchmarks

Table 1 compares ChainQABench with two lines of prior work. On-chain datasets analyze recorded activity but do not evaluate LLMs, while LLM benchmarks pose questions that rarely concern recorded activity and are often verified by execution or an LLM judge. ChainQABench combines the two and adds expert rulebooks and layered evidence. Appendix B provides additional related work.

3 Benchmark Design for Verifiable On-Chain Analysis

3.1 Motivation

To meet these evaluation requirements, we first freeze the snapshot and define analytical rulebooks, then construct benchmark items and independently verify their references. We next organize the source evidence into an evidence hierarchy and define the representation and access conditions used to evaluate LLMs.



(a) Raw-record composition (b) Nested structure of a block-query result

Figure 2: Overview of the Ethereum snapshot used in the benchmark. (a) Composition of transactions, logs, and traces. (b) Nested structure of a block-query result stored in the collector's block/data wrapper, with fields and long values abbreviated.

3.2 Data Sources and Scope

The benchmark draws on two frozen sources: on-chain records from Ethereum JSON-RPC[1] and a Dune label snapshot[2] for attribution.

On-chain records. All benchmark items are constructed and verified on a fixed snapshot of Ethereum mainnet. The snapshot spans blocks 25,369,414–25,419,597 (June 22–28, 2026 UTC), which we call the observation window, and includes full transactions, receipts, logs, traces, and permitted historical-state responses. Although temporally bounded, it contains 16.34M transactions, 39.19M logs, and 105.23M traces. Figure 2 shows how these records divide, with traces dominated by 63.06M calls and logs by 25.62M logs matching the Transfer event signature, and the nested structure of one stored block-query result.

Attribution labels. Because real-world attribution is not recorded in the ledger, we use the Dune label snapshot only where the corresponding rulebook permits, and keep it separate from ledger evidence. Appendix C details source collection and integrity checks.

3.3 Benchmark Item Construction and Validation

Expert-defined analytical rulebooks. On-chain questions are not always resolved by the raw records alone, as their answers can depend on which evidence is admissible and how the analytical boundary is defined. We therefore develop an *expert-defined analytical rulebook* for each task before constructing benchmark items. Five domain experts jointly develop and review these rulebooks, which specify the admissible evidence, required calculations, decision criteria, and cases that should remain *indeterminate* or outside the analytical boundary. By fixing these criteria in advance, the benchmark turns otherwise ambiguous analytical questions into *reproducible evaluation targets*. The task-specific rulebooks are documented in Appendix D.

Expert-curated benchmark items. Using the fixed Ethereum snapshot, the domain experts identify real cases that satisfy the task definitions and turn them into benchmark items. Each item is reviewed for task relevance, clarity, and whether the requested answer is supported by the specified evidence. Items that do not satisfy these criteria are revised or excluded.

Reference construction and verification. For each approved item, we derive the reference from the same snapshot under its corresponding rulebook. An independent verifier reproduces the required calculations and rule-based decisions before release, keeping the references independent of

[1]Retrieved from the public archive-node endpoint of PublicNode (`https://ethereum-rpc.publicnode.com`) in July 2026.

[2]Dune Analytics (`https://dune.com`), query results retrieved July–September 2026.

Table 2: Benchmark tasks, evidence requirements, and analytical boundaries. The four tasks differ in their evaluation targets, the evidence and reasoning required to answer them, and the analytical boundaries within which that evidence supports conclusions. C / A: numbers of computational and adjudicative items; Fields: required fields.

Task	Evaluation target	Evidence and reasoning	Inference scope	C / A	Fields
T1: Asset Flows	ERC-20 asset flows and holdings changes for a specified address set and block interval	Boundary-aware transfer aggregation, asset-specific accounting, and permitted state queries	Inflow, outflow, and net change under the declared accounting rules	16 / 24	231
T2: Exchange Attribution	Attribution of an address or address set to a controlling or operating entity	Address relations combined with admissible dated labels	Separates structural association, common control, exchange role, and named attribution	9 / 31	185
T3: MEV-Related Interactions[3]	Computations and judgments on sandwich pairs and arbitrage or liquidation transactions	Execution order, trading routes, asset flows, and protocol events	Pattern-specific judgment under explicit operational conditions	12 / 28	262
T4: AMM Wash-Trading Patterns[4]	Swap-group pattern analysis and pool/window distribution screening	Ordered swaps, flow closure, actor relations, alternative explanations, and distribution statistics	Operational pattern classification, not intent or legal wrongdoing	21 / 19	265
Total				58 / 102	943

the evaluated LLMs and evidence conditions. Appendix E details the reference construction and verification procedure.

3.4 Tasks and Benchmark Composition

Task coverage. The benchmark covers four complementary forms of on-chain analysis: asset flows, exchange attribution, MEV-related interactions, and AMM wash-trading patterns. We select these tasks to span different combinations of *evidence and reasoning requirements*, without claiming to cover all forms of blockchain analysis. Table 2 summarizes each task's evaluation target, evidence and reasoning requirements, and *analytical boundary*.

Benchmark composition. The resulting benchmark contains 160 items, balanced with 40 items in each of the four tasks. The collection includes both *computational* and *adjudicative* items and spans multiple answer formats and reasoning requirements. Figure 3 describes its substantive coverage using 11 item-level topic families. Each topic is grounded in a recurring economic activity or analytical concept in on-chain data. A single question may involve multiple topics. For example, a question about a change in holdings may require transfer accounting and an asset-specific rule at the same time. Appendices F and G provide detailed benchmark composition and example questions with references.

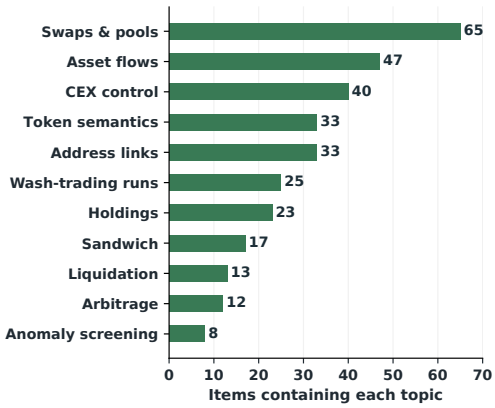


Figure 3: Topic families in the benchmark.

3.5 Evidence Hierarchy and Evidence Conditions

Controlled evidence representations. We compare evidence conditions within each LLM while holding the snapshot, questions, rulebooks, and references fixed. The first three conditions expose

[3]T3 includes sandwich interactions, cyclic arbitrage, and liquidation, commonly studied in maximal extractable value (MEV) analysis.

[4]T4 evaluates an operationalized wash-trading pattern in automated market maker (AMM) pools. Distribution screening alone does not establish a structural verdict.

Table 3: Evidence hierarchy constructed for the benchmark. L1 through L3 form the evidence hierarchy from raw records to protocol actions and analytical units, while L4 provides complementary attribution context. Derived objects retain provenance links to the lower-layer evidence supporting their construction.

Layer	Representative objects	Construction	Supports claims about
L1: Raw Records	Transactions, receipts, logs, traces, state responses	JSON-RPC collection	Recorded execution and ordering
L2: Protocol Actions	Transfers, swaps, mint/burn, liquidation	Decoding	Protocol-level actions
L3: Analytical Units	Net flows, state checkpoints, interaction groups, cycles, bundles	Rulebook-defined grouping	Quantities and interaction structures under explicit analytical boundaries
L4: Attribution Context	Structural relations, address-entity relations, dated labels	Structural inference and label snapshots	Attribution within the available source and temporal scope

progressively richer analytical structure; the fourth applies retrieval over the same evidence hierarchy.

Evidence hierarchy. Table 3 summarizes how we organize on-chain evidence into progressively more structured representations, moving from recorded execution to protocol-level actions and then to analytical units. Here, *raw* denotes source-level RPC outputs before benchmark-specific decoding or analytical grouping, including client-derived traces and historical-state responses. Starting from **L1: Raw Record**s, decoding interprets low-level execution to construct **L2: Protocol Action**s. We construct **L3: Analytical Units** from L1 records, L2 actions, or both through grouping and computation under explicit analytical boundaries, such as address sets, assets, time intervals, or transaction groups. **L4: Attribution Context** is complementary to this progression rather than another layer of abstraction, providing admissible structural relations and dated external labels when attribution is required. Throughout the construction process, each derived object retains provenance links to the lower-layer evidence needed to inspect or recompute it.

Evidence conditions. Table 4 summarizes the four evidence conditions we compare over the same benchmark items. Each item has a fixed execution path across conditions: 110 items receive evidence in context, and 50 query condition-specific tables through SQL. RAW provides raw records and admissible attribution context. +ACTIONS adds protocol actions, and +UNITS adds analytical units; both supply all evidence within the item's analytical boundary. +RETRIEVAL uses the same evidence hierarchy as +UNITS but retrieves only question-relevant evidence for in-context items. For SQL items, +UNITS and +RETRIEVAL use the same tables and schema, and the reported comparison reuses the same responses for these 50 items. Their difference therefore concerns retrieval for the 110 in-context items.

Table 4: Four evidence conditions, each building on the one above. ✓: layers given to the LLM (L4 only where the rulebook admits it). Illustrative evidence: what the LLM sees for a T1 net-flow question.

Condition	L1	L2	L3	L4	Access mechanism	Illustrative evidence (T1)
RAW	✓			✓	Full analytical boundary	Hex-encoded log fields and traces
+ACTIONS	✓	✓		✓	Full analytical boundary	Decoded transfers with token and amount
+UNITS	✓	✓	✓	✓	Full analytical boundary	Transfers grouped by address, asset, and interval
+RETRIEVAL	✓	✓	✓	✓	Question-relevant subset	Question-relevant transfers and flows only

3.6 EVALUATION PROTOCOL

Evaluation setup. Each LLM-condition score covers all 160 items, with questions, rulebooks, required fields, and references fixed. Appendix H presents the task-specific system prompts. Evidence content and available table schemas depend on the condition. Appendix I provides examples of execution-time system-message assembly. These comparisons evaluate the specified evidence conditions rather than isolating the evidence hierarchy, prompting, and tool use as independent effects.

Answer scoring. The scoring specification fixes required fields, types, and tolerances. Integers are compared exactly in the declared unit; address strings are compared after case normalization, with no other transformation applied. Lists are compared in order when order is semantically meaningful, records must contain every required key, and decimal fields are compared using their declared tolerances. Parsing failures and missing required fields are incorrect. The main results use relaxed parsing, which accepts a valid field mapping without the outer answer wrapper; strict scoring treats such responses as unanswered. The primary metrics are question accuracy and micro-averaged field accuracy. Question accuracy requires all required fields to be correct; field accuracy measures correctness across the required fields. Detailed metric definitions and output-format audits are provided in Appendix J.

4 Experimental Results

4.1 Main Results

Settings. We study how well LLMs can analyze recorded on-chain activity and which stages of evidence preparation they can perform themselves or benefit from receiving in advance. To evaluate LLM performance on ChainQABench, we test four closed models and seven open-weight models under the four evidence conditions, examining how their ability to answer correctly changes across Raw, +Actions, +Units, and +Retrieval.

The closed models are GPT-5.4-mini and GPT-5.6-terra (OpenAI, 2026a;b), and Gemini-3.1-Flash-Lite and Gemini-3.8-Flash (Google DeepMind, 2026a;b). The open-weight models are Gemma4-12B and Gemma4-31B (Google DeepMind, 2026c), Qwen3-VL-8B-Instruct (Qwen Team, 2025b) and Qwen3-30B-Instruct (Qwen Team, 2025a), Qwen3.5-9B and Qwen3.8-27B (Qwen Team, 2026a;b), and Qwen3.8-27B-Finance (nico248000000000, 2026).

Table 5: Question accuracy and field accuracy (%) under the four evidence conditions. Bold and underlined values indicate the best and second-best results in each column, respectively.

LLM	Raw		+Actions		+Units		+Retrieval	
	Question	Field	Question	Field	Question	Field	Question	Field
Closed models								
GPT-5.4-mini	11.2	46.6	15.0	49.7	11.9	47.5	15.6	48.9
GPT-5.6-terra	26.2	59.7	31.9	63.3	35.0	66.5	33.1	64.2
Gemini-3.1-Flash-Lite	20.0	52.1	23.8	57.2	26.9	58.5	27.5	59.5
Gemini-3.8-Flash	**53.1**	**77.4**	**53.1**	**79.3**	**51.2**	**79.4**	**50.0**	**76.2**
Open-weight models								
Gemma4-12B	13.1	29.6	13.1	34.7	11.2	37.3	12.5	37.6
Gemma4-31B	23.1	53.8	30.6	60.1	33.1	62.4	33.1	62.8
Qwen3-VL-8B-Instruct	5.0	29.9	5.6	35.9	6.2	34.4	7.5	42.6
Qwen3-30B-Instruct	10.0	29.1	10.0	41.2	8.8	35.5	10.6	42.2
Qwen3.5-9B	10.6	41.9	10.6	43.3	9.4	40.7	15.6	48.3
Qwen3.8-27B	32.5	63.4	35.0	62.7	36.2	65.0	35.6	65.0
Qwen3.8-27B-Finance	<u>35.6</u>	<u>64.1</u>	<u>38.8</u>	<u>67.9</u>	<u>40.6</u>	<u>68.2</u>	<u>40.6</u>	<u>66.7</u>

ChainQABench remains challenging. Gemini-3.8-Flash achieves the highest question accuracy at 53.1%, while the other ten LLMs remain below 41% across all conditions (Table 5). Field accuracy is substantially higher, reaching 79.4%, showing that partial correctness is considerably higher than full-answer correctness.

+Actions provides the most consistent gains. Moving from Raw to +Actions improves question accuracy for seven LLMs and degrades none. Further evidence structuring does not yield monotonic gains: +Units improves six LLMs and degrades five, while +Retrieval improves six and degrades three. The effect also varies substantially across LLMs; Gemma4-31B increases from 23.1% under Raw to 33.1% under +Units, whereas Gemini-3.8-Flash does not improve beyond its Raw performance.

4.2 How does performance vary across tasks and evidence conditions?

T3 shows the largest improvement, primarily under +Actions. Figure 4 compares mean question accuracy across 11 LLMs for each task and evidence condition. T3 shows the largest increase from Raw to +Units (16.1% to 25.0%, +8.9 pp). Its MEV-related judgments require connecting transaction order, trading routes, and asset flows across multiple transactions. +Actions makes protocol actions and quantities explicit, while +Units further organizes related actions within the item's analytical boundary. Most of the improvement is already observed under +Actions (+8.0 pp relative to Raw).

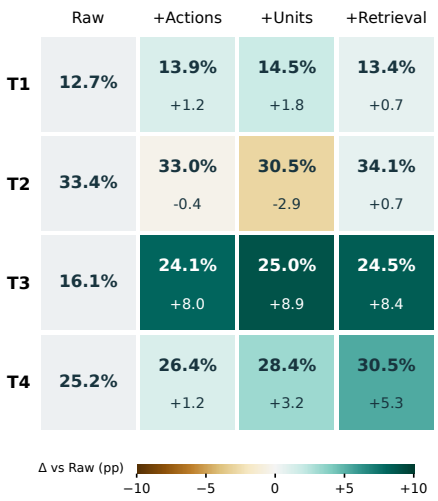


Figure 4: Mean strict question accuracy

Evidence conditions have task-dependent effects. T1 has the lowest mean question accuracy, remaining below 15% in all conditions, and shows only a small gain under +Units (+1.8 pp). Its accounting requirements within the analytical boundary therefore remain challenging even when more structured evidence is provided. T2 depends on attribution context that is complementary to the on-chain evidence hierarchy; additional organization of on-chain activity alone does not establish control or identity. Its mean question accuracy decreases from 33.4% under Raw to 30.5% under +Units, before increasing to 34.1% under +Retrieval. T4, whose judgments depend on ordered swaps, flow closure, and actor relations, increases from 25.2% to 30.5% across the four conditions.

Taken together, these results show that no single evidence representation consistently improves all tasks. Instead, the usefulness of evidence preparation depends on the reasoning requirements of the task. ChainQABench therefore enables comparison of where different layers of evidence structure are useful and where LLMs remain limited despite additional evidence preparation.

4.3 Where does +Actions help across topics?

The effect of +Actions varies across topics. Figure 5 compares the change in question accuracy from Raw to +Actions across the eleven topics in Figure 3. Question accuracy improves for ten of eleven LLMs on liquidation, eight on asset flows, and seven on sandwich questions, but for only two on arbitrage and two on anomaly screening. The largest gains occur where decoded protocol events and quantities reduce record-level interpretation, whereas smaller gains arise when answering requires further composition or aggregation over multiple actions.

Representation effects depend on reasoning requirements. These results show that the usefulness of an evidence representation depends on the reasoning required by the topic. By comparing the same questions across evidence conditions, ChainQABench reveals where additional evidence structure is useful and where LLMs remain limited despite it. Topic memberships overlap and

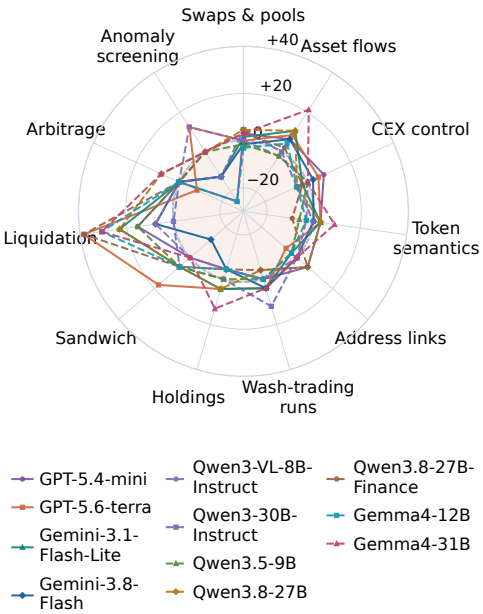


Figure 5: Strict question accuracy change from Raw to +Actions (pp).

Table 6: Qwen3.8-27B-Finance on one T3 sandwich item. The same question is answered correctly only once the evidence exposes what it requires: decoding fixes the event counts, and the analytical unit supplies the relation between the swaps. Background colors link the five displayed fields across the reference, excerpts, and submitted values; ✓/× mark correct/incorrect values. Correct counts all 10 required fields. Quotes are verbatim; … marks omissions.

Question (paraphrased). Do two swaps in a Uniswap V3 pool form a sandwich attack? Report whether another trader's swap lies between them at a worse price (τ=0.005), how many liquidity events the front and back transactions contain, and the verdict.
Reference. victim true; worse at τ true; Mint 1; Burn/Collect 2; verdict supported. The other five fields are correct in every condition.

Condition	Evidence, reasoning, and submitted values	Correct
RAW	*Evidence (raw RPC record):* "…"topics": ["0x0c396cd9…", …" *LLM:* "…logIndex 0x28 (40): Mint event from 0xc1df…bb1b…" "…The evidence is scoped to only these 2 transactions.… No third-party same-direction swap exists…" ⇒ victim false×; worse at τ false×; Mint 0×; Burn/Collect 1×; verdict contradicted×	5/10
+ACTIONS	*Evidence (protocol action):* "…univ3 pool FORTH/WETH(0xc1df…bb1b) liquidity added…" *Evidence (ledger row):* "…log #40: … emitted event V3_STYLE Burn…" ⇒ victim false×; worse at τ false×; Mint 1✓; Burn/Collect 2✓; verdict contradicted×	7/10
+UNITS	*Evidence (analytical unit):* "sandwich adjudication unit (ordered pair): …swap #6 … and #8 …, 1 swaps in between…" *LLM:* "…The intermediate swap #7 is by sender 0xfe0c…f65a…" ⇒ victim true✓; worse at τ true✓; Mint 1✓; Burn/Collect 2✓; verdict supported✓	10/10
+RETRIEVAL	*LLM:* "…The swap between them is #7 (node 437)… The price difference ratio is 0.01888 (1.888%), which is well above τ = 0.005 (0.5%)…." ⇒ victim true✓; worse at τ true✓; Mint 1✓; Burn/Collect 2✓; verdict supported✓	10/10

range from 8 to 65 items, so these results describe overlapping subsets rather than independent tasks.

5 QUALITATIVE ANALYSIS

Case study of a T3 sandwich item. Section 4 reports aggregate performance changes across evidence conditions. Here, we examine one T3 sandwich item to illustrate how the available evidence and the correctness of individual required fields change across the four conditions. We select T3 because it shows the largest aggregate improvement, and analyze the responses of Qwen3.8-27B-Finance in Table 6.

Evidence layers expose complementary information. Under RAW, the LLM must infer event types from raw signature hashes and misidentifies a liquidity event, producing incorrect event counts. It also fails to identify the intermediate swap required by the sandwich rule, resulting in incorrect relation fields and verdict. Under +ACTIONS, explicit protocol events correct the Mint and Burn/Collect counts, but the relation between transactions remains unresolved. +UNITS makes this relation explicit as an analytical unit, allowing the LLM to identify the intermediate swap, compare prices, and answer all fields correctly. +RETRIEVAL preserves the same conclusion with a narrower evidence set.

This example illustrates the distinct roles of the evidence layers: protocol actions expose the semantics of individual events, whereas analytical units expose relations among them. More broadly, the rulebook-defined fields and layered evidence in ChainQABench enable inspection of which required information becomes explicit across conditions and which answer errors persist.

6 CONCLUSION

We introduced ChainQABench, an evidence-grounded benchmark of 160 on-chain questions with expert-defined rulebooks and independently verified references. Across eleven LLMs, decoding improves or preserves question accuracy, while grouping and retrieval show mixed effects. ChainQABench reveals where evidence preparation helps and where reasoning failures remain.

AI USE STATEMENT

Generative AI tools assisted with drafting and revising the manuscript, refining its language and presentation, and editing software code. The authors take responsibility for the final manuscript, code, and other artifacts, including all scientific claims and conclusions.

ETHICS STATEMENT

This research was conducted in accordance with the ICLR Code of Ethics. ChainQABench uses publicly accessible Ethereum records and a Dune label snapshot. Public availability does not eliminate privacy risks: address linkage and external labels can enable profiling of users or entities. We therefore restrict attribution to the admissible evidence and temporal constraints specified by the task rulebooks. Structural associations and external labels do not independently establish real-world identity, ownership, intent, or wrongdoing (Appendices A.3, A.4, and C.4). The benchmark is intended for evaluating on-chain reasoning; it should not be used for deanonymization or unsupported allegations about individuals or entities.

REPRODUCIBILITY STATEMENT

Section 3 specifies the Ethereum observation window, benchmark construction, and evidence conditions. Appendix C documents the RPC collection methods, stored record formats, Dune label snapshot, provenance links, and source-integrity checks. Appendices D, E, and F provide the task rulebooks, reference construction and independent verification procedures, and benchmark composition. Section 4 identifies the evaluated LLMs, while Section 3.6 and Appendices H, I, and J describe the evaluation protocol, task-specific prompts, examples of execution-time system-message assembly, and scoring criteria. Appendix M reports mean input and output tokens per item for the closed models and GPU assignments for the open-weight models.

REFERENCES

Yael Amsterdamer, Daniel Deutch, and Val Tannen. Provenance for aggregate queries. In *Proceedings of the 30th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pp. 153–164. Association for Computing Machinery, 2011. doi: 10.1145/1989284.1989302.

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan Routledge, and William Yang Wang. FinQA: A dataset of numerical reasoning over financial data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 3697–3711, 2021. doi: 10.18653/v1/2021.emnlp-main.300.

