

Bridging Databases and Documents: Data-Algorithm Co-Design for Hybrid Question Answering

Ruofan Wu^{1*} Boyi Liu^{2†} Fan Shu¹ Yite Wang²
Zhewei Yao² Yuxiong He² Feng Yan¹

¹University of Houston

²Snowflake AI Research

Abstract

Large Language Models (LLMs) increasingly rely on multi-tool orchestration to resolve complex enterprise queries. However, answering hybrid questions requires tightly coupled reasoning steps between structured data retrieval and unstructured semantic search, where a single execution error triggers an irreversible cascade. Training models to navigate these strict cross-modal pipelines via standard Reinforcement Learning (RL) is notoriously difficult. Because successful multi-hop trajectories are rare, unguided exploration leads to severe data waste and sparse rewards. While recent algorithmic advancements improve RL optimization dynamics, they still fundamentally rely on the highly inefficient process of discovering successful trajectories from scratch. In this paper, we advocate for a data-algorithm co-design paradigm to overcome this exploration bottleneck. First, we introduce HyDRA-Bench¹, a novel environment simulating realistic hybrid reasoning where agents must seamlessly interleave SQL and search. Second, we propose TRACE-GRPO, an adaptive RL framework that tightly integrates the data generation pipeline with policy optimization. By converting intermediate reasoning traces generated during dataset synthesis into executable, action-level hints that serve as proxy goals, TRACE-GRPO transforms sparse unguided rollouts into a dense, highly efficient curriculum. This approach not only stabilizes optimization by preventing gradient vanishing but also drives “in-context to in-weights” skill elicitation. Empirical results demonstrate that TRACE-GRPO drastically accelerates convergence, enabling an 8B open-weights model to surpass Claude-Opus-4.5 on hybrid reasoning, while exhibiting robust generalization to broader multi-hop text environments like MuSiQue.

1 Introduction

Large Language Models (LLMs) are increasingly deployed for complex question answering, often relying on external information sources to ground their responses and mitigate hallucinations. For instance, Retrieval-Augmented Generation (RAG) (Lewis et al., 2020; Gao et al., 2023; Jeong et al., 2024) extracts factual support from unstructured text corpora, while Text-to-SQL models (Pourreza & Rafiei, 2024; Wang et al., 2025; Yao et al., 2025) navigate structured databases to compute precise empirical results. However, real-world enterprise applications frequently demand the synthesis of knowledge from both domains simultaneously. As illustrated in Figure 1 (Part 1), answering a query like “Who is the CEO of the company with the highest revenue drop?” requires a multi-step process: first identifying the target entity via database operations, and subsequently retrieving complementary information from text. This demands an agent capable of dynamically interleaving database operations and semantic retrieval within a unified reasoning process.

*Work done during the internship at Snowflake AI Research.

†Project lead.

¹<https://huggingface.co/datasets/Snowflake/HyDRA-Bench>

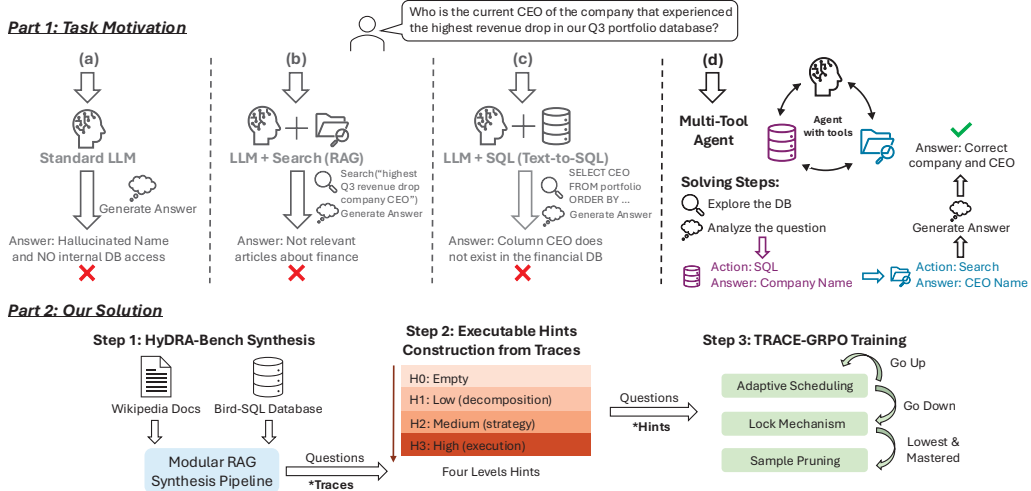


Figure 1: Overview of the hybrid reasoning task and our proposed framework. (Top) Pure LLMs (a) hallucinate, while single-tool RAG (b) and Text-to-SQL (c) pipelines fail due to incomplete context. A multi-tool agent (d) resolves these complex queries by interleaving SQL and semantic search. (Bottom) Our co-design solution involves three stages. Step 1 synthesizes the HyDRA-Bench dataset. Step 2 extracts intermediate reasoning traces from this synthesis to construct a four-tier hint hierarchy ($H_0 \sim H_3$). Step 3 trains the agent via TRACE-GRPO, dynamically annealing hint levels through adaptive scheduling, a unidirectional lock mechanism, and mastery-based pruning.

Despite this necessity, building effective multi-tool agents for hybrid reasoning remains highly challenging. At a fundamental level, these tasks require tightly coupled coordination between deterministic SQL execution and fuzzy semantic retrieval. A single mistake at any step can cascade through the cross-modal dependency chain and invalidate the entire reasoning trajectory. As a result, existing approaches based on prompting (Yao et al., 2022) or agentic frameworks (Wu et al., 2025) perform poorly: they are not explicitly optimized for interleaved multi-tool orchestration and lack the structural competence required for complex database reasoning (He et al., 2025). This includes challenges such as navigating large schemas or recovering from silent execution failures (Piao et al., 2026), leading to low end-to-end accuracy on hybrid queries. A natural solution is to train such agents directly. However, this introduces a second layer of difficulty. On the data side, there is a scarcity of high-quality supervision that captures realistic, interleaved reasoning between SQL execution and semantic retrieval (Chen et al., 2020b). Such trajectories require precise alignment between structured queries and unstructured evidence, making them difficult to collect at scale. On the optimization side, even with training data, standard Reinforcement Learning (RL) remains highly inefficient. Because successful multi-hop trajectories that correctly interleave SQL and search are exceedingly rare, unguided exploration suffers from severe data inefficiency and sparse rewards (Zhang et al., 2025a; Liu et al., 2025a), where any intermediate error yields no useful learning signal. As a result, models struggle to discover and internalize the correct cross-modal reasoning patterns.

To address these challenges, we propose a data–algorithm co-design framework that tightly couples data generation with policy optimization (see Figure 1 (Part 2)). We introduce HyDRA-Bench (Hybrid Database and Retrieval Agent Benchmark), a benchmark designed for hybrid reasoning between SQL execution and semantic retrieval in enterprise settings. Importantly, HyDRA-Bench provides structured reasoning traces generated during data synthesis, which we convert into executable, action-level hints for policy learning. Building on this design, we propose TRACE-GRPO (Trace-Reused Adaptive Curriculum Exploration GRPO), an adaptive reinforcement learning framework that converts synthesis-time traces into action-level hints to guide policy optimization. Unlike prior hint-based methods (Zhang

et al., 2025b; Qu et al., 2026), which rely on externally provided reasoning trajectories or privileged solution prefixes, our approach requires no additional human annotation and remains fully aligned with the data generation process. TRACE-GRPO transforms sparse, unguided exploration into efficient, guided learning by leveraging trace-derived hints and an adaptive curriculum that adjusts the level of guidance based on training progress. This unified framework substantially improves training efficiency. TRACE-GRPO accelerates convergence and enables smaller open-weights models to narrow the gap with much larger proprietary models. Moreover, because it relies only on trace-augmented data, our approach generalizes naturally to broader multi-hop reasoning settings. We demonstrate this on MuSiQue (Trivedi et al., 2022), which features longer multi-hop reasoning chains, where our method continues to achieve strong performance gains.