James Cheney, Laura Chiticariu, and Wang-Chiew Tan. Provenance in databases: Why, how, and where. *Foundations and Trends in Databases*, 1(4):379–474, 2009. doi: 10.1561/1900000006.

Lin William Cong, Xi Li, Ke Tang, and Yang Yang. Crypto wash trading. *Management Science*, 69 (11):6427–6454, 2023. doi: 10.1287/mnsc.2021.02709.

Yingwei Cui and Jennifer Widom. Lineage tracing for general data warehouse transformations. In *Proceedings of the 27th International Conference on Very Large Data Bases*, pp. 471–480, 2001. URL https://www.vldb.org/conf/2001/P471.pdf.

European Securities and Markets Authority. Maximal extractable value: Implications for crypto markets. ESMA TRV Risk Analysis ESMA50-481369926-29744, European Securities and Markets Authority, July 2025.

Rundong Gan, Le Wang, Liang Xue, and Xiaodong Lin. Exposing stealthy wash trading on automated market maker exchanges. *ACM Transactions on Internet Technology*, 24(4), 2024. doi: 10.1145/3689631.

S. Gebreab, A. Musamih, K. Salah, and R. Jayaraman. LLM-based exploration and analysis of real-time and historical blockchain data. *Expert Systems with Applications*, 298:129851, 2026. doi: 10.1016/j.eswa.2025.129851.

Google DeepMind. Gemini 3.1 Flash-Lite Model Card. https://deepmind.google/models/model-cards/gemini-3-1-flash-lite/, 2026a. Accessed September 26, 2026.

Google DeepMind. Gemini 3.8 Flash Model Card. https://deepmind.google/models/model-cards/gemini-3-8-flash/, 2026b. Accessed September 26, 2026.

Google DeepMind. Gemma 4 Model Card. https://ai.google.dev/gemma/docs/core/model_card_4, 2026c. Accessed September 26, 2026.

Jiacheng Guo, Suozhi Huang, Zixin Yao, Yifan Zhang, Yifu Lu, Jiashuo Liu, Zihao Li, Nicholas Deng, Qixin Xiao, Jia Tian, Kanghong Zhan, Tianyi Li, Xiaochen Liu, Jason Ge, Chaoyang He, Kaixuan Huang, Lin Yang, Wenhao Huang, and Mengdi Wang. CryptoBench: A dynamic benchmark for expert-level evaluation of LLM agents in cryptocurrency. *arXiv preprint arXiv:2512.00417*, 2025.

Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh V. Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. G-Retriever: Retrieval-augmented generation for textual graph understanding and question answering. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Nan Huo, Xiaohan Xu, Jinyang Li, Per Jacobsson, Shipei Lin, Bowen Qin, Binyuan Hui, Xiaolong Li, Ge Qu, Shuzheng Si, Linheng Han, Edward Alexander, Xintong Zhu, Rui Qin, Ruihan Yu, Yiyao Jin, Feige Zhou, Weihao Zhong, Yun Chen, Hongyu Liu, Chenhao Ma, Fatma Ozcan, Yannis Papakonstantinou, and Reynold Cheng. BIRD-INTERACT: Re-imagining Text-to-SQL Evaluation via Lens of Dynamic Interactions. In *International Conference on Learning Representations*, 2026.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. SWE-bench: Can Language Models Resolve Real-World GitHub Issues? In *International Conference on Learning Representations*, 2024.

George Kappos, Haaroon Yousaf, Rainer Stütz, Sofia Rollet, Bernhard Haslhofer, and Sarah Meiklejohn. How to peel a million: Validating and expanding Bitcoin clusters. In *31st USENIX Security Symposium (USENIX Security 22)*, pp. 2207–2223. USENIX Association, 2022. URL https://www.usenix.org/conference/usenixsecurity22/presentation/kappos.

Polina Kirichenko, Mark Ibrahim, Kamalika Chaudhuri, and Samuel J. Bell. AbstentionBench: Reasoning LLMs Fail on Unanswerable Questions. *arXiv preprint arXiv:2506.09038*, 2025.

Eugenie Lai, Gerardo Vitagliano, Ziyu Zhang, Om Chabra, Sivaprasad Sudhir, Anna Zeng, Anton A. Zabreyko, Chenning Li, Ferdi Kossmann, Jialin Ding, Jun Chen, Markos Markakis, Matthew Russo, Weiyang Wang, Ziniu Wu, Michael J. Cafarella, Lei Cao, Samuel Madden, and Tim Kraska. KramaBench: A Benchmark for AI Systems on Data-to-Insight Pipelines over Data Lakes. In *International Conference on Learning Representations*, 2026.

Kelvin Lubbertsen, Michel van Eeten, and Rolf van Wegberg. Ghost clusters: Evaluating attribution of illicit services through cryptocurrency tracing. In *34th USENIX Security Symposium*, pp. 1357–1374, 2025.

Bingqiao Luo, Zhen Zhang, Qian Wang, and Bingsheng He. Multi-chain graphs of graphs: A new approach to analyzing blockchain datasets. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

nico248000000000. Qwen3.8-27B-finance Model Card. https://huggingface.co/nico248000000000/Qwen3.8-27B-finance, 2026. Accessed September 26, 2026.

Curtis Northcutt, Anish Athalye, and Jonas Mueller. Pervasive label errors in test sets destabilize machine learning benchmarks. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/f2217062e9a397a1dca429e7d70bc6ca-Abstract-round1.html.

OpenAI. Introducing GPT-5.4 mini and nano. https://openai.com/index/introducing-gpt-5-4-mini-and-nano/, 2026a. Accessed September 26, 2026.

OpenAI. GPT-5.6 System Card. https://deploymentsafety.openai.com/gpt-5-6, 2026b. Accessed September 26, 2026.

Zhuoran Pan, Yue Li, Zhi Guan, Jianbin Hu, and Zhong Chen. Intent2Tx: Benchmarking LLMs for Translating Natural Language Intents into Ethereum Transactions. *arXiv preprint arXiv:2604.27763*, 2026.

Seongwan Park, Woojin Jeong, Yunyoung Lee, Bumho Son, Huisu Jang, and Jaewook Lee. Unraveling the MEV enigma: ABI-free detection model using graph neural networks. *Future Generation Computer Systems*, 153:70–83, 2024. doi: 10.1016/j.future.2023.11.014.

Kaihua Qin, Liyi Zhou, and Arthur Gervais. Quantifying Blockchain Extractable Value: How dark is the forest? In *IEEE Symposium on Security and Privacy*, 2022.

Qwen Team. Qwen3-30B-A3B-Instruct-2507 Model Card. https://huggingface.co/Qwen/Qwen3-30B-A3B-Instruct-2507, 2025a. Accessed September 26, 2026.

Qwen Team. Qwen3-VL-8B-Instruct Model Card. https://huggingface.co/Qwen/Qwen3-VL-8B-Instruct, 2025b. Accessed September 26, 2026.

Qwen Team. Qwen3.5-9B Model Card. https://huggingface.co/Qwen/Qwen3.5-9B, 2026a. Accessed September 26, 2026.

Qwen Team. Qwen3.8-27B Model Card. https://huggingface.co/Qwen/Qwen3.8-27B, 2026b. Accessed September 26, 2026.

Fabian Schär, Anneke Kosse, Tara Rice, Takeshi Shirakami, and Jirapat Siridhasanakul. The anatomy of stablecoin transactions. BIS Working Papers 1359, Bank for International Settlements, 2026.

Issa Sugiura, Takashi Ishida, Taro Makino, Chieko Tazuke, Takanori Nakagawa, Kosuke Nakago, and David Ha. EDINET-Bench: Evaluating LLMs on Complex Financial Tasks using Japanese Financial Statements. In *International Conference on Learning Representations*, 2026.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 809–819, 2018. doi: 10.18653/v1/N18-1074.

Friedhelm Victor. Address clustering heuristics for ethereum. In *Financial Cryptography and Data Security*, pp. 617–633, 2020. doi: 10.1007/978-3-030-51280-4_33.

Colin White, Samuel Dooley, Manley Roberts, Arka Pal, Ben Feuer, Siddhartha Jain, Ravid Shwartz-Ziv, Neel Jain, Khalid Saifullah, Sreemanti Dey, Shubh-Agrawal, Sandeep Singh Sandha, Siddartha Naidu, Chinmay Hegde, Yann LeCun, Tom Goldstein, Willie Neiswanger, and Micah Goldblum. LiveBench: A Challenging, Contamination-Limited LLM Benchmark. In *International Conference on Learning Representations*, 2025.

Changhao Wu, Luyu Chen, Kai Wang, Weili Han, and Hongfeng Chai. SandWatch: Towards detecting sandwich attacks in Ethereum using a dual-task graph neural network. *ACM Transactions on the Web*, 2026. doi: 10.1145/3787205.

Pei Yang, Wanyi Chen, Ke Wang, Lynn Ai, Eric Yang, and Tianyu Shi. EVM-QuestBench: An execution-grounded benchmark for natural-language transaction code generation. *arXiv preprint arXiv:2601.06565*, 2026.

Jiahao Yu, Xian Wu, Hao Liu, Wenbo Guo, and Xinyu Xing. BlockScan: Detecting Anomalies in Blockchain Transactions. In *Advances in Neural Information Processing Systems*, 2025.

Fengbin Zhu, Wenqiang Lei, Youcheng Huang, Chao Wang, Shuo Zhang, Jiancheng Lv, Fuli Feng, and Tat-Seng Chua. TAT-QA: A Question Answering Benchmark on a Hybrid of Tabular and Textual Content in Finance. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 3277–3287, 2021. doi: 10.18653/v1/2021.acl-long.254.

Fengbin Zhu, Junfeng Li, Liangming Pan, Wenjie Wang, Fuli Feng, Chao Wang, Huanbo Luan, and Tat-Seng Chua. Towards Temporal-Aware Multi-Modal Retrieval Augmented Generation in Finance. In *Proceedings of the 33rd ACM International Conference on Multimedia*, 2025.

APPENDIX CONTENTS

A Blockchain and Ethereum Background **15**
A.1 Accounts, transactions, and contract execution . . . 15
A.2 Tokens, protocols, and DeFi actions . . . 15
A.3 Addresses, entities, and labels . . . 15
A.4 Observable evidence and interpretive limits . . . 15
A.5 Evidence objects used in this benchmark . . . 16

B Additional Related Work **16**

C Data Sources, Validation, and Attribution **17**
C.1 Source representations and frozen records . . . 17
C.2 Actual downloaded records and provenance . . . 17
C.3 Source-integrity validation and exclusion rules . . . 19

D Task Rulebooks **21**

E Reference Construction Algorithm **34**

F Benchmark and Answer Composition **34**

G Task-wise Question and Answer Examples **37**

H Task-Specific Experimental Prompts **40**

I Examples of Execution-Time System-Message Assembly **56**

J Evaluation Metrics and Scoring Views **56**

K Additional Quantitative Analysis **57**
K.1 Scoring audit and auxiliary metrics . . . 57
K.2 Task results . . . 58
K.3 Topics from Figure 3 . . . 59
K.4 Difficulty levels . . . 61
K.5 Execution paths . . . 61
K.6 Answer recoveries and regressions . . . 62
K.7 Capability families . . . 62

L Additional Qualitative Analysis **63**

M Token Usage and Hardware **66**

A BLOCKCHAIN AND ETHEREUM BACKGROUND

This section introduces only the concepts needed to interpret the data, claims, and evidence used in the benchmark. A blockchain is an append-only ledger replicated by a network. Transactions are grouped into ordered blocks, and each accepted block commits to both its predecessor and the resulting ledger state. The block number therefore identifies a position in one canonical history, while a transaction describes an attempted state transition at that position. The observation window used throughout this paper is the inclusive Ethereum mainnet interval from block 25,369,414 through block 25,419,597 (2026-06-22–2026-06-28 UTC).

A.1 ACCOUNTS, TRANSACTIONS, AND CONTRACT EXECUTION

Ethereum uses an account model. An externally owned account (EOA) is controlled by a private key; a contract account (CA) executes deployed bytecode when called. Both are identified by 20-byte addresses. A transaction names a sender, optional destination, native-asset value, and input bytes. A call to a contract can trigger nested calls, create other contracts, transfer assets, and update state. The transaction sender, the contract receiving a call, the contract emitting an event, and the economic actor inferred from several transactions can therefore be different addresses.

A successful or reverted execution is summarized by a receipt. Receipts contain status, gas use, and ordered logs. A log records the emitting contract, indexed topics, unindexed data, transaction coordinates, and a log index. Traces expose internal calls, contract creation, and native asset movement that do not appear as top-level transactions. Traces are client-derived execution views rather than separately committed consensus objects, so the benchmark records their source and validation status.

A.2 TOKENS, PROTOCOLS, AND DEFI ACTIONS

Ether (ETH) is Ethereum's native asset. Fungible tokens are usually contract managed and commonly expose ERC-20 `Transfer` events; token quantities must be interpreted in the token's integer base unit. A native ETH transfer may appear in a transaction or internal trace, whereas an ERC-20 transfer is normally reconstructed from a token-contract log. Wrapped ETH (WETH) and some protocol-specific accounting systems require additional event handling.

A decentralized exchange (DEX) lets users exchange assets through smart contracts. An automated market maker (AMM) holds liquidity in a pool and updates its reserves or accounting state as swaps execute. Lending protocols let users supply collateral, borrow assets, and liquidate undercollateralized positions. Routers and manager contracts may coordinate many pools, so a contract address alone does not always identify the relevant pool. This is why the benchmark retains protocol-specific pool identifiers and transaction/log order when reconstructing swaps, arbitrage, liquidation, and AMM interaction patterns.

A.3 ADDRESSES, ENTITIES, AND LABELS

An address is an on-chain identifier; an entity is an analytical grouping of one or more addresses; a label is a dated external assertion that associates an address or entity with a role or name. These concepts are not interchangeable. Repeated transfers, a common deployer, or a shared event fingerprint can establish a structural relation without proving common ownership. Likewise, an exchange label can support an attribution only under its stated source constraints and within its temporal validity. The benchmark preserves structural relations, external labels, their provenance, and any conflicts separately.

A.4 OBSERVABLE EVIDENCE AND INTERPRETIVE LIMITS

Blocks, transactions, receipts, and logs support claims about recorded execution. Decoding can support claims about transfers or protocol actions when the relevant event and contract semantics are known. Traces and historical state calls add execution or state evidence under their declared collection conditions. Intent, real-world identity, counterfactual harm, and complete ownership are generally not directly observable. The benchmark therefore evaluates claims within the analytical

boundary defined by each rulebook and allows indeterminate or out of scope outcomes when the evidence is insufficient or a claim falls outside that analytical boundary.

A.5 Evidence objects used in this benchmark

The collector obtains blocks with full transaction bodies, receipts, logs, traces, and explicitly permitted historical state responses. Each derived transfer or protocol action retains the block number, transaction hash, and log index or trace path needed to find its source. External owner and exchange labels remain separate dated assertions. These source coordinates let a reference be checked against the downloaded records while preserving the difference between a ledger fact, a protocol action, a structural inference, and an external attribution.

B Additional Related Work

B.1 Financial QA and General Data-Analysis Benchmarks

Financial QA benchmarks differ in their inputs and reference construction. FinQA pairs expert-annotated numerical questions with reasoning programs (Chen et al., 2021), while TAT-QA combines financial tables and text (Zhu et al., 2021). EDINET-Bench evaluates report-level fraud detection, earnings forecasting, and industry classification (Sugiura et al., 2026). FinTMMBench uses expert-designed templates and reasoning guidelines for temporally scoped questions over tables, news, prices, and charts (Zhu et al., 2025). KramaBench evaluates data-to-insight pipelines over heterogeneous data lakes (Lai et al., 2026).

B.2 Reference Construction and Abstention

Reference construction is itself an evaluation choice (Northcutt et al., 2021). SWE-bench uses tests to assess repository-level repairs (Jimenez et al., 2024), and BIRD-INTERACT couples dynamic database interactions with executable test cases (Huo et al., 2026). LiveBench uses objective reference values to reduce dependence on subjective judging (White et al., 2025). FEVER explicitly represents insufficient evidence (Thorne et al., 2018), and AbstentionBench tests when models should decline to answer (Kirichenko et al., 2025). Our design follows this emphasis on checkable outcomes and explicit handling of insufficient evidence. We do not claim that abstention or executable scoring is new; the contribution sought here is their integration with layer-specific evidence requirements for on-chain claims.

B.3 On-chain Measurement and Entity Attribution

On-chain measurement depends on the unit of analysis. Schär et al. (2026) distinguish stablecoin transfer events from the compound transactions in which they occur, showing why event counts need not represent independent payments. Qin et al. (2022) quantify extraction through sandwich attacks, arbitrage, and liquidation using behavior-specific analyses. Wash-trading studies similarly use different observations: Cong et al. (2023) examine statistical regularities on centralized exchanges, whereas Gan et al. (2024) analyze AMM interactions and address relationships. These studies inform our task families, but their findings do not make a rule universal across protocols or establish intent for every detected pattern (Wu et al., 2026; European Securities and Markets Authority, 2025). Our items expose the relevant scope, grouping rule, and evidentiary limits as part of the rulebook.

Entity attribution introduces another evidential boundary. Victor (2020) develop Ethereum address-clustering heuristics, while Ghost Clusters evaluates service attribution against independently obtained ground truth in cryptocurrency tracing (Lubbertsen et al., 2025). These contributions motivate distinguishing structural association from a named identity and measuring coverage separately from correctness (Kappos et al., 2022).

B.4 Graph Representations and Evidence Retrieval

Graphs of Graphs represents transactions within tokens as local graphs and relationships between tokens as a global graph, supporting graph-learning tasks (Luo et al., 2024). Our evidence hierarchy

uses a different organizing principle: abstraction from source observations to actions, flows, and attribution, with derivation links for checking claims. This distinction concerns representation and evaluation targets, not a claim that existing transaction graphs contain no useful evidence.

Graph-based retrieval provides a way to select a manageable context. G-Retriever formulates retrieval over textual graphs using a prize-collecting Steiner tree objective (He et al., 2024), while FinTMMBench's TMMHybridRAG incorporates temporal information into graph and dense retrieval (Zhu et al., 2025). On-chain evidence adds a requirement that semantic relevance and connectivity alone do not settle: a retrieved aggregate must retain the raw records required to check its derivation (Cui & Widom, 2001; Amsterdamer et al., 2011). Our proposed comparison therefore separates raw and hierarchical representations from the retrieval algorithm and accounts for provenance expansion within the context budget. Together with Section 2.2, this positions the study at the intersection of benchmark verification and evidence representation.

C Data Sources, Validation, and Attribution

C.1 Source representations and frozen records

Ethereum execution clients return method-specific objects or arrays inside JSON-RPC response envelopes. The collector calls `eth_getBlockByNumber` with full transactions, `eth_getBlockReceipts`, `eth_getLogs`, and supported trace methods including `trace_block` and `debug_traceBlockByNumber`. An RPC envelope contains `jsonrpc`, `id`, and either `result` or `error`; addresses, hashes, byte strings, and quantities are commonly represented as `0x`-prefixed strings. The project collector extracts each successful `result`, adds the requested block number, and stores one record per line as compressed JSON Lines: {`"block": b, "data": result`}. The local JSONL record is therefore a faithful stored result with a project wrapper, rather than a byte-for-byte copy of the complete RPC envelope.

Dune exposes decoded blockchain datasets as relational tables and displays query outputs as rows and columns. Its result API transports those rows, column metadata, and query metadata in JSON. The project freezes the retrieved rows and schema locally as Parquet tables. Thus, Dune's source representation is a relational query result, while Parquet is the project's immutable storage format. The DEX extracts contain fields such as block number, transaction hash, protocol, pool, token pair, and amounts; address-label extracts contain an address, role or owner, source, and available timestamp when supplied.

The two access paths have different evidential roles. RPC blocks, receipts, logs, traces, and permitted state calls determine ledger-derived references. Labels in the Dune label snapshot can support attribution in T2. Only two multi-initiator T4 items use the Dune label snapshot, and only to exclude labeled service addresses from candidate intermediary paths. Dune MEV query-result tables are external cross-checks: they never determine a benchmark verdict. A cross-check records agreement, disagreement, or non-applicability; disagreement triggers review and does not automatically replace the RPC-derived reference.

Table 7: Source formats, stored representations, and roles in ChainQABench.

Source	Frozen representation	Role in this study
Ethereum JSON-RPC objects and arrays	successful `result` values in a block-number wrapper; gzip JSONL	ledger evidence and reference computation
Dune relational query-result tables	typed rows and schema in Parquet	dated address labels; separate external cross-check tables

C.2 Actual downloaded records and provenance

The examples below contain selected fields from the frozen retrieved records, not invented values. Fields and other members of the same block or table are omitted only for readability. Each description gives the selector needed to identify the record within its source category. RPC excerpts retain the project's JSONL wrapper. Dune examples are actual relational rows rendered as JSON objects for readable display; Parquet itself is not JSON.

RPC block and transaction
Selector: block 25,369,414; first transaction in the block.


```
{
  "block": 25369414,
  "data": {
    "hash": "0xc8
    dee100024b4dffed465946f702232175bdf3c7b50e6ef04f7c7b6070cf158c",
    "parentHash": "0x2
    a051467e479f951738e65dd832917d13f8cd9f80751217d335030cb83e037a5",
    "number": "0x1831b46",
    "timestamp": "0x6a387b0b",
    "transactions": [{
      "hash": "0xb4
    ba2a80de33aa76ca8a7929d762f4f5820b7672c04f7f6bc46c1effbcdf6a79",
      "from": "0x5875db54cd9ae2b2a875e09bb731772297ae9d92",
      "to": "0x0000000aa232009084bd71a5797d089aa4edfad4",
      "transactionIndex": "0x0",
      "value": "0x0"
    }]
  }
}
```


Example C.1: Stored RPC block with selected fields from its first transaction.

RPC log
Selector: block 25,369,414; first stored log (`logIndex=0x0`).


```
{
  "block": 25369414,
  "data": [{
    "address": "0xa0b86991c6218b36c1d19d4a2e9eb0ce3606eb48",
    "topics": [
      "0xddf252ad1be2c89b69c2b068fc378daa952ba7f163c4a11628f55a4df523b3ef",
      "0x0000000000000000000000000000000000004444c5dc75cb358380d2e3de08a90",
      "0x000000000000000000000000000000aa232009084bd71a5797d089aa4edfad4"
    ],
    "data": "0x0
    0000000000000000000000000000000000000000000000000000000000000426c5",
    "transactionHash": "0xb4
    ba2a80de33aa76ca8a7929d762f4f5820b7672c04f7f6bc46c1effbcdf6a79",
    "logIndex": "0x0",
    "removed": false
  }]
}
```


Example C.2: Stored RPC log with the ERC-20 Transfer signature and encoded fields.

The first topic is the recorded ERC-20 `Transfer` signature. Decoding its indexed addresses and data word is a subsequent operation.

Dune DEX cross-check row

Selector: transaction `0xb428ef903d58b545d393026c7501c649 3c41b31dc3e35ce7589884adb87b1ac6`, `evt_index=152`; contract `0xc85aa0aeb2458f9f35f0224f773cd90a96ce813c`.
External cross-check only; not an input to the T4 verdict.


```
{
  "blockchain": "ethereum",
  "project": "uniswap",
  "version": "3",
  "block_number": 25369415,
  "tx_hash": "0xb428
    ef903d58b545d393026c7501c6493c41b31dc3e35ce7589884adb87b1ac6",
  "project_contract_address": "0xc85aa0aeb2458f9f35f0224f773cd90a96ce813c",
  "token_pair": "NAORIS-USDT",
  "tx_index": 25,
  "evt_index": 152,
  "token_sold_amount_raw": "159439915",
  "token_bought_amount_raw": "3928242005682208778514"
}
```


Example C.3: Dune DEX row retained for external cross-checking.

Dune address-label row

Selector: address `0x4ba0650f0d94b36dce68f53f8b08b480a283169d`.


```
{
  "address": "0x4ba0650f0d94b36dce68f53f8b08b480a283169d",
  "blockchain": "ethereum",
  "cex_name": "Coinsbit",
  "distinct_name": "Coinsbit 4",
  "added_date": "2025-06-25"
}
```


Example C.4: Dune address-label row with attribution and date fields.

This row illustrates the name and date fields retained for label provenance.

C.3 Source-integrity validation and exclusion rules

Derived evidence is admitted only after the raw-record consistency checks in Table 8. These checks establish reproducibility under the snapshot; they do not independently validate a real-world attribution or other semantic inference.

Table 8: Raw-record validation applied before constructing derived evidence.

Check	Validation criterion	Treatment on failure
Block continuity	Each retained block references the preceding retained block through its parent hash.	Recollect or quarantine the affected range.
Transaction–receipt agreement	Each collected transaction has one matching receipt in its block.	Mark the block incomplete and exclude receipt-dependent evidence.
Log agreement	Separately collected logs match receipt logs at transaction and log coordinates.	Retain the discrepancy for audit and exclude the unresolved event.
Trace qualification	Each trace is associated with its client, method, block, and transaction.	Qualify or withhold trace-dependent evidence.
Historical-state anchoring	A state response is associated with the requested block after provider, block, and response checks.	Retain ledger-flow evidence without asserting an absolute-state anchor.

Decoder coverage and rulebook coverage are treated separately: a recognized event can remain outside a task's supported domain, whereas an undecoded event cannot be interpreted as evidence that an action did not occur. A derived value is retained only when its source coordinates, decoding rule, unit and sign convention, grouping boundary, and rulebook are sufficient for recomputation. Unknown decodings, unavailable historical state, and conflicting external labels remain explicit rather than being mapped to a negative result.

C.4 Structural attribution signals and threshold selection

Structural relations and contract fingerprints are first computed from RPC evidence within the observation window, before any Dune labels are joined. Labels from the Dune label snapshot that predate the observation window are introduced only afterward where the rulebook explicitly permits label-based naming or propagation. This process produces association, deployment, and contract-family signals for candidate construction. These signals do not by themselves establish common ownership or operational control.

A *sweep feeder* is defined on distinct address-to-address edges formed from successful nonzero-value traces and decoded token transfers. It is an address whose edges within the observation window have exactly one distinct destination. A destination is retained as a sweep hub when it is not classified as a contract account or an EIP-7702 account and has at least 500 such feeders. Only feeders with fewer than 100 observed touches within the observation window are assigned to that hub. The resulting edge is an inferred association: it records that a feeder sends only to one destination in the observation window, rather than proving that the destination controls the feeder.

A *factory* edge is taken from a successful `CREATE` trace. Child addresses are deduplicated before counting, and a deployer is retained when it has at least 100 children within the observation window. This edge establishes deployment but does not establish ownership; proxy factories can deploy contracts operated by unrelated users. If one address receives both sweep and factory attributions, the factory edge takes precedence because the deployment event is directly recorded on-chain.

A contract *fingerprint* is the set of distinct event signatures (`topics[0]`) emitted by an address. Exact signature-set clusters are used first, with the rarest observed signature as a fallback. An owner name is propagated only when the labeled members of the selected cluster contain exactly one distinct owner. The signal therefore supports a contract family or type; it is not direct proof of common operation.

Table 9: Structural attribution signals and validation on the snapshot. Temporal recurrence measures stability rather than identity accuracy.

Signal	Inclusion or propagation rule	Output	Validation and interpretation
Sweep	At least 500 one-destination feeders; attribute only feeders with fewer than 100 touches	245 hubs	94.0% second-half recurrence; median scale ratio 0.906; inferred association
Factory	At least 100 distinct children from successful `CREATE` traces	110 deployers	90.9% second-half recurrence; median scale ratio 0.948; deployment only
Unique union	Sweep $\cup$ factory; count two central-address overlaps once	353 unique hubs/deployers	67 named and 286 unnamed
Fingerprint	Exact event-signature set, then rare-signature fallback; propagate only from one known owner	9,617 addresses	98.81% leave-one-out precision among members with labels predating the observation window (7,891/7,986); family/type signal

The broader attribution analysis contains 1,495,466 addresses, or 21.78% of the observed address universe, while retaining 88.61% of address touches and 99.15% of log emissions. Sweep and factory rules attribute 1,322,024 addresses to the 353 structural centers in Table 9. Across direct la-

bels, structural attribution, and fingerprint-derived categories, 1,336,124 addresses are non-unknown (89.34% of the analyzed address set), including 12,161 directly labeled addresses (0.81%). These are coverage statistics and should not be interpreted as identity accuracy.

The sweep and factory thresholds are selected without labels. We split the observation window at block 25,394,505 and reconstruct each relation independently in the first and second halves. For a candidate full-window threshold T, the first-half threshold is $T/2$. We measure whether a first-half entity appears in the second half and the median ratio between its second- and first-half sizes. Scanning thresholds from large to small, we retain the upper threshold before a lower candidate reduces second-half recurrence by at least five percentage points and the median scale ratio by at least 0.05. Sweep candidates must additionally have a mean feeder-EOA share of at least 0.70.

For sweep hubs, lowering the threshold from 500 to 200 changes second-half recurrence from 94.0% to 81.3% and the median scale ratio from 0.906 to 0.755; the feeder-EOA share at 500 is 0.808. For factories, lowering the threshold from 100 to 50 changes recurrence from 90.9% to 79.9% and the median scale ratio from 0.948 to 0.893. These discontinuities select 500 and 100, respectively. The 100-touch feeder cap is a fixed activity filter rather than an optimized breakpoint: varying it from 10 touches to no cap changes the number of attributed addresses by approximately 12%, so the analysis records the choice explicitly rather than treating it as empirically identified. This cap is distinct from the 100-touch lower bound used to define the broader analyzed address set.

Only fingerprint propagation has a directly measured precision estimate. Sweep recurrence and factory recurrence measure temporal stability, not identity accuracy. Accordingly, the rulebooks treat sweep, deployment, and fingerprint evidence according to their different analytical boundaries.

D TASK RULEBOOKS

Each rulebook below is written for the author who constructs the benchmark items of one task. It specifies what must be fixed before an item is built, which evidence may select a case and which may determine its reference, how the reference is derived, graded, and independently verified, the numerical thresholds and adjudication order, and the boundary cases that items should probe. Fields that do not affect task semantics or reproduction are omitted. The system prompt given to the answering LLM for each task is reproduced in Appendix H; each rulebook heading points to the corresponding prompt.

An item is either computational, requesting typed derived values, or adjudicative, requesting typed values together with a verdict and an evidence grade (Appendix F; verdict meanings in Appendix F.3). The grades are `anchor_gold` (external anchor or direct confirmation), `flow_gold` (ledger alone without an absolute-balance anchor; T1 only), `silver` (conditions pass, but some evidence is estimated or heuristic), and `contested` (evidence conflicts or does not close); an out-of-scope item carries `out_of_scope` for both.

D.1 T1 Rulebook: Asset Flows

Prompt: p. 41

T1.1 Scope

T1 measures the holdings change of ERC-20 asset X for a fixed address set U over an inclusive Ethereum block interval $[b_0, b_1]$ within the observation window (blocks 25,369,414–25,419,597; 2026-06-22–2026-06-28). Holdings combine ERC-20 balances $\sum \texttt{balanceOf}(X, a)$ for $a \in U$ and Uniswap V4 PoolManager ERC-6909 claim balances $\sum \texttt{balanceOf}(a, \texttt{id})$, where $\texttt{id} = \texttt{uint256}(\texttt{uint160}(X))$. The PoolManager is `0x000000000004444c5dc75cb358380d2e3de08a90`.

Native ETH, ERC-721, ERC-1155, and periods outside the observation window are outside scope. T1 does not infer investment intent, buying motivation, or price impact.

Before the target window is inspected, the item fixes U, X, and $[b_0, b_1]$; the token behavior (`standard_erc20`, `weth`, `nonstandard`, or `not_erc20`); the permitted archive calls; the claim wording; and the requested output fields. A set whose membership depends on transfers observed within $[b_0, b_1]$ is circular and yields `indeterminate/contested`.

T1.2 Admissible evidence

Admissible evidence consists of ERC-20 `Transfer(address,address,uint256)` logs with exactly three topics; canonical WETH `Deposit(address,uint256)` and `Withdrawal(address,uint256)` logs; PoolManager ERC-6909 `Transfer(address,address,address,uint256,uint256)` logs; mint, burn, pause, proxy-upgrade, and ownership-transfer events; and successful transaction and trace metadata needed to delimit the interval.

Token amounts are kept as raw integers; `decimals` serves only for display. Candidate graphs, labels, model outputs, and summary statistics may select cases but cannot determine the reference.

When archive access is permitted, ERC-20 `balanceOf(a)` and PoolManager `balanceOf(a,id)` (selector `0x00fdd58e`) are queried at $b_0 - 1$ and b_1. A value is accepted only when two independent providers agree. An item whose balance anchor lacks this agreement is not built; runtime code on which the providers disagree makes `privilege_capable` `unknown`. Runtime code is obtained from the successful creation trace for a token deployed during the observation window, and from archive `eth_getCode` at b_1 for an older token.

Static privilege capability is determined by the frozen selectors in Table T1-1.

Table T1-1. Selectors for static privilege capability

Capability	Selector
`mint`	`0x40c10f19`
`burn` variants	`0x9dc29fac`, `0x42966c68`
`pause`, `unpause`	`0x8456cb59`, `0x3f4ba83a`
`upgradeTo`, `upgradeToAndCall`	`0x3659cfe6`, `0x4f1ef286`
`transferOwnership`	`0xf2fde38b`

Runtime bytecode is scanned as opcodes, and only PUSH4 operands are collected. The flag `privilege_capable` takes the values in Table T1-2. Selector presence indicates capability, not access-control semantics.

Table T1-2. Values of `privilege_capable`

Value	Condition
`true`	A frozen selector is present, or a `Paused`, `Unpaused`, `Upgraded`, or `OwnershipTransferred` event appears in the observation window
`false`	The available code contains neither signal
`unknown`	Otherwise

T1.3 Deterministic computation

For qualifying ERC-20 transfers,

$$\begin{aligned} \texttt{inflow} &= \sum \texttt{value} \quad \text{where } \texttt{to} \in U \text{ and } \texttt{from} \notin U, \\ \texttt{outflow} &= \sum \texttt{value} \quad \text{where } \texttt{from} \in U \text{ and } \texttt{to} \notin U, \\ \texttt{erc20_delta_raw} &= \texttt{inflow} - \texttt{outflow}. \end{aligned}$$

Transfers within U contribute zero. A zero sender denotes minting and a zero recipient denotes burning. For canonical WETH only,

$$\begin{aligned} \texttt{erc20_delta_raw} = {} & \text{transfer inflow} - \text{transfer outflow} \\ & + \sum \texttt{Deposit.wad} \quad \text{where } \texttt{dst} \in U \\ & - \sum \texttt{Withdrawal.wad} \quad \text{where } \texttt{src} \in U. \end{aligned}$$

In a PoolManager ERC-6909 transfer, `topics[1:4]` encode `from`, `to`, and currency `id`, and the data encodes `caller` and `amount`. Only transfers with $\texttt{id} = \texttt{uint256}(\texttt{uint160}(X))$ are kept, and the same external-flow rule applies:

$$\begin{aligned} \texttt{erc6909_delta_raw} &= \text{claim inflow} - \text{claim outflow}, \\ \texttt{flow_delta_raw} &= \texttt{erc20_delta_raw} + \texttt{erc6909_delta_raw}. \end{aligned}$$

Both legs are required because PoolManager deposits and withdrawals can change the representation of the same holdings. With archive anchors,

$$\texttt{balance_delta_raw} = [\text{ERC20 balance at } b_1 - \text{ERC20 balance at } b_0 - 1] + [\text{ERC6909 claim balance at } b_1 - \text{claim balance at } b_0 - 1].$$

The claim is compared with `balance_delta_raw` when available and with `flow_delta_raw` otherwise. A ledger/archive disagreement does not contradict a claim matching the accepted archive anchor.

The field `change_cause` takes the values in Table T1-3.

Table T1-3. Values of `change_cause`

Value	Condition
`mint`	A three-topic transfer minting into U
`burn`	A burn from U
`mint_and_burn`	Both a mint into U and a burn from U
`transfer_only`	Otherwise

T1.4 Adjudication order

Table T1-4 lists the rules in order; the first matching rule applies.

Table T1-4. Adjudication order for T1

Order	Condition	Verdict
1	Unsupported assets, native ETH (`id=0`), a token that self-destructed within the window, periods outside the observation window, or requests for an absolute balance level rather than a change	`out_of_scope/out_of_scope`
2	A circular address set, a relevant proxy upgrade, a pause or ownership event affecting U, nonstandard behavior without an anchor, or unavailable or conflicting required evidence	`indeterminate/contested`
3	A matching claim	`supported`
	A direct archive mismatch	`contradicted/anchor_gold`
	A ledger-only mismatch	`contradicted/contested`
4	A supported two-provider archive result	Grade: `anchor_gold`
	Deterministic ledger closure	Grade: `flow_gold`
	Detected or unknown privilege capability, or anchored nonstandard behavior	Grade of a supported result capped at `silver`

A pause event affects U when a pause-state event occurs inside the interval while U has a positive balance; without an archive balance, positive cumulative in-window flow through that block or a later transfer involving U is used. In addition, a claim restricted to ERC-20 change is contradicted when ERC-6909 claims moved under the combined-holdings definition. Mint or burn does not invalidate a purely numeric claim but makes a causal claim of buying or accumulation indeterminate. For an interval partially overlapping the observation window, the reference still records the reconstructable flow on the intersection even when the verdict is out of scope.

T1.5 Output and boundary cases

The reference records the item identifiers, interval, and token behavior; inflow, outflow, `erc20_delta_raw`, `erc6909_delta_raw`, and `flow_delta_raw`; optionally `balance_delta_raw`; and cause and privilege annotations. Raw integers have zero tolerance, and a display conversion may use only the item-specific decimal tolerance. A verdict item always requests `flow_delta_raw`, `token_behavior`, `change_cause`, `privilege_capable`, the verdict, the grade, and `reason_code`; an archive-enabled

item also requests `balance_delta_raw`. Table T1-5 lists the required fields and the `reason_code` values, which name the rule of Table T1-4 that applied.

Table T1-5. Required fields and `reason_code` values for T1

Field	Meaning
`inflow_raw`, `outflow_raw`, `internal_raw`	Inflow sum, outflow sum, and sum of moves only within U
`transfer_delta_raw`	Net counted from `Transfer` alone (before the WETH correction)
`erc20_delta_raw`, `erc6909_delta_raw`	ERC-20 portion (with the WETH correction) and PoolManager ERC-6909 claim portion
`flow_delta_raw`	Ledger net, ERC-20 portion plus claim portion
`balance_before_raw`, `balance_after_raw`, `balance_delta_raw`	Holdings at $b_0 - 1$ and at b_1, and their difference
Rule	**`reason_code`**
1	`window_outside_w26`, `absolute_balance`, `unsupported_asset`, `selfdestruct_token`
2	`address_set_not_fixed`, `position_frozen`, `mint_burn_on_u`, `proxy_upgrade_in_window`, `owner_change_on_u`, `nonstandard_no_anchor`
3	`claim_vs_anchor`, `claim_vs_ledger_only`
4	`capability_detected`, `capability_unknown`, `nonstandard_cap`, `balance_not_ledger`, `anchor_match`, `ledger_only`

Table T1-6 lists the boundary cases that items should probe, with the treatment the reference applies.

Table T1-6. Boundary cases for T1

Case to probe	Reference treatment
A self-transfer or a transfer between members of U	Contributes zero
A four-topic ERC-721 `Transfer`	Carries a token ID, not an ERC-20 amount
A PoolManager deposit plus claim minting	Must not be double-counted as a loss
Fee-on-transfer or rebasing behavior without an anchor	Unresolved
An interval containing a proxy upgrade	Must be split or adjudicated as indeterminate

T1.6 Verification

Item inputs must be frozen before answer computation, and all fields are recomputed from raw logs and permitted calls. An independent verifier must check topic cardinality, interval inclusivity, WETH adjustments, V4 currency IDs, zero-address semantics, archive boundaries, and every requested typed field. A mismatch stays under review; an independent verifier match confirms implementation consistency.

Rulebook D.1: T1: Asset Flows.

D.2 T2 Rulebook: Exchange Attribution

Prompt: p. 44

T2.1 Scope

Each T2 item adjudicates one claim: (1) two addresses share a controller; (2) an address or fixed set is controlled by a centralized exchange; or (3) it is controlled by named exchange E. These claims are distinct: common control does not imply exchange control, and exchange control does not identify an exchange. A missing connection does not prove separate control.

Personal identity, internal customer accounts, complete wallet inventories, and control outside the observation window are outside scope. Before construction, the item fixes the claim type and wording; the target addresses and the claimed role or name; the target interval; the frozen ledger and Dune label snapshots; the flow definitions and thresholds; and the required fields.

T2.2 Admissible evidence

Ledger evidence may establish structural relations and roles through successful top-level ETH transfers, relevant ERC-20 transfers, deployer relations, access-control evidence, and deterministic counterparty and forwarding counts. For flow features, an ETH transfer is a top-level transaction with positive value, a successful receipt, and a non-null recipient; internal traces are excluded unless explicitly requested.

The sole label source is the frozen Dune label snapshot for the observation window, which records owner/exchange membership, owner key and category, exchange name, label timestamp, account type, and contract name. A copied or propagated label stays dependent on its seed and is not another evidence family. Table T2-1 gives the evidence use of a label by its timing.

Table T2-1. Evidence use by label timing

Label timing	Evidence use
Established before `2026-06-22 00:00:00 UTC`	PROVEN; may support or oppose a claim
Established during or after the observation window	UNPROVEN; may select candidates only
No label	Never negative evidence

Direct confirmation requires an official operator disclosure, a verifiable signature or record, a controlled on-chain deposit/withdrawal test, or official investigative or seizure material. None is present in the frozen benchmark.

T2.3 Evidence families and roles

Table T2-2 defines the role signals and evidence families.

Table T2-2. Role signals and evidence families

Signal	Definition
Sweep-type sender	Sent at least three ETH transfers to the target and nowhere else in the observation window
Sweep signal	At least 10 sweep-type senders
Ledger role R	`inflow_counterparties` at least 1,000 and `outflow_counterparties` at least 100 for distinct ETH counterparties within the observation window
Flow family F	Sweep signal or R; both form one family because they use the same ETH-transfer graph
Label family L	A PROVEN Dune label supplies an exchange role; multiple Dune rows or fields count as one family

An address has an exchange role when it is on the Dune exchange list and its category is not an other-service category, or when its `owner_category` is exactly `Centralized Exchange`.

An on-list address whose category exactly matches an other-service category has an internal conflict.

Table T2-3 lists the frozen other-service vocabulary for off-list addresses.

Table T2-3. Other-service vocabulary for off-list addresses

Category		
`Decentralized Exchange`	`Staking`	`Gaming`
`Lending`	`Token Issuance`	`Casino`
`Oracle`	`Governance`	`Privacy Services`
`Yield Vaults`	`Creators and brands`	`Asset Management`
`Derivatives`	`Payment Processing`	`Perpetuals`
`Bridge`	`Insurance`	`MEV`

Matching is exact; `Centralized Exchange` is not a substring match for `Decentralized Exchange`. An unlisted category is role-unknown.

Same-controller evidence counts four relation types, not occurrences: direct ETH transfer, common funding source, same deployer, and shared access-control evidence. Repeated instances of one type count once. Frequent transfers to an exchange do not establish that the sender is exchange-controlled.

T2.4 Adjudication

Two general evidence rules apply. With no PROVEN support, a PROVEN label giving a different role or name yields `contradicted/contested`; and coexisting PROVEN support and opposition yields `indeterminate/contested`. Table T2-4 gives the claim-specific verdicts.

Table T2-4. Verdicts by claim type

Claim type	Condition	Verdict
Exchange control	L and F hold, R holds, and no other-service role or internal conflict exists	`supported/silver`
	No L support and a PROVEN label supplies another service role	`contradicted/contested`
	At most one family, failed R, absent or UNPROVEN labels, or internal conflict	`indeterminate/contested`
Named exchange	At least two independent PROVEN label families identify the exact name and role without conflict	`supported/silver`
	No PROVEN support for E and a PROVEN label identifies another exact name	`contradicted/contested`
	One supporting family, mixed support and opposition, or unconfirmed timing	`indeterminate/contested`
Same controller	Directly confirmed root and at least two distinct relation types	`supported/silver`
	Direct confirmation of different controllers in the same period	`contradicted/anchor_gold`
	Otherwise	`indeterminate/contested`

Names are compared exactly after case normalization. With a single label family and no directly confirmed root, no item can carry a `supported` reference for a named-exchange or same-controller claim; missing links are not negative evidence.

An address-set claim is `contradicted/contested` if any member is contradicted, `supported/silver` if all members are supported, and `indeterminate/contested` otherwise. Negative claims use the same evidence rules. Claims about personal identity, internal customer accounts, wallet completeness, or future or out-of-window control (answer types

personal_identity, internal_customer_account, wallet_completeness, future_control; none when in scope) are out_of_scope/out_of_scope. A compound claim is adjudicated on its in-scope component first: a contradiction stays contradicted/contested, and otherwise the compound claim is out of scope.

T2.5 Output and boundary cases

The reference records the claim type, target addresses and interval, and claimed role or name; exact flow counts and relation types; label timing and conflict status; evidence-family indicators; and the requested verdict, evidence grade, and uncertainty reason. In the reference, counts are exact, addresses are lower-case, names and categories reproduce the frozen table exactly, and dates are written as YYYY-MM-DD. Table T2-5 lists the boundary cases that items should probe, with the treatment the reference applies.

Table T2-5. Boundary cases for T2

Case to probe	Reference treatment
A user sending to an exchange	Not thereby exchange-controlled
Exchange-to-exchange transfer	Does not establish common ownership
Bridges, processors, custody services, casinos, DEX routers, treasuries, and market makers	Can exhibit exchange-like flow
A combined service label	Does not permit ledger flow to choose one controller name
Prefix-related exchange names whose frozen owner keys differ	Kept distinct
A disagreement between category and list	Left unresolved

T2.6 Verification

The claim, targets, interval, and snapshots must be frozen before computation, and candidate generation must be kept separate from adjudication. An independent implementation must recompute top-level ETH flow counts, apply the label time rule, count evidence families by underlying method, enforce exact name and category matching, and verify address-set and compound-claim aggregation. An independent verifier match confirms consistency but cannot create direct confirmation.

Rulebook D.2: T2: Exchange Attribution.

D.3 T3 Rulebook: MEV-Related Interactions

Prompt: p. 47

T3.1 Common scope and evidence

T3 separately adjudicates sandwich pairs, arbitrage transactions, and liquidation transactions within the observation window (Ethereum blocks 25,369,414–25,419,597). It does not assign a generic MEV label or infer intent, legal status, counterfactual victim loss, or liquidation discount without a valuation surface.

Admissible evidence comprises archive-node blocks and transactions, receipts and logs, and execution traces. ERC-20 amounts come only from three-topic Transfer(address, address,uint256) logs; four-topic ERC-721 transfers are excluded. Native ETH moves only through successful positive-value call traces. Logs are ordered by (block_number,transaction_index,log_index), and raw integer token amounts are retained. Dune-derived tables may select or cross-check cases but do not replace RPC-derived computation.

T3.2 Swap decoding

Table T3-1 lists the decoded swap events and their pool identity.

Table T3-1. Decoded swap events

Venue	Event signature	Pool identity
Uniswap V2	`Swap(address,uint256,uint256, uint256,uint256,address)`	Log address
Uniswap V3	`Swap(address,address,int256,int256, uint160,uint128,int24)`	Log address
Uniswap V4	`Swap(bytes32,address,int128,int128, uint160,uint128,int24,uint24)`	Log address plus pool ID
Curve	`TokenExchange(address,int128, uint256,int128,uint256)`	Log address
Curve	`TokenExchangeUnderlying(address, int128,uint256,int128,uint256)`	Log address
Curve	`TokenExchange(address,uint256, uint256,uint256,uint256)`	Log address
Balancer	`Swap(bytes32,address,address, uint256,uint256)`	Log address plus pool ID

Uniswap V2 deltas are `amountIn` − `amountOut`; V3 uses the first two signed `int256` data words; and V4 user-side signed `int128` amounts are negated to obtain pool-side deltas. Direction is the sign of the pool-side token0 delta. Execution price is trader tokens received divided by tokens given, in base units, and only same-direction swaps are compared.

T3.3 Uniswap V4 gate

For PoolManager `0x000000000004444c5dc75cb358380d2e3de08a90`, the gate is applied in order.

1. The PoolKey is recovered from `Initialize(bytes32,address,address, uint24,int24,address,uint160,int24)` and its pool ID verified.
2. ERC-20, successful ETH-call, and ERC-6909 settlement are included.
3. For ERC-6909 `Transfer(address,address,address,uint256, uint256)`, the indexed `from`, `to`, and currency `id` are read, then `caller` and `amount` from the data.
4. The first applicable status is assigned: unknown PoolKey (`pool_unknown`); swap-return-delta hook, `0x08` or `0x04` (`hook_return_delta`); liquidity modification or donation (`v4_liquidity_or_donate`); settlement that does not close (`not_closing`); or `closes`. When requested, it is recorded as `v4_status`; the field is not requested for transactions without V4 swaps.

Only `closes` is adjudicable. Every other status, including a missing required PoolKey or settlement surface, is `out_of_scope/out_of_scope`. Absence of an ERC-20 transfer alone is insufficient when ETH and ERC-6909 evidence closes settlement.

T3.4 Sandwich

T3.4.1 Target and checks

Every ordered same-pool swap pair with a block difference of 0–5 inclusive, including same-transaction pairs, is enumerated, and candidates are counted before verdict filters.

Front and back legs must have opposite pool-side directions, and a third-party swap between them must share the front leg's direction. Its minimum execution price must be below `front_price`$(1-\tau)$, with adopted $\tau = 0.005$; sensitivity outputs may additionally use 0.0005, 0.001, and 0.002. Equal transaction senders provide the strongest attribution; a shared destination or final recipient is weaker, and a shared router designated by the item is not attribution evidence; same-transaction forwarding is followed to the final recipient. Same-block execution provides the strongest ordering evidence. A liquidation event in either outer transaction disqualifies the claim.

T3.4.2 Adjudication

Table T3-2 gives the sandwich verdicts.

Table T3-2. Sandwich verdicts

Condition	Verdict
Reciprocal legs, third-party victim, worse price at $\tau = 0.005$, same sender, same block, no liquidation	`supported/anchor_gold`
Same structure with destination or final-recipient attribution or a cross-block span of at most five	`supported/silver`
Nonreciprocal pair, span over five blocks, no third-party victim, or liquidation event	`contradicted/anchor_gold`
Unresolved decoding or attribution, or unresolved worse-price failure	`indeterminate/contested`

Different senders alone do not prove different controllers. A positive round trip is reported but is not a verdict condition. Same-pool liquidity added before and removed after the victim neither disqualifies nor promotes the claim by itself.

T3.5 Arbitrage

T3.5.1 Target and computation

The target population is transactions with at least two decoded swaps, and the beneficiaries are `transaction.from` and `transaction.to`. For each asset, receipts are added and payments subtracted across three-topic ERC-20 transfers, canonical WETH deposit/withdrawal events, successful ETH calls, and PoolManager ERC-6909 claims. ETH and WETH are merged. ETH paid by a beneficiary to the block miner is reported but excluded from the beneficiary's net change, and therefore from cycle closure and gain.

Cycle closure uses the full per-asset beneficiary net change; a transaction with both a decreased and an increased asset is routing, not a closed cycle.

Gain compared with gas is measured more narrowly. It includes only ETH/WETH exchanged between beneficiaries and decoded-swap contracts, excludes flows associated with `Mint`, `Burn`, `Collect`, or `ModifyLiquidity`, and converts other increased assets only with a rate from a decoded swap in the same transaction. Gas is computed as `gas_used` × `effective_gas_price`. External prices and oracles are inadmissible.

An exchange leg whose endpoints emit no decoded swap is undecoded and takes precedence over the pool-count, cycle, and gas tests. Lending and repayment of the same token is a flash-loan leg, not an exchange leg.

T3.5.2 Adjudication order

Table T3-3 lists the rules in order; the first matching rule determines the verdict.

Table T3-3. Adjudication order for arbitrage

Order	Condition	Verdict
1	No decoded swap or a non-closing V4 state	`out_of_scope/out_of_scope`
2	Failed transaction	`contradicted/anchor_gold`
3	Undecoded exchange leg	`indeterminate/contested`
4	Single-pool exchange, or both a decreased and an increased asset	`contradicted/anchor_gold`
5	Successful transaction with no decreased asset, at least three pools, no undecoded leg, and admissible gain above gas	`supported/anchor_gold`
6	Closed successful two-pool cycle, or an increased asset lacking an in-transaction conversion rate	`supported/silver`
7	Otherwise	`indeterminate/contested`

T3.6 Liquidation

T3.6.1 Target and checks

The frozen event surface consists of the event signatures in Table T3-4.

Table T3-4. Liquidation event signatures

Event signature
`LiquidationCall(address,address,address,uint256,uint256,address,bool)`
`LiquidationCall(address,address,address,uint256,uint256,address)`
`LiquidateBorrow(address,address,uint256,address,uint256)`
`AbsorbDebt(address,address,uint256,uint256)`
`Liquidate(address,address,address,uint256,uint256,uint256,uint256)`
`Bite(bytes32,address,uint256,uint256,uint256,address,uint256)`
`Bark(bytes32,address,uint256,uint256,uint256,address,uint256)`
`Liquidate(bytes32,address,address,uint256,uint256,uint256,uint256,uint256)`

The call selectors are `0x00a718a9`, `0xf5e3c462`, `0x6f70e66f`, `0xc3cecfd2`, and `0xd8eabcb8`. Detection alone is not completion evidence and may include failed outer or child calls.

A positive claim requires an explicit frozen liquidation event; successful outer and actual liquidation calls; different borrower and liquidator; debt-asset outflow from the liquidator; collateral-asset inflow to the liquidator; and transaction-level closure. The event's liquidator field controls the role and need not equal the outer sender. A selector-only reverted call, or one without event and flow, is a direct negative.

T3.6.2 Adjudication

Table T3-5 gives the liquidation verdicts. The item states whether the emitting contract is on the frozen cross-check list.

Table T3-5. Liquidation verdicts

Condition	Verdict
All evidence closes and the emitter appears on the frozen cross-check list	`supported/anchor_gold`
All evidence closes and the emitter is absent from that list	`supported/silver`
Failed outer/actual call, self-liquidation, or selector-only unclosed flow	`contradicted/anchor_gold`
Incomplete flow or asset identity or a collateral-discount claim	`indeterminate/contested`

T3.7 Common outputs and verification

Non-adjudicable V4 states, transactions with no decoded swap, cross-chain activity, and activity outside the observation window are `out_of_scope/out_of_scope`. A partially decoded swap transaction with an unresolved exchange leg is `indeterminate/contested`. Priority-fee rank, direct miner payment, and prior actor activity are circumstantial; they do not determine the MEV type.

The reference contains only declared typed fields. Integers are exact, and execution price uses an absolute tolerance of 10^{-6}.

An independent implementation must reconstruct every required field from registered evidence and match the reference. It must verify pool identity, ordering, signed amounts, V4 closure, attri-

bution, asset flows, gas, and verdict precedence. Human-readable explanations do not substitute for typed fields.

Rulebook D.3: T3: MEV-Related Interactions.

D.4 T4 Rulebook: AMM Wash-Trading Patterns

Prompt: p. 52

T4.1 Scope

T4 tests whether a fixed group of swaps in one AMM pool satisfies ledger-observable wash-trading criteria within the observation window (Ethereum blocks 25,369,414–25,419,597). It does not adjudicate legal wash trading or manipulative intent. Items either adjudicate a supplied structural episode or compute pool/window distribution statistics; distribution screening never establishes a structural verdict.

The included venues are verified Uniswap V2 pairs, Uniswap V3-signature emitters, and Uniswap V4 pools whose hooks cannot alter settlement amounts. V3 inclusion is signature-based and does not independently verify every emitter against the official factory. Other venues and unsupported token behavior are outside scope.

For a verdict item, the supplied `(transaction_hash, log_index)` list is the group; surrounding swaps are neither added nor removed. Before computation, the pool, venue, and asset pair; ordered swap IDs; transaction senders and destinations; pool-side deltas; same-transaction transfers and swaps; successful traces; V4 hook metadata; the ETH relation graph; the Dune service-label snapshot; and the requested output fields must be frozen.

Candidate origin and scores do not determine the answer. Only for whole-pool counts may maximal consecutive runs with the same transaction sender generate candidates, and the item must disclose that sender equality is guaranteed by construction.

T4.2 Swap evidence and scope gate

Table T4-1 lists the decoded swap events, pool identity, and pool-side deltas.

Table T4-1. Decoded swap events and pool-side deltas

Venue	Swap signature	Pool identity and delta
Uniswap V2	`Swap(address,uint256,uint256, uint256,uint256,address)`	Log address; `amountIn − amountOut`
Uniswap V3	`Swap(address,address,int256, int256,uint160,uint128, int24)`	Log address; signed data words 0 and 1
Uniswap V4	`Swap(bytes32,address,int128, int128,uint160,uint128, int24,uint24)`	PoolManager plus pool ID; negate user-side signed words 0 and 1

The Swap-event sender may be a router; `transaction.from` is used as the initiator. ERC-20 amounts come from three-topic transfer logs, and native ETH comes from successful `call` traces, including the top-level call.

An item is `out_of_scope/out_of_scope` when the venue is unsupported; a V2-signature emitter has no `Sync` evidence during the observation window; the group has fewer than 2 or more than 10 swaps; token behavior is fee-on-transfer, rebasing, backdoored, or unreconstructable; or a V4 hook can alter settlement. A 2–10-swap nonadjacent group stays in scope and is structurally contradicted.

For V4, PoolManager `0x000000000004444c5dc75cb358380d2e3de08a90` is checked in two steps.

1. The PoolKey is recovered from `Initialize(bytes32,address,address, uint24,int24,address,uint160,int24)` and its pool ID verified.

2. The hook is classified. A zero hook, or a hook with neither `beforeSwapReturnDelta (1<<3)` nor `afterSwapReturnDelta (1<<2)`, is in scope; either bit places the pool out of scope; and an unavailable hook or a pool-ID mismatch is `indeterminate/contested`.

The dynamic-fee flag `0x800000` alone does not affect scope.

T4.3 Adjudication order

Adjudication starts with the scope gate of T4.2 and continues through the gates below in the order given, stopping at the first matching step.

T4.3.1 Structure

The group is evaluated in pool order.

1. No outside same-pool swap may lie between the group's first and last swap.
2. At least one asset must have both positive and negative pool-side deltas.
3. $\texttt{last_block} - \texttt{first_block} \leq 200$, inclusively.
4. For each asset, the ratio is computed as

$$\texttt{ratio} = \left|\sum \delta\right| \Big/ \sum |\delta|.$$

   Closure passes when $\min(\texttt{ratio_token0}, \texttt{ratio_token1}) \leq 0.05$, compared as an exact fraction. The item records each asset's ratio and whether both assets pass (`closure_both_assets_pass`); passing on both assets is not required.

Any directly observed structural failure yields `contradicted/anchor_gold`.

T4.3.2 MEV exclusion

Any confirmed exclusion yields `contradicted/anchor_gold`.

Sandwich. The exclusion requires two group transactions in the same block with the same sender or destination and opposite directions, and, between them in pool order, a same-direction swap from another initiator, which may lie outside the supplied group. If X is the asset removed by the first leg, it also requires

$$0.9 \;\leq\; \frac{X \text{ returned by second leg}}{X \text{ removed by first leg}} \;\leq\; 1.1.$$

Cyclic arbitrage. A group transaction is cyclic arbitrage when it swaps in at least two pools and the actor set $\{\texttt{transaction.from}, \texttt{transaction.to}\}$ has no negative asset net flow and at least one positive flow. Transfers within the actor set are excluded; three-topic ERC-20 transfers, PoolManager ERC-6909 claims mapped by currency ID, and successful native-ETH calls are summed. WETH `Deposit`/`Withdrawal` events are not included in this T4 exclusion calculation.

Rebase-style loop. Within one transaction, the same pool receives asset X in one swap and sends the same amount of X in another. No additional price condition applies.

T4.3.3 Initiator relation

A single initiator passes. For multiple initiators, an undirected graph is constructed from successful, positive-value, top-level ETH transfers observed no later than the group's final block; every Dune-labeled address and every unlabeled hub with at least 100 distinct ETH counterparties within the observation window are excluded as intermediate nodes. A direct connection or one permitted intermediate passes. Connection only through a service or hub, or no connection within two hops, yields `indeterminate/contested`. A missing connection does not prove separate control.

T4.3.4 Benign explanation

Once the prior gates pass, any of these yields `indeterminate/contested`: a multi-pool route that is not cyclic arbitrage; initiator LP activity in the same pool within $[\texttt{first_block} - 200,\ \texttt{last_block} + 200]$, including V2/V3 `Mint` or `Burn` and V4 `ModifyLiquidity`; or mixed token0 signs within one transaction that do not satisfy the rebase rule. Active-bot status, frequent trading, market-maker inventory, treasury movement, public-contract identity, and external MEV labels are not deterministic verdict evidence.

T4.3.5 All gates passed

When every gate passes, the verdict is `supported/anchor_gold` for one initiator and `supported/silver` for multiple initiators connected through the permitted ETH graph.

T4.4 Distribution screening

For a fixed Uniswap V2 pool and window, pools with at least 300 swaps are screened. One incoming-asset amount is used per swap (`amount0In` when nonzero, otherwise `amount1In`), in absolute value. Tests are not run with fewer than 100 amounts. Table T4-2 defines the three tests.

Table T4-2. Distribution screening tests

Test	Rule
Benford	Chi-square with 8 degrees of freedom against $P(d) = \log_{10}(1 + 1/d)$; fail above 15.507
Roundness	Share p of amounts that are multiples of $10^{\max(d-3,1)}$, where d is the number of digits; expected share 0.10; $z = (p - 0.10)/\sqrt{0.10 \cdot 0.90/n}$; fail when $z \leq 1.96$, that is, when amounts do not cluster at round values
Tail	Sort nonzero amounts descending; $k = \max(\lfloor 0.1\, n \rfloor, 20)$; $x_{\min}$ is the k-th largest; $\alpha = k/\sum \log(x/x_{\min})$ over the top k; unavailable with fewer than 50 nonzero amounts; fail outside $1 < \alpha < 2$

At least two failures mark a pool suspicious for screening. Statistics alone always yield `indeterminate/contested` because stablecoin and arbitrage flows can contaminate trade-size distributions.

T4.5 Output and boundary cases

For structural items, the reference records pool/group identity and ordered swap IDs, initiators and block span, signed asset deltas and closure ratios, V4 hook status, MEV exclusions, initiator relation and benign alternatives, and the requested verdict and evidence grade. For screening items, it records the pool/window, all three statistics and failure indicators, and contamination status. Counts and raw amounts have zero tolerance; ratios use item-declared precision. Table T4-3 lists the boundary cases that items should probe, with the treatment the reference applies.

Table T4-3. Boundary cases for T4

Case to probe	Reference treatment
Exactly 10 swaps are in scope; 11 are not.	
A span of exactly 200 blocks passes; 201 fails.	
Closure	Uses the minimum ratio, so only one asset must satisfy 0.05
Passing numeric tests	Does not override the V4 scope gate
ERC-6909-only settlement	Can establish a cyclic-arbitrage exclusion
A router path	Can meet direction and closure while still being a benign alternative
A service address or high-degree hub	Cannot manufacture a two-hop relation

T4.6 Verification

The exact group must be frozen before computation. An independent implementation must reconstruct pool order, deltas, ratios, bounds, V4 PoolKeys and permission bits, exclusion signals, relation paths, benign explanations, and all typed verdict fields from raw logs, traces, and the fixed Dune service-label snapshot. Gates must be applied in the stated order, and screening kept separate from structural adjudication. An independent verifier match confirms consistency but does not remove the one-week graph, label-coverage, WETH-wrap, or hook-code limitations.

Rulebook D.4: T4: AMM Wash-Trading Patterns.

E Reference Construction Algorithm

The rulebooks in Appendix D define the admissible evidence, operations, and decision criteria for each task. Algorithm 1 summarizes the offline construction and verification workflow. It operates on fixed item specifications; the English question is a presentation of that specification rather than executable input.

Algorithm 1: Verified reference construction

Input: candidate specifications $\mathcal{C}$; task rulebooks $\mathcal{R} = \{R_{\mathrm{T1}}, \ldots, R_{\mathrm{T4}}\}$; frozen RPC snapshot S_{RPC}; permitted Dune tables S_{Dune}.
Output: balanced verified benchmark $\mathcal{B} = \{(r_i, a_i^\star, E_i^\star)\}_{i=1}^{160}$.

$\mathcal{B} \leftarrow \emptyset$
for each $r_i \in \mathcal{C}$ **do**
  $t_i \leftarrow \mathsf{Task}(r_i)$; $R_i \leftarrow \mathsf{Rules}(r_i, \mathcal{R})$
  $\widehat{E}_i \leftarrow \mathsf{CollectEvidence}(r_i, S_{\mathrm{RPC}}, S_{\mathrm{Dune}}, R_i)$
  $X_i \leftarrow \mathsf{NormalizeDecodeOrder}(\widehat{E}_i, R_i)$
  $\widehat{a}_i \leftarrow \mathsf{Compute}_{t_i}(r_i, X_i, R_i)$
  if $\mathsf{RequiresAdjudication}(r_i)$ **then**
    $\widehat{a}_i.\mathrm{verdict} \leftarrow \mathsf{Adjudicate}_{t_i}(r_i, X_i, \widehat{a}_i, R_i)$
  end if
  $(\widetilde{a}_i, \widetilde{E}_i) \leftarrow \mathsf{IndependentVerify}_{t_i}(r_i, S_{\mathrm{RPC}}, S_{\mathrm{Dune}}, R_i)$
  $m_i \leftarrow \mathsf{MatchRequired}((\widehat{a}_i, \widehat{E}_i), (\widetilde{a}_i, \widetilde{E}_i))$
  if $m_i = \mathrm{false}$ **then**
    $\mathsf{MarkForReview}(r_i)$; **continue**
  end if
  $(a_i^\star, E_i^\star) \leftarrow (\widehat{a}_i, \widehat{E}_i)$
  $\mathcal{B} \leftarrow \mathcal{B} \cup \{(r_i, a_i^\star, E_i^\star)\}$
end for
require $|\{r_i \in \mathcal{B} : \mathsf{Task}(r_i) = t\}| = 40 \quad \forall t \in \{\mathrm{T1}, \mathrm{T2}, \mathrm{T3}, \mathrm{T4}\}$
return $\mathsf{Freeze}(\mathcal{B})$

CollectEvidence retains the source selectors needed to reproduce each item. RPC records supply ledger calculations. Frozen Dune rows are admitted only where the item-specific rulebook calls for T2 label evidence, T3 cross-checking or evidence grading, or T4 service-address exclusion. They do not replace an RPC-derived ledger calculation.

The task operator $\textsc{Compute}_t$ denotes token-flow, state-comparison, aggregate, and diagnostic computations for T1; label, flow, structural, and coverage computations for T2; interaction reconstruction and aggregate statistics for T3; and swap-group reconstruction plus pool- or protocol-level screening for T4. IndependentVerify is implemented separately from reference construction and recomputes the required fields and supporting evidence fields from the frozen sources. A mismatch prevents release.

The final machine-readable benchmark stores the typed references, scoring criteria, and verification status. Supporting evidence records remain in the corresponding construction artifacts and are withheld from the evaluated LLM. Independent verifier agreement establishes reproducibility under the snapshot and item-specific rulebook; it does not establish the external validity of labels or the completeness of the underlying data.

F Benchmark and Answer Composition

F.1 Benchmark profile

The benchmark contains 160 items across four on-chain tasks, with 40 items per task. Each item presents one English question and has one structured reference. Together, the 160 references contain

943 required fields and 1,086 typed scalar values after lists and records are recursively expanded. These quantities correspond to three different denominators: questions for question accuracy, required fields for field accuracy, and scalar values for the scale of answer content. Scalar values are not independent questions or evaluation samples.

Table 10: Task-wise benchmark composition and reference scale. Computational items request derived values without a claim verdict; adjudicative items also request a verdict under the assigned rulebook.

Task	Items	Computational	Adjudicative	Required fields	Scalar values
T1	40	16	24	231	263
T2	40	9	31	185	227
T3	40	12	28	262	281
T4	40	21	19	265	315
Total	**160**	**58**	**102**	**943**	**1,086**

F.2 Capability coverage

Table 11 groups the 23 stored `capability_axis` labels into five reporting families. Each item has exactly one source label, so each task column sums to 40. The families are a presentation taxonomy rather than an empirical skill scale.

Table 11: Fixed mapping from the 23 source capability labels to the five reporting families used for capability analyses.

Reporting family	Source capability labels	T1	T2	T3	T4
Claim verdicts	Verdict: supported; indeterminate; contradicted; out of scope; supported + indeterminate; supported + contradicted	23	30	27	17
Aggregation & statistics	Evidence computation; basic aggregation; group aggregation; pivot aggregation; statistical estimation	6	3	4	9
Relations & set operations	Multi-table join; bidirectional join; set operations; deduplication	5	1	3	3
Protocol decoding & order	Event decoding; ordering; record identification	1	1	3	7
Robustness & scope	Thresholding; numeric precision; exception handling; data-limit awareness; reproducibility awareness	5	5	3	4

F.3 Claim-verdict semantics

Many items fall into the *Claim verdicts* family because practical on-chain questions often require deciding whether heterogeneous and incomplete evidence supports a bounded claim, rather than only computing a descriptive aggregate. Table 12 states the closure condition for each scalar verdict.

Table 12: Operational meaning of the four scalar claim verdicts.

Verdict	Interpretation under the assigned rulebook
`supported`	All required positive conditions hold and no disqualifying condition is observed.
`contradicted`	Direct evidence violates a necessary condition or establishes a defined opposing condition; missing support alone is insufficient.
`indeterminate`	The item is in scope, but incomplete, conflicting, temporally mismatched, or otherwise insufficient evidence prevents closure.
`out_of_scope`	The predicate, protocol, asset behavior, observation interval, or required evidence lies outside the task's declared domain.

The compound construction labels *supported + indeterminate* and *supported + contradicted* identify multi-target or multi-claim items that cover both outcomes within one item. They describe coverage rather than additional verdict values.

F.4 Difficulty and answer shape

Table 13: Difficulty and structured-answer diagnostics for the frozen release. Multiple requested values or subclaims remain part of one item.

Release diagnostic	Count
Difficulty labels	17 easy; 60 medium; 83 hard
Top-level answer shape	157 of 160 items require multiple fields; median 5; maximum 16
Adjudicative verdict shape	102 adjudicative items: 95 scalar verdicts and 7 list- or key-structured multi-verdict answers
Scalar verdict references	31 supported; 25 contradicted; 27 indeterminate; 12 out of scope

Reference matching criteria

Integers	Exact match in the declared unit.
Addresses	Case-normalized without changing the address value.
Lists and records	Meaningful list order and all required record keys are preserved.
Decimals	Unless an item declares another tolerance, $\lvert x - \hat{x}\rvert \le 10^{-6}\max(1, \lvert\hat{x}\rvert)$.
Question accuracy	Every required field must match under its declared type and rule.

Scoring Specification F.1: Required-field matching criteria.

An independent verifier matches the reference for every item under the same snapshot and task rulebooks, checking calculation and serialization. A separate human expert then reviews the results of the independent verifier as an additional validation step. These checks do not establish the external validity of the underlying sources.

F.5 Task definitions and claim units

T1: Asset Flows. For an address set U and asset X, the transfer-based net change over observation window W is the amount entering U from outside minus the amount leaving U:

$$\Delta(U, X, W) = \sum_{e \in \mathcal{T}(X,W)} v_e\big(\mathbf{1}[\mathrm{to}(e) \in U] - \mathbf{1}[\mathrm{from}(e) \in U]\big), \tag{1}$$

where $\mathcal{T}(X, W)$ contains ERC-20 transfers of X within W. Transfers internal to U cancel. The equation gives the transfer component; the item's asset accounting incorporates WETH wrapping and unwrapping and Uniswap V4 ERC-6909 claim minting and burning where required, without double counting a movement. Non-standard assets whose changes cannot be recovered from transfers are restricted under the applicable rule. When state anchors are permitted, the boundary balance difference is reported separately from the ledger-derived net flow.

T2: Exchange Attribution. Common control, control by a centralized exchange, and control by a named exchange are distinct predicates. A supported name may imply an exchange role, but a role alone does not identify an operator. Absence from a label table or failure to find a linkage path does not establish a negative claim.

T3: MEV-Related Interactions. The task tests operational conditions for sandwich, cyclic-arbitrage, and liquidation subtypes rather than asking whether an interaction is MEV in the abstract (Qin et al., 2022). A sandwich requires ordered front- and back-running swaps around a third-party trade; cyclic arbitrage requires a multi-pool route within one transaction and the beneficiary's net flow; liquidation requires debt repayment, collateral receipt, successful execution, and any item-specific state conditions.

T4: AMM Wash-Trading Patterns. An item supplies a fixed swap group G. For asset a, its closure ratio uses the signed change $\delta(s, a)$ in swap s from the pool's perspective:

$$\rho_G(a) = \frac{\left|\sum_{s \in G} \delta(s, a)\right|}{\sum_{s \in G} |\delta(s, a)|}. \tag{2}$$

For the pool's two assets a_0 and a_1, the closure condition is

$$\min\{\rho_G(a_0), \rho_G(a_1)\} \leq 0.05, \tag{3}$$

using exact rational arithmetic. An undefined ratio with a zero denominator does not pass. Closure alone is insufficient: the rule also checks group adjacency, temporal span, direction reversal, actor linkage, protocol support, and alternative explanations such as MEV or liquidity adjustment. The label is an operational classification, not a finding of intent or legal wrongdoing (Gan et al., 2024).

G Task-wise Question and Answer Examples

The examples below are verbatim English items from the frozen 160-item benchmark. Each example separates the evaluated question from its held-back structured reference and identifies the evidence admissible for that item. The values shown are the complete required fields for the selected reference.

G.1 Asset flows and holdings

T1: asset-flow computation

Item profile. Computation · easy · three integer fields

Question
For address `0x1ab4973a48dc892cd9971ece8e01dcc7688f8f23`, between blocks 25369414 and 25369513, compute how much token `0xa0b86991c6218b36c1d19d4a2e9eb0ce3606eb48` it received, how much it sent, and therefore its net increase or decrease. Report all three as integers in the token's base unit.

Structured reference

Field	Reference value
`inflow_raw`	`91095567`
`outflow_raw`	`70274909357`
`flow_delta_raw`	`-70183813790`

Admissible evidence. `logs`.

Example G.1: Complete question–answer example for the T1 asset-flow task.

G.2 EXCHANGE ATTRIBUTION

T2: exchange attribution

Item profile. Adjudication · hard · structured record

Question
Is address `0x28c6c06298d514db089934071355e5743bf21d60` controlled by an exchange? Report the value produced by each of the three methods and the final adjudication.

Structured reference

Field	Reference value
techniques	3
label_sources	1
sweep	748
in_d	222405
out_d	15582
claim_verdict	supported
label_status	silver

Admissible evidence. `blocks`, `receipts`, and the Dune label snapshot.

Example G.2: Complete question–answer example for the T2 exchange-attribution task.

G.3 MEV ADJUDICATION

T3: sandwich adjudication

Item profile. Adjudication · hard · structured record

Question
Adjudicate whether the first and last of the three consecutive swaps in block 25376136 of Uniswap V2 pool `0x000dd0b1528a396c6ea2e09796a7b4a534e9e37b` form a sandwich. Check round trip, common owner of the two legs, third-party middle transaction, same-block ordering, and absence of an accompanying liquidation.

Structured reference

Field	Reference value
reciprocal	true
same_sender	true
victim_third_party	true
same_block	true
liq_absent	true
claim_verdict	supported
label_status	anchor_gold

Admissible evidence. `logs`, `blocks`, and `receipts`.

Example G.3: Complete question–answer example for the T3 sandwich-adjudication task.

G.4 AMM WASH-TRADING ADJUDICATION

T4: AMM pattern adjudication

Item profile. Adjudication · hard · structured record

Question
In Uniswap V2 pool `0x1d88f9632f83ba8516f7a457015babc3b86870e7`, a same-sender run beginning at block 25405824 contains ten trades. Adjudicate whether it is wash trading, state whether each of the five conditions passes, and give the final conclusion.

Structured reference

Field	Reference value
`span`	`9`
`min_ratio`	`0.0`
`single_initiator`	`true`
`mev_checks`	`[0, 0, 0]`
`claim_verdict`	`supported`
`label_status`	`anchor_gold`

Admissible evidence. `logs`, `blocks`, and `receipts`.

Example G.4: Complete question–answer example for the T4 AMM wash-trading task.

H TASK-SPECIFIC EXPERIMENTAL PROMPTS

The following pages present the English system prompts for T1–T4. Each prompt is addressed to the LLM that answers the benchmark items: it states how to read the question and the supplied evidence, what to compute and in which order, how to reach a verdict and evidence grade, which pitfalls to avoid, and which values an answer may take. Repeated metadata and the names of the two verdict-output fields are omitted. The author-facing rulebooks for each task are in Appendix D, and each prompt heading names its rulebook; the T2 prompt's *same entity* and *specific name* correspond to the rulebook's *same controller* and *named exchange*. The Output contract and Raw data sections were appended, in that order, to the end of every prompt; they are printed once, before T1, with Raw data first and the Output contract tables condensed, preceded by a short *Before answering* list that collects instructions stated across the prompts, and the end of each prompt marks where they follow.

Common Section Appended to Every System Prompt (T1–T4)

```
## Before answering

1. Read `answer_fields` and the schema in the question. Answer only those
    fields, with the stated types.
2. Check the task's scope conditions first. Out of scope is not a negative.
3. Apply the task's verdict order and stop at the first matching row.
4. A verdict question takes a verdict and a grade together, from the
    allowed pairs in the Output contract.

## Raw data

W26 is Ethereum mainnet blocks 25,369,414--25,419,597
    (2026-06-22--2026-06-28). Use raw responses from a single archive-node
    JSON-RPC; there are no derived ledger tables.

| Name | RPC method | Contents |
|---|---|---|
| `blocks` | `eth_getBlockByNumber` | blocks and transaction bodies |
| `receipts` | `eth_getBlockReceipts` | status, `gasUsed`, `
    effectiveGasPrice`, and logs |
| `logs` | `eth_getLogs` | address, topics, and data |
| `traces` | `trace_block` | internal calls, creations, and self-destructs
    |

**No token-transfer table is provided.** Decode `Transfer` directly from `
    logs`:

- 3 topics: ERC-20; amount in `data`;
- 4 topics: ERC-721; token ID indexed;
- 1 topic: nonstandard form with all three values in `data`.

**No contract list is provided.** A successful `create` in `traces`
    supplies the deployed address and bytecode.

Handle all values as integers in raw token units. Write ratios to the
    number of digits requested.

## Output contract

| Verdict | Meaning | Allowed grade |
|---|---|---|
| `supported` | evidence supports the claim | `anchor_gold`, `flow_gold` (
    T1 only), `silver` |
| `contradicted` | evidence opposes the claim | `anchor_gold` for direct
    negative evidence; `contested` for weak opposition |
| `indeterminate` | evidence is weak or conflicting, or label timing does
    not cover the claimed period | `contested` |
| `out_of_scope` | the task does not cover the question | `out_of_scope` |

Grades: `anchor_gold` = confirmed by an external anchor or direct
    confirmation; `flow_gold` = confirmed from the ledger alone without an
    absolute-balance anchor; `silver` = conditions pass, but some evidence
```

```
    uses estimation or heuristics; `contested` = evidence conflicts or does
     not close.
```

Prompt H.1: Shared instructions appended to all four task-specific system prompts.

T1 System Prompt: Asset Flows Rulebook: Appendix D.1

```
## Task

An address set `U`, a token `X`, and a block window `[b0, b1]` (inclusive)
    are given.
The claim has the form "`U`'s holdings of `X` changed by Δ during `[b0, b1
    ]`". Adjudicate this claim and answer the values used in the verdict.
A question that does not ask for a verdict computes only the requested
    values.

## Holdings computation

### Ledger net: `flow_delta_raw`

Sum in-window `Transfer`s by direction:
  - inflow `inflow_raw` = sum of `value` where `to ∈ U` and `from ∉ U`
  - outflow `outflow_raw` = sum of `value` where `from ∈ U` and `to ∉ U`
  - transfers that move only within `U` (including self-transfers) are
    counted in neither inflow·outflow
  - if the sender is `0x0...0` it is a mint, if the receiver is `0x0...0`
    it is a burn. Both are included in the ledger net
  - if `X` is WETH (`0xc02aaa39b223fe8d0a0e5c4f27ead9083c756cc2`), add the
    sum of `wad` in `Deposit(dst∈U)` and subtract the sum of `wad` in `
    Withdrawal(src∈U)`.
    These two events do not emit a `Transfer`
  - **Uniswap V4 claims are holdings too.** Among the PoolManager (`0
    x000000000004444c5dc75cb358380d2e3de08a90`) ERC-6909 `Transfer(address
    caller, address indexed from, address indexed to, uint256 indexed id,
    uint256 amount)` events, those with `id = uint256(X)` are summed with
    the same direction rule as above (`topics[1]`=from · `topics[2]`=to · `
    topics[3]`=id · second word of `data` = amount). If from is `0x0...0`
    it is a claim mint, if to is `0x0...0` it is a claim burn
  - this claim portion is `erc6909_delta_raw`, the rest (`Transfer` net +
    WETH correction) is `erc20_delta_raw`, and `flow_delta_raw` = `
    erc20_delta_raw` + `erc6909_delta_raw`

### Archive anchor: `balance_delta_raw`

Only when archive access is available, compute `Σ_{a∈U} balanceOf(a)@b1 −
    balanceOf(a)@(b0−1)` plus the PoolManager claim balance delta `Σ_{a∈U}
     PoolManager.balanceOf(a, uint256(X))@b1 − ...@(b0−1)` (selector `0
    x00fdd58e`). Adopt it only when two different providers return the same
     value.

If the claimed window extends beyond W26, `flow_delta_raw` counts only the
    part inside W26 (it is not used in the verdict, since ① catches it).
    If `X` is not an ERC-20, there are no 3-topic `Transfer`s and it is 0.

All values are decimal integers in the token's raw units. Negative values
    are allowed and no commas are used. `value` can have up to 78 digits,
    so it is handled as an integer.

### Archive queries

Each question states whether archive-node queries (`eth_call` · `
    eth_getCode`) are available. Use them only when explicitly available;
    otherwise they are unavailable.

| Name | RPC method | Contents |
|---|---|---|
```

```
| `archive_eth_call` | `eth_call` (token balanceOf · PoolManager `balanceOf
    (address,uint256)`) · `eth_getCode` --- at a specified block | balance·
    bytecode at an arbitrary block. Only when the question allows it |

## Verdict order

Go through the following in order and **stop at the first place that
    catches.** `reason_code` is the code of the row where you stopped.

### 1. Out of scope: `out_of_scope/out_of_scope`

| `reason_code` | Condition |
|---|---|
| `window_outside_w26` | `[b0, b1]` does not lie entirely within W26
    (25369414 ~ 25419597) |
| `absolute_balance` | asks for the balance level at a point in time, not a
     change |
| `unsupported_asset` | `X`'s `Transfer` has 4 topics (ERC-721, the third
    indexed value is a tokenId), or it is ERC-1155·native ETH. A V4 claim
    with `id = 0` (the native ETH currency, `X = 0x0...0`) is also not an
    ERC-20 and belongs here |
| `selfdestruct_token` | `X` self-destructed within the window |

Even if the value can be read via the archive, if ① catches it is out of
    scope. The archive is used only to confirm boundary balances for claims
     inside W26.

### 2. Holds unresolved by an anchor: `indeterminate/contested`

| `reason_code` | Condition |
|---|---|
| `address_set_not_fixed` | `U` is not fully enumerated ("A etc.") or
    includes addresses chosen by looking at in-window transfers (e.g. "the
    counterparties A exchanged with in this window"). The address set must
    be fixed before computation |
| `position_frozen` | there is a `Paused`·`Unpaused` of `X` within the
    window and `U`'s `X` balance at that block is greater than 0 (without
    an archive: the cumulative in-window net transfers up to that block >
    0, or there are transfers after it) |
| `mint_burn_on_u` | there is a `0x0 → U` mint or a `U → 0x0` burn within
    the window, and the claim explains that change by a cause other than
    mint·burn, such as buying·accumulation. If the claim states **only the
    amount of change**, do not stop here; go on to ③ |
| `proxy_upgrade_in_window` | there is an `Upgraded` of `X` within the
    window. A claim over a partial window that does not include the upgrade
     block `ub` (`[b0, ub−1]` or `[ub, b1]`) is not caught by this row |
| `owner_change_on_u` | the previous·new owner in an `OwnershipTransferred`
     of `X` within the window is in `U` |
| `nonstandard_no_anchor` | `X` is nonstandard and the archive is not
    available in this question |

### 3. Claim-value comparison

| Case | Result | `reason_code` |
|---|---|---|
| archive available · Δ ≠ `balance_delta_raw` | `contradicted` / `
    anchor_gold` | `claim_vs_anchor` |
| archive not available · Δ ≠ `flow_delta_raw` | `contradicted` / `
    contested` | `claim_vs_ledger_only` |

Even with the same magnitude, a different direction (increased/decreased)
    is a different value. Even for a nonstandard token, if the archive
    balance delta directly refutes the claim, it is `claim_vs_anchor`.

### 4. Matching claim: `supported`; select the first grade

| Condition | Grade | `reason_code` |
|---|---|---|
| `privilege_capable = true` | `silver` | `capability_detected` |
| `privilege_capable = unknown` | `silver` | `capability_unknown` |
```

```
| `X` emitted `TransferShares(address,address,uint256)` in W26 | `silver` |
     `nonstandard_cap` |
| archive available · Δ = `balance_delta_raw` ≠ `flow_delta_raw` | `silver
    ` | `balance_not_ledger` |
| confirmed by the archive anchor | `anchor_gold` | `anchor_match` |
| ledger only, without an archive | `flow_gold` | `ledger_only` |



## Properties used in the verdict

### `privilege_capable`: `true | false | unknown`

- `true` : `X`'s code contains one of the eight selectors below, or `X`
    emitted `Paused`·`Unpaused`·`Upgraded`·`OwnershipTransferred` anywhere
    in W26 (exercise is evidence of capability)
  - `0x40c10f19`(mint) · `0x9dc29fac`(burn) · `0x42966c68`(burn) · `0
    x8456cb59`(pause) · `0x3f4ba83a`(unpause) · `0x3659cfe6`(upgradeTo) ·
    `0x4f1ef286`(upgradeToAndCall) · `0xf2fde38b`(transferOwnership)
  - selectors are collected by scanning the bytecode from the start,
    skipping the operand length of PUSH-family opcodes, and gathering the
    operands of PUSH4
- `false` : the code was obtained and neither of the above holds
- `unknown` : the code cannot be obtained
- an address without code (`0x0...0`·EOA) yields empty code, hence `false`
- code source --- for tokens deployed in W26, the result code of a
    successful `create` in `traces`; for tokens deployed earlier, archive `
    eth_getCode@b1` (only when the question allows it); if two providers
    return different code, the code cannot be obtained (`unknown`)

### `token_behavior`: `standard_erc20 | weth | nonstandard | not_erc20`

| Value | Meaning |
|---|---|
| `standard_erc20` | a token whose balance moves by exactly the `value` of
    `Transfer` |
| `weth` | WETH --- `Deposit`·`Withdrawal` are included in the ledger net |
| `nonstandard` | emitted `TransferShares(address,address,uint256)` in W26,
     or the archive balance delta differs from the ledger net (rebase·
    transfer tax) |
| `not_erc20` | an asset that is not an ERC-20 --- `Transfer` with 4 topics
     (ERC-721)·ERC-1155·native ETH (V4 currency `0x0...0`). ① `
    unsupported_asset` |

### Uniswap V4 holdings

Uniswap V4 is not a token behavior and **not a hold reason.** Having called
     the PoolManager alone does not stop the verdict --- with flash
    accounting, even if there is no `Transfer` in the middle of a swap,
    settlement (settle·take) leaves an ERC-20 `Transfer`, and the portion
    left in the PoolManager remains as an ERC-6909 claim. Both are counted
    by the definitions above. A value that counts only ERC-20 and omits
    claims is not a holdings change. If a hook contract holds claims, they
    are the holdings of that hook address.

`0x...dEaD` is an ordinary address. A burn is only `to = 0x0...0`. A claim
    of the form "sent to dEaD, hence burned" is scored as a holdings claim
    with dEaD as `U`.

### `change_cause`: `transfer_only | mint | burn | mint_and_burn`

This records whether a mint (`0x0 → U`)·burn (`U → 0x0`) contributed to `U
    `'s in-window change. Only 3-topic `Transfer`s are considered --- if
    none, `transfer_only`.

## Answer fields

Answer only the fields the question selects in `answer_fields`. Amount
    fields are integers in raw units.
```

```
Verdict questions always request `flow_delta_raw` · `token_behavior` · `
    change_cause` · `privilege_capable` · `reason_code`; archive-enabled
    questions also request `balance_delta_raw`.

| Field | Meaning |
|---|---|
| `inflow_raw` · `outflow_raw` · `internal_raw` | inflow sum · outflow sum
    · sum of moves only within `U` |
| `transfer_delta_raw` | net counted from `Transfer` alone (before WETH
    correction) |
| `erc20_delta_raw` · `erc6909_delta_raw` | ERC-20 portion net (including
    WETH correction) · PoolManager ERC-6909 claim portion net |
| `flow_delta_raw` | ledger net = ERC-20 portion + claim portion (WETH
    after correction) |
| `balance_before_raw` · `balance_after_raw` · `balance_delta_raw` |
    holdings at `b0−1` · holdings at `b1` · their difference (token balance
     + PoolManager claim balance) |
| `token_behavior` · `change_cause` · `privilege_capable` | the values of '
    Properties used in the verdict' above |
| `reason_code` | the code of the row where the verdict order stopped (
    exactly the values in the tables above) |

(The common section shown before T1 --- Raw data and Output contract --- is
     appended here.)
```

Prompt H.2: T1 system prompt for Asset Flows.

T2 System Prompt: Exchange Attribution — Rulebook: Appendix D.2

```
## Task

Adjudicate one claim about an address (or an address set). The claim
    belongs to **one** of the three questions below.

1. **Same entity** --- are two addresses controlled by the same party
2. **Exchange control** --- is this address controlled by a centralized
    exchange
3. **Specific name** --- is this address controlled by a specific exchange
    E

The three are different questions. Confirming exchange control does not
    yield a name.
Being the same entity does not make it an exchange either. **"Not E" is not
     "not an exchange".** Failing to find a link does not make them
    different entities either.
Adjudicate the claim sentence as written in the question; do not answer a
    different question instead.

## Evidence that may be used

The ledger has no names. There is **one kind** of label data, used together
     with the ledger. No other external nameplate·tag is used as evidence.

- **labels** --- the frozen benchmark table joining Dune labels to W26
    addresses. Columns used:
  `in_owner_addresses` (listed among Dune owner labels) · `owner_key` · `
    owner_category` ·
  `in_cex_addresses` (listed on the Dune exchange list) · `cex_name` · `
    cex_distinct_name` · `dune_created_at` (label establishment time) ·
  `account_type` (EOA/CA) · `dune_contract_name`.
  For rows without a label, `has_dune_label` is NULL, not false.

### Time --- PROVEN / UNPROVEN

- The only labels counted as evidence are **Dune labels established before
    the start of W26 (2026-06-22 00:00:00 UTC)** (PROVEN).
```

```
- A Dune label created **during or after** W26 is a candidate (UNPROVEN),
    because evidence must cover the target's entire period.
- UNPROVEN labels are **used neither as supporting nor as opposing evidence
    .** This time rule applies to all three questions.

### Evidence families --- a signal identical to a copy counts as one

- **L label family** --- a PROVEN Dune label has the exchange role ("Role"
    below).
- **F flow family** --- sweep-type senders are 10 or more, or ledger role R
     is satisfied.
  Sweep and R come from the same ETH transfers, so even when both stand
    they are **1 family**. Bridges·payments·custody·casinos produce the
    same flow.
- The only families are L and F. The name family is Dune alone, so there is
     never a question of whether two name families are independent of each
    other.
- A name carried over by on-chain structural attribution is a copy of the
    original address's Dune label and is not a family.
- **The absence of a label is not a negative.**

### Fixed definitions

- ETH transfer = a top-level transaction with `value>0`, `receipt_status
    =1`, and nonempty `to_address`. Exclude internal calls (traces) unless
    the question asks.
- Sweep-type sender = an address that sent at least 3 transfers to the
    target and no ETH elsewhere in W26.
- Inflow counterparties = distinct ETH senders to the target; outflow
    counterparties = distinct ETH recipients from the target.
- Ledger role R = at least 1,000 inflow and 100 outflow counterparties.
### Role --- read only from PROVEN labels

- **Exchange role**: listed on the Dune exchange list (`in_cex_addresses`)
    and the category is not in the other-service vocabulary below, or `
    owner_category` is `Centralized Exchange`.
- **Internal label conflict**: listed on the list but the category is in
    the other-service vocabulary → goes to a verdict hold. Do not read it
    as the exchange role from the listing alone, nor as a different service
     role from the category alone.
- **Other service role**: off the list and `owner_category` **exactly
    matches** one of the following ---
  `Decentralized Exchange`·`Lending`·`Oracle`·`Yield Vaults`·`Derivatives`·
    `Bridge`·`Staking`·`Token Issuance`·`Governance`·
  `Creators and brands`·`Payment Processing`·`Insurance`·`Gaming`·`Casino`·
    `Privacy Services`·`Asset Management`·`Perpetuals`·`MEV`.
  `Decentralized Exchange` contains `Centralized Exchange` as a substring,
    so partial matching is not used.
- Everything else (including off-list `Custody Services`, including no
    label) is **role unknown**. If a listed address's category is a term
    outside the vocabulary above (e.g. `Custody Services`), it is not a
    conflict but the exchange role.

## Verdict criteria

Supporting and opposing evidence are separated as follows.
- If there is **no** PROVEN evidence supporting the claim and a PROVEN
    label points to a different role·different name → `contradicted`/`
    contested`.
- If supporting and opposing evidence are **both** present (they oppose
    each other) → `indeterminate`/`contested`.



### Exchange control

| Condition | Verdict | Grade |
|---|---|---|
```

```
| L and F both stand (2 families) **and** R is satisfied **and** there is
    no other service role·internal label conflict | `supported` | `silver`
    |
| no L and a PROVEN label has a different service role | `contradicted` | `
    contested` |
| everything else (1 family or fewer · R not satisfied · internal label
    conflict · only UNPROVEN labels · no label) | `indeterminate` | `
    contested` |

### Specific name `E`

| Condition | Verdict | Grade |
|---|---|---|
| two or more mutually independent PROVEN label families support E, the
    role matches and there is no conflict --- this data has only one PROVEN
     family (Dune), so this does not stand | `supported` | `silver` |
| no PROVEN evidence supports E and a PROVEN label points to a different
    name | `contradicted` | `contested` |
| everything else (only 1 PROVEN family supports · support and opposition
    oppose each other · timing unconfirmed) | `indeterminate` | `contested`
     |

### Same entity

| Condition | Verdict | Grade |
|---|---|---|
| a directly confirmed root **and** two or more different relation types
    support the same control --- this data has no direct confirmation (
    operator disclosure·signature·controlled deposit/withdrawal test), so
    this does not stand | `supported` | `silver` |
| everything else --- the same however many relation types stand. Failing
    to find a link is not opposing evidence | `indeterminate` | `contested`
     |

Count relation types, not instances:

- transfer: one side sends ETH to the other;
- common funding source: the same address sends ETH to both;
- same deployer: the same address deployed both; and
- access control: an owner-added event was emitted.

Having sent to an exchange many times is not evidence of a deposit address.

### Address-set and negative claims

Adjudicate each address-set element with the same table, then combine:

- any `contradicted` element: `contradicted/contested`;
- all elements `supported/silver`: `supported/silver`;
- otherwise: `indeterminate/contested`.

Negative claims use the same evidence rules; label absence does not support
     negation.

`anchor_gold` cannot arise from this data, positive or negative. It
    requires a target-period official operator disclosure, verifiable
    signature, controlled deposit/withdrawal test, or official
    investigative material; none exists. A label category is not direct
    confirmation.

## Out of scope (`out_of_scope/out_of_scope`)

The following claims are not adjudicated. If the out-of-scope type is asked
    , answer with the value in parentheses.
- a person's real identity (`personal_identity`)
- the link between an exchange's internal customer account and an address
    (`internal_customer_account`)
- a claim that an exchange's entire set of wallets is completely contained
    in some address set (`wallet_completeness`)
- control in a period outside W26 (including the future) (`future_control`)
```

```
If in scope, it is `none`.

### Compound claims

If one claim bundles an in-scope element (the three questions above) with
    an out-of-scope element, adjudicate the in-scope element first with the
     tables above.
If the in-scope element is `contradicted`, the whole claim is `contradicted
    `/`contested` (if one bundled claim is false, the whole is false).
Otherwise the whole claim is `out_of_scope`/`out_of_scope` because of the
    out-of-scope element. Retroactively applying a label without source·
    validity period to W26 is handled by the evidence-use rule ("Time"
    above), not by the whole claim.

## How to answer in T2


- Dictionaries keyed by address use **lowercase addresses** as keys.
- Strings such as categories·names are written exactly as they appear in
    the label data.
- If a date is asked, answer as `YYYY-MM-DD`.
- "There are no derived tables" in the common Raw data section refers to
    the ledger. T2 additionally uses the one labels table above.

(The common section shown before T1 --- Raw data and Output contract --- is
     appended here.)
```

Prompt H.3: T2 system prompt for Exchange Attribution.

T3 System Prompt: MEV-Related Interactions Rulebook: Appendix D.3

```
## Task

Three types are adjudicated **separately**. One question fixes one of the
    three types and one claim sentence. **"Is this MEV" is not asked.**

| Type | Verdict unit | What is examined | Source of profit |
|---|---|---|---|
| **Sandwich** | an **ordered pair** (front, back) in the same pool | is it
     a round trip by one entity, with a third party's same-direction trade
    in between executed at a worse price | the price loss borne by the
    victim |
| **Arbitrage** | a **transaction (tx)** | did it close a cycle across
    several pools and increase the base asset | price differences between
    pools |
| **Liquidation** | a **transaction (tx)** | is it a successful liquidation
     that repaid someone else's debt and took the collateral | the
    liquidation bonus set by the protocol |

Adjudicate only structural, attribution, and economic conditions
    confirmable on the ledger. Do not infer intent, victim-loss amounts,
    profit amounts, or counterfactuals such as "had that transaction not
    existed".

## Verdict targets

| Type | Target |
|---|---|
| Sandwich | every ordered pair of decoded swaps in the same pool where the
     second comes later in `(block_number, transaction_index, log_index)`
    order and **the block-number difference is between 0 and 5 inclusive**.
     **Two swaps within the same transaction are also a pair.** No
    direction·sender·victim·success condition |
| Arbitrage | transactions with **2 or more** decoded swaps |
```

```
| Liquidation | transactions in which one of the 8 liquidation events below
     was emitted or that have an internal call (a call in `traces`)
    starting with one of the 5 liquidation selectors |

**Decoded swap = the 7 event signatures below** (the swap surface). Other
    signatures (e.g. the swap events of PancakeSwap V3 · Bancor) are not
    decoded.

| venue | Signature | Pool key |
|---|---|---|
| Uniswap V2 | `Swap(address,uint256,uint256,uint256,uint256,address)` |
    log address |
| Uniswap V3 | `Swap(address,address,int256,int256,uint160,uint128,int24)`
    | log address |
| Uniswap V4 | `Swap(bytes32,address,int128,int128,uint160,uint128,int24,
    uint24)` | log address + `topics[1]` (poolId) |
| Curve | `TokenExchange(address,int128,uint256,int128,uint256)` | log
    address |
| Curve | `TokenExchangeUnderlying(address,int128,uint256,int128,uint256)`
    | log address |
| Curve (uint256-index variant) | `TokenExchange(address,uint256,uint256,
    uint256,uint256)` | log address |
| Balancer | `Swap(bytes32,address,address,uint256,uint256)` | log address
    + `topics[1]` (poolId) |

`topics[1]` is the first indexed argument after topic0. **A pool is not a
    contract address** --- the V4 PoolManager holds thousands of pools at
    one address, and the Balancer Vault also holds many pools at one
    address.

## Common swap decoding

| venue | Pool-side net change (token0, token1) | Swap recipient address |
|---|---|---|
| UniV2 | `amount0In − amount0Out`, `amount1In − amount1Out` (data words
    0~3, uint256) | the last indexed `to` in `topics` |
| UniV3 | data words 0·1 read as int256, as is | indexed `recipient` |
| UniV4 | data words 0·1 (int128) are **user-side**, so flip the sign |
    none (PoolManager settlement) |

## Uniswap V4 gate

- **Pool information (PoolKey)** is read from the `Initialize(bytes32,
    address,address,uint24,int24,address,uint160,int24)` log of the
    PoolManager `0x000000000004444c5dc75cb358380d2e3de08a90`.
  topics = [topic0, poolId, currency0, currency1], data words 0~4 = fee ·
    tickSpacing · hooks · sqrtPriceX96 · tick. Currency `0x0000...0000` is
    native ETH.
  Pools initialized before W26 are queried with archive-node `eth_getLogs`
    from the PoolManager deployment block 21,688,329 (50,000 blocks or
    fewer per call).
- **hook permission bits** = the low 14 bits of the hooks address. Only
    **`0x08` beforeSwapReturnDelta** and **`0x04` afterSwapReturnDelta**
    matter here (see Admissibility).
- **ERC-6909 claims** = the PoolManager's `Transfer(address,address,address
    ,uint256,uint256)` --- topics = [topic0, from, to, id], data = [caller,
     amount].
  id is the currency address written as an integer. If from is 0 it is a
    mint (a claim arises on assets deposited in the PoolManager), if to is
    0 it is a burn.
- **V4 net-amount closure** --- the per-currency sum of the pool-side
    amounts (−amount0, −amount1) over the V4 Swaps in the transaction
    equals, **for every currency**, the amount that entered the PoolManager
     − the amount that left in the same transaction
  (ERC-20 `Transfer` · ETH of successful calls · 6909 burns − mints); then
    the net amount closes.

### Admissibility
```

```
If the PoolKey cannot be verified, or any condition below applies, the
    transaction is `out_of_scope/out_of_scope`; out of scope is not a
    negative.

1. The hook of a swapped pool has a **swap return delta bit (`0x08` or `0
    x04`)** set --- the hook can alter the caller's settlement amount, so
    the Swap event amounts are not the actual settlement.
2. The same transaction contains a PoolManager `ModifyLiquidity` or `Donate
    ` --- the events carry no token amounts, so the net amount cannot be
    closed.
3. The V4 net amount does not close.

A V4 transaction that avoids all three conditions is adjudicated exactly
    like other pools even if it has not a single ERC-20 Transfer (even if
    it settles only via ETH·6909). A pool whose Initialize cannot be found
    is a pool identification failure.

### `v4_status`

If asked, return the first applicable result: `pool_unknown` (a pool whose
    Initialize cannot be found) · `hook_return_delta` (a hook pool with the
     swap return delta bit set) · `v4_liquidity_or_donate` (V4
    ModifyLiquidity/Donate in the same transaction) · `not_closing` (net
    amount not closed) · `closes` (adjudicable). This field is not asked
    for transactions without V4 swaps.

**Direction** = the sign of the pool-side token0 net change. **Execution
    price** = tokens received by the trader ÷ tokens given (integer ratio
    in raw units). Execution prices are compared only between same-
    direction swaps --- the received÷given of an opposite-direction swap is
     in reciprocal units and is not compared.

## Sandwich

### Checks

| Condition | How to check |
|---|---|
| Round trip | the directions of the two swaps (sign of the pool-side
    token0 net change) are opposite |
| One owner (attribution) | ① the senders (`from`) of the two transactions
     match ② shared destination (`to`) --- but **an address the question
    designates as a shared router is not attribution evidence** ③ common
    recipient --- the final recipient addresses of the two swaps coincide (
    excluding the pool address; if the recipient is a shared router, follow
     to the receiver to which that router re-sent tokens or ETH in the same
     transaction) |
| Victim | **between** the two swaps there is a same-direction swap by an
    address other than the front/back sender, and its lowest execution
    price `vmin` < front swap execution price × (1 − $\tau$). **$\tau$ = 0.005** |
| Ordering control | the two transactions are within one block |

### Notes

- A round-trip quantity net gain (amount received back behind > amount
    given in front) is a **weighting item**; it does not decide the verdict
    .
- Same-pool liquidity provision (Mint · ModifyLiquidity) in the front
    transaction and same-pool liquidity withdrawal (Burn · Collect ·
    ModifyLiquidity) in the back transaction are **accompanying features of
     a JIT-liquidity sandwich**. They are not grounds for disqualification.
- The sender of the front/back transactions is not counted as a victim ---
    a victim is by definition a third party.
- What is adjudicated is the satisfaction of the conditions. How much the
    victim lost or the attacker earned requires counterfactuals and is not
    a claim.

### Verdict

| Condition | Verdict | Grade |
```

```
|---|---|---|
| round trip + intermediate third-party worse price (τ) + senders match +
    one block + no liquidation events in the front/back transactions | `
    supported` | `anchor_gold` |
| as above but attribution rests only on ②·③, or the two transactions are
    in different blocks | `supported` | `silver` |
| not a round trip / block difference over 5 / no third-party victim (no
    same-direction swap by another address in between at a price worse by τ
    ) / liquidation events in the front/back transactions / **direct
    evidence** that the two senders are different entities (differing
    addresses alone are not direct evidence) | `contradicted` | `
    anchor_gold` (negative) |

| swaps undecoded · amount parsing failure · attribution unresolved (none
    of ①②③ stands) | `indeterminate` | `contested` |

## Arbitrage

### Beneficiaries and asset accounting

Close the cycle **within one transaction**. Beneficiaries = the transaction
    's `from` and `to`. **Per-asset net change** sums the amount that
    entered the beneficiaries — the amount that left, over the following
    evidence.

1. 3-topic ERC-20 `Transfer`
2. WETH (`0xc02aaa39b223fe8d0a0e5c4f27ead9083c756cc2`) `Deposit(address,
    uint256)` is + for the target address, `Withdrawal(address,uint256)` is
     — (WETH wrapping does not emit a `Transfer`)
3. ETH of successful internal calls --- the `value` of calls in `traces`
    with `call_type = call` and `status = 1`. The value of a `delegatecall`
     repeats the caller's msg.value and is not a move. Child calls under a
    reverted call have status 0.
4. ERC-6909 claims of the PoolManager --- claims received by the
    beneficiary +, claims sent —. Claims are merged into the currency asset
     of the id (currency 0 and WETH are ETH/WETH).

**ETH and WETH are merged into one asset (ETH/WETH).** ETH the beneficiary
    sent to **that block's `miner` is a builder payment**, so it is not
    included in the net change and is recorded separately (a cost
    circumstance). An asset with a net change of 0 neither decreased nor
    increased.

**The net gain (profit) compared against gas cost is limited to flows
    exchanged with the decoded swap contracts.** Only the WETH·ETH the
    beneficiary exchanged with contracts that emitted decoded swap logs is
    summed.
In V4 the PoolManager is that contract --- the WETH·ETH moves between the
    beneficiary and the PoolManager, and the ETH·WETH 6909 claims minted to
     (+)·burned from (—) the beneficiary, are the swap flow.
**Liquidity provision·withdrawal (Mint·Burn·Collect·ModifyLiquidity) flows
    in the same transaction are excluded from the net gain** --- flows
    exchanged with contracts that emitted such events are not swap profit (
    they are provision·return of liquidity principal).
Decreased assets·increased assets (cycle closure·exchange) are determined
    by the per-asset net change above.

### Exchange legs, conversion, and gas

An **exchange leg** is two addresses exchanging different ERC-20 tokens
    with each other within one transaction (among the tokens one side gave
    there is one not received back, and among the tokens received there is
    one not given).
If neither of the two is a contract that emitted a decoded swap log, it is
    an **undecoded leg**. Lending the same token and receiving the same
    amount back (a flash loan) is not an exchange leg.

**Conversion uses only the exchange rates of the decoded swaps within that
    transaction. No external price oracle or market price is used.** If the
```

```
     increased asset is not ETH/WETH and no decoded swap in the transaction
     exchanged ETH·WETH, comparison against gas cost is not possible.
Gas cost = `gas_used × effective_gas_price`.

### Verdict

Apply the first matching row.

| Order | Condition | Verdict | Grade |
|---|---|---|---|
| 1 | a UniV4 non-adjudicable segment or an undecoded-signature segment ("
    Common to the three types") | `out_of_scope` | `out_of_scope` |
| 2 | failed transaction | `contradicted` | `anchor_gold` (negative) |
| 3 | an undecoded leg exists | `indeterminate` | `contested` |
| 4 | both a decreased and an increased asset (an exchange that gave A and
    received B) / single pool | `contradicted` | `anchor_gold` (negative) |
| 5 | no undecoded leg + 0 decreased assets + 3 or more distinct pools +
    success + net gain compared against gas cost (the ETH/WETH flow
    exchanged with the decoded swap contracts, or its conversion at in-
    transaction exchange rates) > gas cost | `supported` | `anchor_gold` |
| 6 | no undecoded leg + 0 decreased assets + (only 2 pools / the increased
     asset cannot be converted to ETH/WETH at in-transaction exchange rates
    ) | `supported` | `silver` |
| 7 | none of the rows above applies (some positive conditions are refuted
    on the ledger but no negative row applies either) | `indeterminate` | `
    contested` |

Passing through several pools alone does not make it an arbitrage.

## Liquidation

### Events and selectors

The 8 liquidation events --- `LiquidationCall(address,address,address,
    uint256,uint256,address,bool)` · `LiquidationCall(address,address,
    address,uint256,uint256,address)` ·
`LiquidateBorrow(address,address,uint256,address,uint256)` · `AbsorbDebt(
    address,address,uint256,uint256)` · `Liquidate(address,address,address,
    uint256,uint256,uint256,uint256)` ·
`Bite(bytes32,address,uint256,uint256,uint256,address,uint256)` · `Bark(
    bytes32,address,uint256,uint256,uint256,address,uint256)` ·
`Liquidate(bytes32,address,address,uint256,uint256,uint256,uint256,uint256)
    `.
The 5 liquidation selectors --- `0x00a718a9` · `0xf5e3c462` · `0x6f70e66f`
    · `0xc3cecfd2` · `0xd8eabcb8`.

### Checks

| Condition | How to check |
|---|---|
| Explicit event | a liquidation event was emitted |
| Role separation | the event's borrower and liquidator differ |
| Debt repayment | the debt asset leaves the liquidator |
| Collateral acquisition | the collateral asset enters the liquidator |
| Flow closure | the two above close within one transaction |

### Verdict

| Condition | Verdict | Grade |
|---|---|---|
| explicit event emitted by a contract on the frozen Dune cross-check list
    + success + role separation + flow in both directions | `supported` | `
    anchor_gold` |
| all conditions pass but the emitting contract is not on the frozen Dune
    cross-check list | `supported` | `silver` --- **not counted as a
    negative** |
| only a selector without an event and the flow does not close / self-
    liquidation / failure / an ordinary repayment directly confirmed | `
    contradicted` | `anchor_gold` (negative) |
```

```
| flow deficit · asset identification failure · **a judgement that requires
     the discount question** | `indeterminate` | `contested` |

### Notes

- Whether the contract is on the frozen Dune cross-check list is stated by
    the question.
- Even if the transaction receipt succeeded, the liquidation call itself
    may have reverted --- check the status of the internal call.
- There is no price data to compare the values of collateral and debt, so
    the discount question cannot be adjudicated.

## Common to the three types

A high in-block priority-fee rank (`effective_gas_price − base_fee_per_gas
    `) or direct payment to the block `miner` is circumstantial ordering
    evidence. It identifies neither MEV type nor satisfied conditions.

The following are `out_of_scope` / `out_of_scope` --- **out of scope is not
     a negative.**
- **UniV4 non-adjudicable segment**: a transaction whose PoolKey cannot be
    verified or that meets one of the three conditions in "Admissibility"
    above (a swap return delta hook pool · V4 ModifyLiquidity/Donate in the
     same transaction · net amount not closed).
  Having no ERC-20 Transfer alone is not out of scope --- if the net amount
     closes via ETH·6909, adjudicate.
- **Undecoded-signature segment**: a transaction with not a single swap
    decoded by the 7 signatures above (the exchange venue cannot be
    identified on the decoding surface). If only **some** decoded swaps are
     missing, it is not out of scope but the undecoded-leg row of the
    arbitrage table.
- Cross-chain · outside W26.

(The common section shown before T1 --- Raw data and Output contract --- is
     appended here.)
```

Prompt H.4: T3 system prompt for MEV-Related Interactions.

T4 System Prompt: AMM Wash-Trading Patterns Rulebook: Appendix D.4

```
## Task

Adjudicate whether a group of swaps in an AMM pool meets the wash-trading
    **observation criteria**.
It does not adjudicate legal wash trading or manipulative intent.
    Computation questions that count across whole pools also appear.

## Group

- A question asking for a verdict gives the group as a list of `(
    transaction_hash, log_index)`. **That list is the group.**
  Do not remove or add swaps --- even if there is a longer consecutive run
    containing the list, adjudicate only the list the question gives.
- **Adjacency**: when the pool's Swaps are sorted by `(block,
    transaction_index, log_index)`, there must be no swap outside the group
     between the group's first swap and
  its last swap. There may be several initiator addresses (`transaction.
    from`).
- Only questions that count across whole pools use the **candidate
    generator**: maximal runs of consecutive swaps with the same `
    transaction.from`.
  Do not apply this generator to the fixed groups supplied in verdict
    questions.
- The `sender` inside the Swap event may be a router, so it is not used as
    the initiator address.

## Swap decoding
```

```
| venue | Swap signature | Pool | Pool-side delta |
|---|---|---|---|
| UniV2 | `Swap(address,uint256,uint256,uint256,uint256,address)` | log
    address | from the 4 data words (amount0In·amount1In·amount0Out·
    amount1Out, uint256), `d = amountIn − amountOut` |
| UniV3 | `Swap(address,address,int256,int256,uint160,uint128,int24)` | log
     address | data words 0·1 read as int256, as is |
| UniV4 | `Swap(bytes32,address,int128,int128,uint160,uint128,int24,uint24)
    ` | the first indexed argument poolId of the PoolManager `0
    x000000000004444c5dc75cb358380d2e3de08a90` | data words 0·1 are **user-
    side** (negative = the amount the user paid into the pool), so the sign
     must be flipped to get the pool side |

- Per-asset `ratio = |Σ delta| / Σ|delta|`. Floating point deviates at the
    boundary, so compute as fractions.
- Other venue signatures --- Curve `TokenExchange(address,int128,uint256,
    int128,uint256)` · `TokenExchange(address,uint256,uint256,uint256,
    uint256)` ·
  `TokenExchangeUnderlying(address,int128,uint256,int128,uint256)`,
    Balancer `Swap(bytes32,address,address,uint256,uint256)` --- these
    venues are out of scope (step 1).

## Verdict order

**Stop at the first matching step.**

### 1. Scope

If any condition applies, return `out_of_scope/out_of_scope`.

- A venue other than UniV2·UniV3·UniV4 (Curve · Balancer, etc.). Out of
    scope even if all numeric conditions are met.
- A UniV4 pool whose hook can alter the swap settlement amounts --- a pool
    with swap return-delta permission under the **UniV4 hook** rule below.
- A contract that emits the UniV2 Swap signature but whose address never
    emitted `Sync(uint112,uint112)` in W26 --- not an official
    UniswapV2Pair.
- The group's swap count is outside 2~10.
- The token's behavior is fee-on-transfer, rebasing, backdoored, or
    unreconstructable.

#### Uniswap V4 hook

Read a pool's hook address from the same PoolManager's `Initialize(bytes32,
    address,address,uint24,int24,address,uint160,int24)` log
(`topics[1]` = poolId, data word 2 = hooks). For pools initialized before
    W26, find the same log with archive `eth_getLogs`.
The hook is used only if `poolId = keccak256(abi.encode(currency0,
    currency1, fee, tickSpacing, hooks))` matches.
- The low 14 bits of the hook address are the permissions. If `
    beforeSwapReturnDelta` (1<<3) or `afterSwapReturnDelta` (1<<2) is set,
    out of scope.
- If there is no hook (address 0) or both bits are clear, in scope. A
    dynamic fee (`fee == 0x800000`) alone does not change the scope.
- If the hook cannot be confirmed, `indeterminate` / `contested`.

### 2. Structure

If any condition is violated on the ledger, return `contradicted/
    anchor_gold` (negative).

- ① adjacent, and in at least one asset there are both a swap with
    positive delta and a swap with negative delta (direction reversal)
- ② `last_block − first_block ≤ 200` (blocks, not seconds)
- ③ `min(ratio_token0, ratio_token1) ≤ 0.05`. Whether **both** assets are
    0.05 or less (`closure_both_assets_pass`) is recorded separately, but
    the verdict uses ③.

### 3. MEV exclusion
```

```
If any exclusion is confirmed, return `contradicted/anchor_gold`.

#### Sandwich

Find two swaps tx1·tx2 in the group that are different transactions in the
    same block and have the same `transaction.from` or the same `
    transaction.to`,
  and are in opposite directions (token0 into the pool and token1 out, and
    the reverse). In the pool sort order there is, between the two, at
    least one swap in the **same direction** as tx1
  emitted by an initiator address **different** from tx1 (that swap may be
    outside the group).
With X = the token tx1 took out of the pool, `X that tx2 put into the pool
    ÷ X that tx1 took out of the pool` is between 0.9 and 1.1 inclusive.

#### Cyclic arbitrage

A transaction in the group emits Swaps of 2 or more pools (pool = the UniV2
    /V3 log address, the UniV4 poolId), and that tx's actor net flow has **
    no negative token and
  one or more positive tokens**.
  - actors = `{transaction.from, transaction.to}`.
  - net flow = the per-token sum in that tx of values that entered the
    actors, minus values the actors sent. Moves between actors are excluded
    . Three things are included ---
    ERC-20 `Transfer(address,address,uint256)` (3 topics), the PoolManager'
    s ERC-6909 `Transfer(address,address,address,uint256,uint256)` (claims,
     token = id),
    ETH value moves of successful traces (including the top-level call,
    excluding delegatecall·staticcall·callcode). WETH `Deposit`/`Withdrawal
    ` are not included.

#### Rebase-style loop

Within one transaction of the group there are a swap in which X entered the
     same pool and a **different** swap in which X left, and the two
    quantities are equal
  (for UniV2 the amountIn·amountOut words, for UniV3 the absolute values of
     the positive·negative delta).

### 4. Initiator-address relation

One initiator passes. For several initiators:

- ETH-transfer graph = top-level transactions with `value>0` · `
    receipt_status=1` · `to_address` not empty · sender≠recipient,
  `block_number ≤ the group's last block`, ignoring direction.
- **Service addresses are not used as intermediate addresses** ---
    addresses with a Dune label in labels (`has_dune_label`, regardless of
    category).
  No other label data is used.
- **Hubs are not used as intermediate addresses either** --- addresses that
    , even without a label, have 100 or more distinct counterparties in the
     whole W26 ETH-transfer graph.
- If the initiator addresses are all connected by a direct transfer (1 hop)
     or by one intermediate address that is not a service address·hub (2
    hops), pass.

- If they connect only through a service address·hub, or do not connect
    within 2 hops, `indeterminate` / `contested`.
  **Failing to find a link is not evidence of different entities** --- it
    does not go to `contradicted`.

### 5. Benign explanation

If any exists, return `indeterminate/contested`.
```

```
- **Multi-pool route**: a tx in the group passes through 2 or more pools
    but is not the cyclic arbitrage of 3 (it may be a router·aggregator
    route).
- **LP-linked**: in the same pool, a Mint/Burn of a tx whose `transaction.
    from` is the group's initiator address lies within `[first_block − 200,
     last_block + 200]`.
  The same applies if it is inside a tx of the group. Signatures --- UniV2
    `Mint(address,uint256,uint256)` · `Burn(address,uint256,uint256,address
    )`,
  UniV3 `Mint(address,address,int24,int24,uint128,uint256,uint256)` · `Burn
    (address,int24,int24,uint128,uint256,uint256)`,
  UniV4 `ModifyLiquidity(bytes32,address,int24,int24,int256,bytes32)` (same
     poolId).
- **Mixed signs within a tx**: within one tx the delta0 signs of the group'
    s swaps are mixed positive·negative, but it is not the rebase of 3.

A mere possibility is not `contradicted`. What is directly confirmed by
    flows is already caught in 3.

### 6. All pass

- one initiator address: `supported/anchor_gold`;
- step 4 passed through an ETH link: `supported/silver`.

## Not used in the verdict

- "an active trading bot" · "an address that trades repeatedly in the same
    pool" · market-maker inventory adjustment · treasury moves ---
  there is no deterministic rule, so they are **not used as benign
    explanations.**
- What a public contract (router·aggregator·shared execution contract) is
    --- not used as verdict evidence.
- MEV labels from other sources.



## Auxiliary statistical screening

Trade-size distributions can narrow down suspicious pools, but **
    statistical results are not grounds for a verdict.**
A pool item with only statistical results is `indeterminate` / `contested`
    even if all three tests fail.
Screen Uniswap V2 pools with at least **300** swaps. Use one incoming-asset
     amount per swap (`amount0In` when nonzero, otherwise `amount1In`),
    take absolute values, and do not run the tests with fewer than **100**
    amounts.

- Benford: fail if the chi-square statistic (df=8) against `P(d)=log10(1+1/
    d)` exceeds 15.507
- Roundness: the share `p` of amounts that are multiples of `10^max(d−3,1)`
     (`d` = number of digits), compared with expected share 0.10 by `z = (p
     − 0.10)/sqrt(0.10·0.90/n)`; fail if z <= 1.96 (no clustering at round
    values)
- Tail index: sort nonzero amounts in descending order, use `k=max(floor
    (0.1n),20)` and the k-th largest value `x_min`, then compute `alpha=k/
    sum(log(x/x_min))` over the top k. The test is unavailable with fewer
    than **50** nonzero amounts and fails when alpha falls outside 1<alpha
    <2

Failing two or more of the three makes it a suspect. In AMMs, stablecoin·
    arbitrage flows contaminate the distribution, so the pools flagged
    statistically and
the pools passing the structural conditions diverge widely.

(The common section shown before T1 --- Raw data and Output contract --- is
     appended here.)
```

Prompt H.5: T4 system prompt for AMM Wash-Trading Patterns.

I Examples of Execution-Time System-Message Assembly

Appendix H gives the task-specific English instructions. The four examples below illustrate task-prompt selection, additional instructions for the in-context path and SQL path, and output format and reference isolation during execution.

Frozen task-prompt selection.

For each item, the harness selects the English T1, T2, T3, or T4 system prompt from the frozen QA directory using the item's task. The input row also carries `source_dataset`; the system instruction requires the LLM to apply the legacy v1 rules for `20_QA` and the v2.1 rules for `30_QA`, without mixing them. The task-specific text is the portion shown in Appendix H.

Example I.1: Frozen task-prompt selection.

In-context path.

The system message appends the exact-arithmetic calculator protocol, the enabled tool descriptions, field-level citation rules, and, for T1, the archive-query protocol to the selected task prompt. The user message then supplies the retrieved evidence, admissible label rows, an in-memory evidence table when enabled, the question, and the required output format. Evidence is presented as input; values not present in the supplied evidence are not admissible answer support.

Example I.2: In-context path.

SQL path.

The system message appends the mounted-table schema, guarded single-statement SQL procedure, exact-arithmetic calculator protocol, enabled tool descriptions, and field-level citation rules to the selected task prompt. For the completed run, an answer requires at least one successful SQL query or an enabled tool call; population questions that encounter a row cap must be re-queried as an aggregate. The user message contains the item question and its required fields, not a reference.

Example I.3: SQL path.

Output format and reference isolation.

Every final response must be one {`"answer": {...}`} object whose keys are exactly the required fields. Numerical fields must either copy a displayed value or use the supplied exact-arithmetic calculator; verdicts must use the prescribed vocabulary. The frozen `ChainQABench_160_answer_key.csv` file, which contains answer and validation metadata, is used for scoring only and is never part of the LLM input.

Example I.4: Output format and reference isolation.

J Evaluation Metrics and Scoring Views

Every one of the 160 items remains in each full-benchmark denominator. Serving failures, empty outputs, missing fields, and values outside the frozen type or tolerance rule receive zero credit. Let $\mathcal{I}$ denote the items, $\mathcal{F}_i$ the required fields for item i, and $m_{if} \in \{0, 1\}$ the field-match indicator. Question accuracy requires every required field to match, while field accuracy gives each required field one unit of credit:

$$\mathrm{Acc}_q = \frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} \prod_{f \in \mathcal{F}_i} m_{if}, \qquad \mathrm{Acc}_f = \frac{\sum_i \sum_{f \in \mathcal{F}_i} m_{if}}{\sum_i |\mathcal{F}_i|}. \tag{4}$$

The main table uses relaxed parsing: a parseable field mapping is retained when its sole defect is omission of the outer {"answer":...} wrapper. The strict view treats that response as unanswered. Both views use the same value, type, tolerance, and missing-field rules. This distinction concerns the outer wrapper and does not change the reference.

To diagnose dependent fields, the evaluation also computes a cascading-failure score (CFS). If $\mathrm{pa}_i(f)$ is the declared set of upstream fields, a field receives cascading credit only when its own value and all prerequisites are correct:

$$c_{if} = m_{if} \prod_{u \in \mathrm{pa}_i(f)} c_{iu}, \qquad \mathrm{CFS} = \frac{\sum_i \sum_{f \in \mathcal{F}_i} c_{if}}{\sum_i |\mathcal{F}_i|}. \tag{5}$$

CFS is an auxiliary diagnostic; question accuracy and field accuracy remain the primary metrics.

An answer attempt has at least one non-null required field. We report its rate (AR) to distinguish an absent answer from an incorrect attempted one:

$$\mathrm{AR} = \frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} \mathbf{1}[\exists f \in \mathcal{F}_i : \; a_{if} \neq \emptyset]\,. \tag{6}$$

K Additional Quantitative Analysis

We report all 11 LLMs under four evidence conditions on the same 160 items, with a single run per LLM and condition. Tables place Raw, +Actions, +Units, and +Retrieval side by side. All breakdowns use strict scoring, except the explicitly marked relaxed results.

K.1 Scoring audit and auxiliary metrics

Table 14 compares relaxed and strict question accuracy and field accuracy and the diagnostics defined in Appendix J. AR measures answer availability; CFS is an exploratory diagnostic dependent on the declared field-dependency map. The benchmark has 943 required fields. The two marked Gemini +Units CFS entries cover only 110 items on the in-context path; full-benchmark values are unavailable and are not imputed. For the two additional Qwen3 reports, CFS is reconstructed as the uniquely recoverable strict correct-field count minus dependency-invalidated fields, divided by 943, and cross-checked against reports with direct CFS columns.

Table 14: Full-benchmark accuracy and diagnostics (%). Q/F denotes question/field accuracy. *CFS covers 110 items on the in-context path only; all other entries use the full benchmark.

(a) Accuracy: Q/F

LLM	Raw		+Actions		+Units		+Retrieval	
	Relaxed	Strict	Relaxed	Strict	Relaxed	Strict	Relaxed	Strict
GPT-5.4-mini	11.2/46.6	11.2/46.6	15.0/49.7	15.0/49.7	11.9/47.5	11.9/47.5	15.6/48.9	15.6/48.9
GPT-5.6-terra	26.2/59.7	26.2/59.7	31.9/63.3	31.9/63.3	35.0/66.5	35.0/66.5	33.1/64.2	33.1/64.2
Gemini-3.1-Flash-Lite	20.0/52.1	20.0/52.1	23.8/57.2	23.8/57.2	26.9/58.5	26.9/58.5	27.5/59.5	27.5/59.5
Gemini-3.8-Flash	53.1/77.4	53.1/77.4	53.1/79.3	53.1/79.3	51.2/79.4	51.2/79.4	50.0/76.2	50.0/76.2
Qwen3-VL-8B-Instruct	5.0/29.9	5.0/29.7	5.6/35.9	5.6/35.9	6.2/34.4	6.2/33.8	7.5/42.6	7.5/42.4
Qwen3-30B-Instruct	10.0/29.1	10.0/29.1	10.0/41.2	10.0/41.2	8.8/35.5	8.8/35.5	10.6/42.2	10.6/42.2
Qwen3.5-9B	10.6/41.9	10.6/41.7	10.6/43.3	10.6/43.1	9.4/40.7	9.4/40.4	15.6/48.3	15.6/48.3
Qwen3.8-27B	32.5/63.4	32.5/63.4	35.0/62.7	35.0/62.7	36.2/65.0	36.2/65.0	35.6/65.0	35.6/65.0
Qwen3.8-27B-Finance	35.6/64.1	35.6/64.1	38.8/67.9	38.8/67.9	40.6/68.2	40.6/68.2	40.6/66.7	40.6/66.7
Gemma4-12B	13.1/29.6	13.1/29.6	13.1/34.7	13.1/34.7	11.2/37.3	11.2/37.3	12.5/37.6	12.5/37.3
Gemma4-31B	23.1/53.8	23.1/53.6	30.6/60.1	30.6/60.1	33.1/62.4	33.1/62.4	33.1/62.8	33.1/62.8

(b) Diagnostics: strict CFS and AR

LLM	Raw		+Actions		+Units		+Retrieval	
	CFS	AR	CFS	AR	CFS	AR	CFS	AR
GPT-5.4-mini	37.0	82.5	39.8	85.6	37.8	85.0	39.2	85.6
GPT-5.6-terra	49.6	83.8	53.8	83.1	56.7	85.0	55.5	85.6
Gemini-3.1-Flash-Lite	42.2	96.9	47.6	98.8	51.6*	98.8	51.3	99.4
Gemini-3.8-Flash	70.2	95.6	72.2	95.6	76.8*	97.5	70.2	96.2
Qwen3-VL-8B-Instruct	21.7	83.1	27.4	81.9	27.4	81.9	32.1	89.4
Qwen3-30B-Instruct	23.6	75.0	33.0	87.5	29.1	85.6	33.2	88.1
Qwen3.5-9B	30.2	96.9	33.0	95.0	30.8	96.2	37.6	96.9
Qwen3.8-27B	53.4	93.1	54.9	94.4	56.5	94.4	56.4	95.0
Qwen3.8-27B-Finance	55.5	96.2	61.0	96.9	61.3	98.1	60.0	97.5
Gemma4-12B	23.5	63.1	27.6	70.0	29.2	75.0	29.6	73.8
Gemma4-31B	44.8	93.8	51.7	97.5	53.8	97.5	54.9	98.1

K.2 TASK RESULTS

T1–T4 partition the benchmark into 40 items each (Table 15). Field accuracy is micro-averaged over each task's required fields. Table 16 further splits +Retrieval results by execution path. These subsets differ in composition; one correct answer changes T1's six-item SQL-path question accuracy by 16.7 pp.

Table 15: Strict question accuracy and field accuracy by task (%); 40 items per task.

(a) Question accuracy

LLM	Raw				+Actions				+Units				+Retrieval			
	T1	T2	T3	T4	T1	T2	T3	T4	T1	T2	T3	T4	T1	T2	T3	T4
GPT-5.4-mini	5.0	15.0	5.0	20.0	7.5	22.5	10.0	20.0	2.5	17.5	10.0	17.5	7.5	25.0	12.5	17.5
GPT-5.6-terra	15.0	42.5	20.0	27.5	15.0	47.5	35.0	30.0	27.5	40.0	35.0	37.5	17.5	42.5	37.5	35.0
Gemini-3.1-Flash-Lite	2.5	32.5	12.5	32.5	12.5	30.0	22.5	30.0	12.5	30.0	25.0	40.0	12.5	32.5	25.0	40.0
Gemini-3.8-Flash	47.5	62.5	45.0	57.5	45.0	65.0	42.5	60.0	40.0	62.5	47.5	55.0	37.5	65.0	45.0	52.5
Qwen3-VL-8B-Instruct	5.0	5.0	5.0	5.0	2.5	5.0	7.5	7.5	2.5	2.5	15.0	5.0	7.5	5.0	12.5	5.0
Qwen3-30B-Instruct	5.0	12.5	15.0	7.5	5.0	7.5	15.0	12.5	7.5	5.0	15.0	7.5	5.0	10.0	10.0	17.5
Qwen3.5-9B	10.0	12.5	10.0	10.0	5.0	12.5	15.0	10.0	5.0	7.5	12.5	12.5	12.5	15.0	15.0	20.0
Qwen3.8-27B	15.0	57.5	20.0	37.5	17.5	52.5	35.0	35.0	17.5	55.0	32.5	40.0	15.0	57.5	30.0	40.0
Qwen3.8-27B-Finance	25.0	57.5	25.0	35.0	22.5	55.0	40.0	37.5	25.0	52.5	40.0	45.0	17.5	57.5	40.0	47.5
Gemma4-12B	5.0	27.5	5.0	15.0	2.5	22.5	15.0	12.5	5.0	20.0	12.5	7.5	5.0	22.5	10.0	12.5
Gemma4-31B	5.0	42.5	15.0	30.0	17.5	42.5	27.5	35.0	15.0	42.5	30.0	45.0	10.0	42.5	32.5	47.5

(b) Field accuracy

LLM	Raw				+Actions				+Units				+Retrieval			
	T1	T2	T3	T4	T1	T2	T3	T4	T1	T2	T3	T4	T1	T2	T3	T4
GPT-5.4-mini	31.2	54.6	45.0	55.9	32.9	53.5	49.2	62.3	33.3	55.1	44.7	57.4	36.4	56.8	44.3	58.9
GPT-5.6-terra	42.4	71.9	61.8	64.1	45.0	72.4	67.6	68.7	53.2	72.4	67.2	73.2	45.9	75.1	65.3	71.3
Gemini-3.1-Flash-Lite	35.9	69.2	46.2	60.0	43.7	65.9	57.3	62.6	42.4	67.0	57.3	67.9	44.2	67.6	59.5	67.2
Gemini-3.8-Flash	73.6	82.7	75.6	78.9	72.3	84.9	74.4	86.4	71.9	84.9	77.1	84.5	67.5	83.2	72.5	82.6
Qwen3-VL-8B-Instruct	28.1	31.4	28.2	31.3	26.8	36.2	47.7	32.1	26.0	29.2	38.9	38.9	29.9	40.5	47.3	49.8
Qwen3-30B-Instruct	10.0	41.6	30.5	35.5	19.1	47.6	48.9	48.7	22.5	47.0	40.5	34.0	24.2	49.2	50.4	44.9
Qwen3.5-9B	33.3	51.9	41.6	41.9	32.9	47.6	44.7	47.2	28.6	46.0	48.9	38.5	37.2	49.2	50.4	55.1
Qwen3.8-27B	42.0	82.7	61.5	70.6	39.0	73.0	69.8	69.1	45.0	80.5	66.4	70.2	43.3	79.5	67.9	70.9
Qwen3.8-27B-Finance	48.9	78.4	66.8	64.5	47.6	80.0	73.7	71.3	49.4	76.2	71.0	76.2	44.2	80.5	71.0	72.5
Gemma4-12B	16.4	54.0	34.0	19.6	24.2	53.5	47.3	18.1	32.0	50.8	45.8	24.1	25.5	53.0	55.7	18.5
Gemma4-31B	31.6	74.6	53.8	57.7	43.7	73.0	59.9	65.7	42.4	70.3	64.5	72.1	38.5	73.0	67.9	71.7

Table 16: Strict question accuracy (%) under +Retrieval by task and execution path. IC denotes the in-context path; header counts give item denominators.

LLM	T1		T2		T3		T4	
	IC (34)	SQL (6)	IC (26)	SQL (14)	IC (25)	SQL (15)	IC (25)	SQL (15)
GPT-5.4-mini	8.8	0.0	38.5	0.0	12.0	13.3	20.0	13.3
GPT-5.6-terra	17.6	16.7	61.5	7.1	36.0	40.0	36.0	33.3
Gemini-3.1-Flash-Lite	11.8	16.7	42.3	14.3	20.0	33.3	36.0	46.7
Gemini-3.8-Flash	38.2	33.3	80.8	35.7	48.0	40.0	52.0	53.3
Qwen3-VL-8B-Instruct	8.8	0.0	7.7	0.0	12.0	13.3	4.0	6.7
Qwen3-30B-Instruct	2.9	16.7	15.4	0.0	4.0	20.0	20.0	13.3
Qwen3.5-9B	11.8	16.7	23.1	0.0	8.0	26.7	12.0	33.3
Qwen3.8-27B	14.7	16.7	65.4	42.9	24.0	40.0	44.0	33.3
Qwen3.8-27B-Finance	17.6	16.7	69.2	35.7	40.0	40.0	52.0	40.0
Gemma4-12B	5.9	0.0	34.6	0.0	12.0	6.7	12.0	13.3
Gemma4-31B	11.8	0.0	57.7	14.3	32.0	33.3	44.0	53.3

K.3 Topics from Figure 3

Tables 17–18 use the exact item memberships of Figure 3. Topics overlap, so their counts do not sum to 160. Each within-topic condition comparison uses the same items.

Table 17: Strict question accuracy (%) for Figure 3 topics (1/2). Parentheses give overlapping item counts.

LLM	Swaps & pools (65)				Asset flows (47)				CEX control (40)			
	Raw	+Actions	+Units	+Retrieval	Raw	+Actions	+Units	+Retrieval	Raw	+Actions	+Units	+Retrieval
GPT-5.4-mini	7.7	10.8	7.7	7.7	4.3	10.6	6.4	8.5	15.0	22.5	17.5	25.0
GPT-5.6-terra	16.9	16.9	21.5	20.0	17.0	25.5	31.9	25.5	42.5	47.5	40.0	42.5
Gemini-3.1-Flash-Lite	13.8	15.4	20.0	18.5	0.0	10.6	12.8	12.8	32.5	30.0	30.0	32.5
Gemini-3.8-Flash	43.1	41.5	43.1	40.0	44.7	51.1	44.7	40.4	62.5	65.0	62.5	65.0
Qwen3-VL-8B-Instruct	1.5	1.5	3.1	1.5	4.3	2.1	8.5	8.5	5.0	5.0	2.5	5.0
Qwen3-30B-Instruct	6.2	10.8	4.6	9.2	2.1	2.1	4.3	4.3	12.5	7.5	5.0	10.0
Qwen3.5-9B	7.7	6.2	6.2	9.2	6.4	4.3	2.1	6.4	12.5	12.5	7.5	15.0
Qwen3.8-27B	20.0	24.6	23.1	21.5	12.8	23.4	21.3	21.3	57.5	52.5	55.0	57.5
Qwen3.8-27B-Finance	21.5	24.6	27.7	32.3	23.4	25.5	29.8	27.7	57.5	55.0	52.5	57.5
Gemma4-12B	9.2	6.2	4.6	7.7	4.3	8.5	8.5	6.4	27.5	22.5	20.0	22.5
Gemma4-31B	18.5	21.5	27.7	32.3	4.3	25.5	23.4	21.3	42.5	42.5	42.5	42.5

LLM	Token semantics (33)				Address links (33)				Wash-trading runs (25)			
	Raw	+Actions	+Units	+Retrieval	Raw	+Actions	+Units	+Retrieval	Raw	+Actions	+Units	+Retrieval
GPT-5.4-mini	0.0	3.0	3.0	0.0	12.1	18.2	15.2	21.2	16.0	16.0	16.0	20.0
GPT-5.6-terra	9.1	12.1	24.2	18.2	39.4	33.3	39.4	39.4	24.0	28.0	32.0	28.0
Gemini-3.1-Flash-Lite	6.1	9.1	12.1	9.1	21.2	18.2	21.2	21.2	20.0	24.0	36.0	32.0
Gemini-3.8-Flash	48.5	48.5	36.4	33.3	57.6	57.6	54.5	60.6	48.0	52.0	48.0	48.0
Qwen3-VL-8B-Instruct	3.0	0.0	0.0	3.0	6.1	6.1	6.1	3.0	4.0	4.0	0.0	4.0
Qwen3-30B-Instruct	3.0	3.0	3.0	3.0	12.1	12.1	9.1	9.1	4.0	16.0	4.0	16.0
Qwen3.5-9B	6.1	0.0	3.0	6.1	15.2	12.1	9.1	12.1	4.0	4.0	4.0	16.0
Qwen3.8-27B	9.1	12.1	15.2	12.1	42.4	48.5	36.4	42.4	32.0	28.0	36.0	36.0
Qwen3.8-27B-Finance	24.2	15.2	21.2	18.2	45.5	51.5	45.5	45.5	36.0	32.0	40.0	44.0
Gemma4-12B	3.0	0.0	0.0	0.0	15.2	12.1	12.1	12.1	8.0	8.0	4.0	12.0
Gemma4-31B	3.0	12.1	15.2	12.1	33.3	33.3	33.3	33.3	24.0	28.0	36.0	40.0

Table 18: Strict question accuracy (%) for Figure 3 topics (2/2). Parentheses give overlapping item counts.

LLM	Holdings (23)				Sandwich (17)				Liquidation (13)			
	Raw	+Actions	+Units	+Retrieval	Raw	+Actions	+Units	+Retrieval	Raw	+Actions	+Units	+Retrieval
GPT-5.4-mini	4.3	0.0	0.0	0.0	11.8	11.8	5.9	17.6	7.7	23.1	23.1	30.8
GPT-5.6-terra	4.3	8.7	21.7	8.7	11.8	29.4	29.4	29.4	38.5	76.9	76.9	84.6
Gemini-3.1-Flash-Lite	0.0	4.3	4.3	4.3	11.8	17.6	17.6	17.6	30.8	53.8	61.5	69.2
Gemini-3.8-Flash	47.8	43.5	39.1	34.8	52.9	41.2	58.8	64.7	69.2	76.9	76.9	69.2
Qwen3-VL-8B-Instruct	4.3	0.0	0.0	8.7	5.9	5.9	0.0	5.9	15.4	23.1	46.2	38.5
Qwen3-30B-Instruct	0.0	0.0	0.0	0.0	11.8	17.6	5.9	5.9	30.8	30.8	38.5	30.8
Qwen3.5-9B	4.3	4.3	0.0	8.7	11.8	17.6	5.9	11.8	15.4	30.8	30.8	38.5
Qwen3.8-27B	8.7	13.0	8.7	4.3	23.5	29.4	29.4	23.5	38.5	61.5	69.2	69.2
Qwen3.8-27B-Finance	13.0	8.7	13.0	8.7	29.4	35.3	41.2	52.9	38.5	76.9	84.6	69.2
Gemma4-12B	4.3	0.0	0.0	0.0	5.9	11.8	0.0	11.8	7.7	38.5	38.5	23.1
Gemma4-31B	0.0	13.0	13.0	4.3	17.6	17.6	29.4	29.4	30.8	61.5	61.5	61.5

LLM	Arbitrage (12)				Anomaly screening (8)			
	Raw	+Actions	+Units	+Retrieval	Raw	+Actions	+Units	+Retrieval
GPT-5.4-mini	0.0	0.0	0.0	0.0	62.5	37.5	37.5	25.0
GPT-5.6-terra	16.7	8.3	8.3	8.3	50.0	62.5	62.5	62.5
Gemini-3.1-Flash-Lite	0.0	0.0	8.3	0.0	75.0	62.5	75.0	75.0
Gemini-3.8-Flash	25.0	25.0	16.7	8.3	87.5	75.0	50.0	50.0
Qwen3-VL-8B-Instruct	0.0	0.0	0.0	0.0	12.5	25.0	12.5	12.5
Qwen3-30B-Instruct	0.0	0.0	0.0	0.0	25.0	12.5	25.0	25.0
Qwen3.5-9B	0.0	0.0	0.0	0.0	37.5	37.5	50.0	50.0
Qwen3.8-27B	8.3	16.7	8.3	8.3	62.5	62.5	62.5	62.5
Qwen3.8-27B-Finance	16.7	16.7	8.3	8.3	62.5	62.5	62.5	62.5
Gemma4-12B	0.0	0.0	0.0	0.0	50.0	25.0	25.0	25.0
Gemma4-31B	0.0	8.3	8.3	8.3	50.0	50.0	62.5	62.5

K.4 DIFFICULTY LEVELS

Table 19 reports question accuracy and field accuracy for all evidence conditions by difficulty level. Groups differ in task composition and required fields, so observed accuracy need not decrease monotonically from easy to hard.

Table 19: Strict question accuracy and field accuracy by difficulty level (%): easy 17, medium 60, hard 83 items.

(a) Question accuracy

LLM	Raw			+Actions			+Units			+Retrieval		
	Easy	Medium	Hard	Easy	Medium	Hard	Easy	Medium	Hard	Easy	Medium	Hard
GPT-5.4-mini	11.8	8.3	13.3	23.5	13.3	14.5	5.9	10.0	14.5	29.4	10.0	16.9
GPT-5.6-terra	41.2	23.3	25.3	41.2	30.0	31.3	35.3	35.0	34.9	29.4	35.0	32.5
Gemini-3.1-Flash-Lite	17.6	26.7	15.7	29.4	28.3	19.3	35.3	28.3	24.1	41.2	26.7	25.3
Gemini-3.8-Flash	58.8	48.3	55.4	58.8	48.3	55.4	64.7	43.3	54.2	64.7	45.0	50.6
Qwen3-VL-8B-Instruct	11.8	6.7	2.4	11.8	8.3	2.4	11.8	6.7	4.8	11.8	5.0	8.4
Qwen3-30B-Instruct	5.9	13.3	8.4	5.9	11.7	9.6	11.8	8.3	8.4	17.6	10.0	9.6
Qwen3.5-9B	11.8	6.7	13.3	17.6	10.0	9.6	5.9	8.3	10.8	17.6	11.7	18.1
Qwen3.8-27B	52.9	31.7	28.9	52.9	31.7	33.7	58.8	33.3	33.7	58.8	36.7	30.1
Qwen3.8-27B-Finance	52.9	33.3	33.7	52.9	38.3	36.1	64.7	38.3	37.3	58.8	38.3	38.6
Gemma4-12B	5.9	15.0	13.3	17.6	13.3	12.0	11.8	11.7	10.8	17.6	13.3	10.8
Gemma4-31B	23.5	33.3	15.7	41.2	33.3	26.5	47.1	35.0	28.9	41.2	36.7	28.9

(b) Field accuracy

LLM	Raw			+Actions			+Units			+Retrieval		
	Easy	Medium	Hard	Easy	Medium	Hard	Easy	Medium	Hard	Easy	Medium	Hard
GPT-5.4-mini	15.0	27.4	55.4	37.5	32.8	56.6	20.0	29.0	55.9	40.0	30.3	56.2
GPT-5.6-terra	55.0	44.0	65.7	55.0	50.2	68.6	45.0	54.8	72.1	40.0	54.4	69.2
Gemini-3.1-Flash-Lite	30.0	46.5	55.4	47.5	45.6	61.9	52.5	46.5	63.3	57.5	46.5	64.4
Gemini-3.8-Flash	67.5	62.7	83.4	72.5	66.0	84.6	77.5	65.6	84.6	77.5	64.3	80.5
Qwen3-VL-8B-Instruct	20.0	17.4	34.7	20.0	23.7	41.4	22.5	19.5	39.7	25.0	25.7	49.5
Qwen3-30B-Instruct	5.0	22.0	33.1	12.5	28.6	47.6	17.5	23.2	41.1	17.5	25.7	49.7
Qwen3.5-9B	17.5	25.3	49.1	25.0	24.9	50.8	20.0	24.9	47.3	35.0	30.7	55.4
Qwen3.8-27B	62.5	53.5	67.1	62.5	46.5	68.6	67.5	52.7	69.3	70.0	53.5	68.9
Qwen3.8-27B-Finance	65.0	49.4	69.3	70.0	54.4	72.7	77.5	56.0	72.1	72.5	55.2	70.5
Gemma4-12B	17.5	26.1	31.6	22.5	27.0	38.2	22.5	24.5	42.9	27.5	30.3	40.5
Gemma4-31B	40.0	51.5	55.1	55.0	47.7	65.0	65.0	50.6	66.5	57.5	55.2	65.9

K.5 EXECUTION PATHS

Table 20 separates the in-context path (110 items) and SQL path (50 items). These are different items, not a paired path-switching experiment. For the original four open-weight LLMs and two Gemini LLMs, +Units reuses +Retrieval's SQL-path outputs; these are not independent trials. Equal aggregate scores in other reports alone do not establish shared outputs.

Table 20: Strict question/field accuracy (%) by execution path: IC = in-context path (110 items); SQL path = 50 items.

LLM	Raw		+Actions		+Units		+Retrieval	
	IC	SQL	IC	SQL	IC	SQL	IC	SQL
GPT-5.4-mini	11.8/52.7	10.0/24.1	16.4/56.4	12.0/25.6	13.6/53.4	8.0/26.1	19.1/55.1	8.0/26.1
GPT-5.6-terra	29.1/65.7	20.0/37.9	35.5/70.1	24.0/38.4	39.1/72.3	26.0/45.3	36.4/69.3	26.0/45.3
Gemini-3.1-Flash-Lite	16.4/53.9	28.0/45.3	23.6/60.7	24.0/44.3	25.5/61.5	30.0/47.8	26.4/62.7	30.0/47.8
Gemini-3.8-Flash	55.5/81.2	48.0/63.5	55.5/82.3	48.0/68.5	55.5/83.1	42.0/66.0	53.6/79.1	42.0/66.0
Qwen3-VL-8B-Instruct	4.5/33.8	6.0/14.8	4.5/39.1	8.0/24.6	6.4/35.4	6.0/28.1	8.2/46.4	6.0/28.1
Qwen3-30B-Instruct	8.2/31.2	14.0/21.2	9.1/46.4	12.0/22.7	7.3/40.5	12.0/17.2	10.0/49.1	12.0/17.2
Qwen3.5-9B	9.1/46.2	14.0/25.1	8.2/46.1	16.0/32.0	4.5/41.6	20.0/36.0	13.6/51.6	20.0/36.0
Qwen3.8-27B	33.6/66.3	30.0/52.7	37.3/68.5	30.0/41.4	36.4/69.0	36.0/50.2	35.4/69.0	36.0/50.2
Qwen3.8-27B-Finance	37.3/67.7	32.0/50.7	43.6/72.4	28.0/51.2	42.7/72.6	36.0/52.2	42.7/70.7	36.0/52.2
Gemma4-12B	11.8/30.8	16.0/25.1	15.4/39.1	8.0/18.7	13.6/40.9	6.0/24.1	15.4/40.9	6.0/24.1
Gemma4-31B	22.7/54.2	24.0/51.2	33.6/63.5	24.0/47.8	34.5/65.1	30.0/52.2	34.5/65.7	30.0/52.2

K.6 Answer recoveries and regressions

Table 21 counts incorrect-to-correct recoveries and correct-to-incorrect regressions on the same 110 items on the in-context path. Across LLMs, +Actions → +Units yields 53 recoveries and 57 regressions; +Units → +Retrieval yields 62 and 44. These are LLM–item outcomes, not independent replicates or causal effects of individual evidence components. Appendix L examines selected successes and failures.

Table 21: Complete-answer recoveries (+) and regressions (–) among 110 items on the in-context path per LLM.

LLM	Raw → +Actions		+Actions → +Units		+Units → +Retrieval	
	+	–	+	–	+	–
GPT-5.4-mini	9	4	4	7	10	4
GPT-5.6-terra	12	5	9	5	3	6
Gemini-3.1-Flash-Lite	8	0	5	3	6	5
Gemini-3.8-Flash	7	7	7	7	4	6
Qwen3-VL-8B-Instruct	3	3	4	2	5	3
Qwen3-30B-Instruct	4	3	3	5	8	5
Qwen3.5-9B	3	4	3	7	11	1
Qwen3.8-27B	9	5	7	8	2	3
Qwen3.8-27B-Finance	12	5	7	8	6	6
Gemma4-12B	8	4	2	4	4	2
Gemma4-31B	14	2	2	1	3	3

K.7 Capability families

Table 22 retains the five disjoint capability families defined in Table 11, totaling 160 items. This classification differs from the overlapping Figure 3 topics.

Table 22: Strict question accuracy (%) under +Retrieval by capability family. Counts sum to 160.

LLM	Verdict (97)	Aggregation (22)	Relations (12)	Decoding (12)	Analytical boundary (17)
GPT-5.4-mini	17.5	13.6	8.3	8.3	17.6
GPT-5.6-terra	35.1	22.7	25.0	58.3	23.5
Gemini-3.1-Flash-Lite	25.8	36.4	16.7	41.7	23.5
Gemini-3.8-Flash	52.6	40.9	33.3	75.0	41.2
Qwen3-VL-8B-Instruct	9.3	9.1	8.3	0.0	0.0
Qwen3-30B-Instruct	10.3	4.5	8.3	16.7	17.6
Qwen3.5-9B	16.5	18.2	8.3	0.0	23.5
Qwen3.8-27B	34.0	36.4	16.7	66.7	35.3
Qwen3.8-27B-Finance	40.2	40.9	25.0	58.3	41.2
Gemma4-12B	13.4	22.7	8.3	8.3	0.0
Gemma4-31B	33.0	40.9	16.7	58.3	17.6

L ADDITIONAL QUALITATIVE ANALYSIS

These three case studies complement the T3 example in Table 6 with one in-context item each from T1, T2, and T4. The T1 and T2 cases use Qwen3.8-27B-Finance; the T4 case uses Qwen3.5-9B. All follow the same format: a paraphrased question and selected reference fields, followed by model-trajectory excerpts, submitted values, and the complete required-field score under each evidence condition. Cases were selected for contrasting outcomes; they do not estimate the frequency of these errors.

L.1 T1: ASSET FLOWS

Table 23: Asset-type recognition across evidence conditions. Qwen3.8-27B-Finance on one T1 in-context item. Background colors link the displayed fields across the reference, excerpts, and submitted values; ✓/× mark correct/incorrect values. Correct counts all 7 required fields. Quotes are verbatim; … marks omissions.

Question (paraphrased). Did the address's token holdings decrease by 21,658 base units over the specified block interval? Report the net flow, token behavior, cause of change, privilege capability, verdict, evidence grade, and reason code. Archive-node queries are unavailable.
Reference. flow 0; token type `not_erc20`; verdict `out_of_scope`; grade `out_of_scope`; reason `unsupported_asset`. The other 2 fields are correct in every condition.

Condition	Evidence, reasoning, and submitted values	Correct
RAW	*LLM:* "…The token is a standard ERC-20 (3-topic Transfers)…." ⇒ flow 0 ✓; token type `standard_erc20` ×; verdict `contradicted` ×; grade `contested` ×; reason `claim_vs_ledger_only` ×	3/7
+ACTIONS	*Evidence (protocol action):* "…frame root: 0x9217…ad7b called safeTransferFrom on 0x5f7e…49eb…" *Submitted JSON:* "…"flow_delta_raw": {"calc": "3*1 - 9*1"}, "token_behavior": "not_erc20",…" ⇒ flow -6 ×; token type `not_erc20` ✓; verdict `out_of_scope` ✓; grade `out_of_scope` ✓; reason `unsupported_asset` ✓	6/7
+UNITS	*LLM:* "…This is an ERC-721 token, not an ERC-20.…" *LLM:* "…ERC-721 tokens are out of scope.…" ⇒ flow 0 ✓; token type `not_erc20` ✓; verdict `out_of_scope` ✓; grade `out_of_scope` ✓; reason `unsupported_asset` ✓	7/7
+RETRIEVAL	*LLM:* "…The evidence cut is scoped to ETH transfers only.…" *LLM:* "…For token_behavior: the token is a CA (lb1), no evidence of TransferShares or nonstandard behavior…" ⇒ flow 0 ✓; token type `standard_erc20` ×; verdict `contradicted` ×; grade `contested` ×; reason `claim_vs_ledger_only` ×	3/7

Under +ACTIONS, the cited protocol action helps the LLM recognize the unsupported asset type, but the submitted calculation produces an incorrect net flow of -6. Under +UNITS, it identifies an ERC-721 asset and returns all seven required fields correctly. Under +RETRIEVAL, the model trajectory describes evidence about ETH transfers and returns to `standard_erc20`, although the net-flow value remains correct. This case separates numerical correctness from asset-type and analytical-boundary judgments; recovery under +UNITS is not preserved under +RETRIEVAL.

L.2 T2: EXCHANGE ATTRIBUTION

Table 24: Exchange-control judgments with stable attribution fields. Qwen3.8-27B-Finance on one T2 in-context item. Background colors link the displayed fields across the reference, excerpts, and submitted values; ✓/× mark correct/incorrect values. Correct counts all 6 required fields. Quotes are verbatim; … marks omissions.

Question (paraphrased). Was the address controlled by a centralized exchange during the observation window and the following seven days? Check its Dune label category and exchange-list membership, and report the verdict within the observation window, the overall verdict, evidence grade, and type of claim extending beyond the window.
Reference. category `Bridge`; in-boundary verdict `contradicted`; verdict `contradicted`; grade `contested`. The other 2 fields are correct in every condition.

Condition	Evidence, reasoning, and submitted values	Correct
RAW	*LLM:* "…Label check (lb1): owner_category = "Bridge", in_cex_addresses = false.…Per the table, no L family + a PROVEN label with a different service role → contradicted/contested.…" ⇒ category `Bridge`✓; in-boundary verdict `contradicted`✓; verdict `contradicted`✓; grade `contested`✓	6/6
+ACTIONS	*LLM:* "…No PROVEN evidence supports exchange control; a PROVEN label points to a different service role → contradicted/contested.…" ⇒ category `Bridge`✓; in-boundary verdict `contradicted`✓; verdict `contradicted`✓; grade `contested`✓	6/6
+UNITS	*LLM:* "…This is "everything else" → indeterminate/contested.…Since the in-scope element is not contradicted, the whole claim is out_of_scope/out_of_scope." ⇒ category `Bridge`✓; in-boundary verdict `indeterminate`×; verdict `out_of_scope`×; grade `out_of_scope`×	3/6
+RETRIEVAL	*LLM:* "…And there is a PROVEN label pointing to a different service role (Bridge). Per the table: no L + PROVEN label with different service role → contradicted/contested.…" ⇒ category `Bridge`✓; in-boundary verdict `contradicted`✓; verdict `contradicted`✓; grade `contested`✓	6/6

The owner category, exchange-list membership, and identification of future control remain correct across all four evidence conditions. Under +UNITS, however, the LLM changes the in-boundary verdict to indeterminate and the overall verdict to out of scope. Its quoted reasoning shows a different application of the decision criteria despite the stable attribution fields. +RETRIEVAL restores all six fields. The case exposes a change in how attribution context is used, without establishing that the complete inputs are identical.

L.3 T4: AMM WASH-TRADING PATTERNS

Table 25: Incomplete recovery on a wash-trading item. Qwen3.5-9B under the four evidence conditions. Background colors link the displayed fields across the reference, excerpts, and submitted values; ✓/× mark correct/incorrect values. Correct counts all six required fields. Quotes are verbatim; … marks omissions. ∅ denotes a missing value; the unresolved calculation expression is abbreviated.

Question (paraphrased). Does a same-sender run of ten trades in a Uniswap V2 pool, beginning at block 25405824, satisfy the wash-trading criteria? Report whether each of the five conditions passes and give the final conclusion.
Reference. span 9; min ratio 0.0; MEV checks [0, 0, 0]; verdict `supported`; grade `anchor_gold`. The single-initiator field is correct in every condition.

Condition	Evidence, reasoning, and submitted values	Correct
RAW	*LLM:* "…The question states there are ten trades in a same-sender run. However, the evidence only shows one transaction.…" ⇒ span 1 ×; min ratio ∅ ×; MEV checks ∅ ×; verdict `out_of_scope` ×; grade `out_of_scope` ×	1/6
+ACTIONS	*Submitted JSON:* "…"span": "25405829 - 25405824 = 5",…" ⇒ span 5 ×; min ratio unresolved expression ×; MEV checks `false` ×; verdict `supported` ✓; grade `anchor_gold` ✓	3/6
+UNITS	*Evidence (analytical unit):* "…made 10 consecutive swaps (10 txs), blocks 25405824~25405833 (span 9), pool order #1743~#1752" *Submitted JSON:* "…"span":"9","min_ratio":"1/20",…" ⇒ span 9 ✓; min ratio `1/20` ×; MEV checks `false` ×; verdict `supported` ✓; grade `anchor_gold` ✓	4/6
+RETRIEVAL	*LLM:* "…blocks 25405824~25405833 (span 9)…" *LLM:* "…Ratio: I need to calculate the min ratio. From the group_ratios tool, I would get the exact fractions. But since I can't get that, let me estimate.…" ⇒ span 9 ✓; min ratio 0.05 ×; MEV checks `[]` ×; verdict `supported` ✓; grade `anchor_gold` ✓	4/6

Under RAW, the LLM describes evidence for one transaction and returns an incorrect span and verdict. +ACTIONS recovers the verdict but still uses a five-block span and leaves the ratio as an unresolved calculation expression. Under +UNITS, the cited analytical unit identifies all ten swaps and their nine-block span. However, the LLM returns $1/20$ for the minimum ratio rather than the reference value of zero, and supplies a scalar instead of the three required MEV-check values. +RETRIEVAL retains the correct span but returns an estimated ratio of 0.05 and an empty MEV-check list. Thus, the analytical boundary becomes explicit while calculation and output-completeness errors persist; no evidence condition yields a fully correct answer.

M TOKEN USAGE AND HARDWARE

Table 26 reports mean input and output tokens per item for the four closed models, summed across all API requests for that item. Input counts include cached tokens. Under +UNITS, usage covers the 110 independently run in-context items; the 50 SQL-path responses are reused from +RETRIEVAL. The other conditions cover all 160 items, so their averages include both execution paths.

Table 26: Mean API tokens per item, summed across all requests. Input includes cached tokens. †: 110 in-context items; other columns: 160 items.

LLM	Tokens	RAW	+ACTIONS	+UNITS †	+RETRIEVAL
Gemini-3.8-Flash	Input	443,001	411,363	533,924	362,615
	Output	854	698	636	689
GPT-5.6-terra	Input	180,425	204,044	245,918	183,097
	Output	537	398	323	376
Gemini-3.1-Flash-Lite	Input	675,111	625,334	795,815	525,161
	Output	1,388	1,059	923	994
GPT-5.4-mini	Input	306,847	351,272	460,647	289,890
	Output	679	638	651	713

Among the seven open-weight models, Qwen3-VL-8B-Instruct and Qwen3-30B-Instruct were run on NVIDIA RTX A6000 GPUs. Gemma4-12B, Gemma4-31B, Qwen3.5-9B, Qwen3.8-27B, and Qwen3.8-27B-Finance were run on NVIDIA RTX PRO 6000 GPUs.