Our contributions are summarized as follows:

- **A data–algorithm co-design paradigm for hybrid reasoning.** We introduce a unified framework that tightly couples data generation with policy optimization, where structured reasoning traces produced during data synthesis are converted into executable hints for guiding policy learning.
- **HyDRA-Bench: A benchmark for interleaved SQL and retrieval reasoning.** We present a new benchmark that captures realistic hybrid reasoning scenarios, requiring agents to dynamically interleave executable SQL operations with semantic retrieval in enterprise settings.
- **TRACE-GRPO: Trace-driven reinforcement learning.** We propose an adaptive RL framework that leverages trace-derived hints through a dynamic curriculum, where the level of guidance is adjusted based on agent performance, progressively transitioning from guided exploration to autonomous reasoning.
- **Improved efficiency and generalization.** Our approach significantly improves training efficiency, enabling open-weights models to achieve competitive performance on hybrid reasoning and surpass Claude-Opus-4.5 at the 8B scale. It further generalizes to longer multi-hop reasoning tasks, demonstrating robustness to increased reasoning depth.

2 Related Work

Prior work related to our paper can be grouped into three directions. Hybrid table-text question answering studies multi-hop reasoning over structured and unstructured sources (Chen et al., 2020b;a; Zhu et al., 2021; Zhao et al., 2022), but typically treats tables as static contexts rather than interactive databases with executable queries. General tool-use agents extend LLMs with external actions (Patil et al., 2024; Qin et al., 2023; Yao et al., 2022; Team et al., 2025; Zeng et al., 2025; Li et al., 2025), yet they are mainly developed for environments with explicit feedback and struggle in hybrid settings with weak or silent failures. Hint-based reinforcement learning improves exploration in multi-step reasoning (Huang et al., 2025; Fu et al., 2025; Liu et al., 2025b; Yan et al., 2025; Zhang et al., 2025c;b; Ma et al., 2025; Qu et al., 2026), but existing methods largely rely on externally provided hints or trajectories and are not designed for interleaved SQL-retrieval reasoning. A comprehensive discussion of these directions is provided in Appendix 7.1.

3 HyDRA-Bench Synthesis: Trace-Aware Data Generation for Hybrid Reasoning

This section presents the construction of HyDRA-Bench, a dataset for hybrid reasoning between SQL execution and semantic retrieval. Unlike conventional datasets that focus on question-answer pairs, our pipeline synthesizes multi-hop instances that require tightly coupled interaction between the two tools. During this process, it also produces structured intermediate information, such as hop-level questions, tool usage, and intermediate answers, which together form reasoning traces that are later converted into action-level hints for policy learning, enabling a unified data–algorithm co-design.

3.1 Data Sources

We introduce **HyDRA-Bench**, a dataset for hybrid reasoning that requires interleaved interaction between structured databases and unstructured text. Each instance is a multi-step task where neither SQL execution nor retrieval alone is sufficient. We construct this environment by combining databases and queries from BIRD-SQL (Li et al., 2024) with a retrieval corpus built from the Wiki2019 dump (Petroni et al., 2021).

3.2 Trace-Aware Synthesis Pipeline

To synthesize high-quality hybrid reasoning data, we design a modularized pipeline that simulates interaction between two complementary roles: the **Analyst** and the **Searcher**. The Analyst operates over structured databases by issuing executable SQL queries to extract precise relational facts, while the Searcher retrieves contextual or entity-level information from unstructured text corpora. These roles are coordinated through a multi-step synthesis process that enforces cross-tool dependencies via two reasoning patterns: Analyst-to-Search (A2S) and Search-to-Analyst (S2A). This design ensures that each synthesized instance requires interleaved tool usage, while naturally producing structured intermediate states that together form reasoning traces.

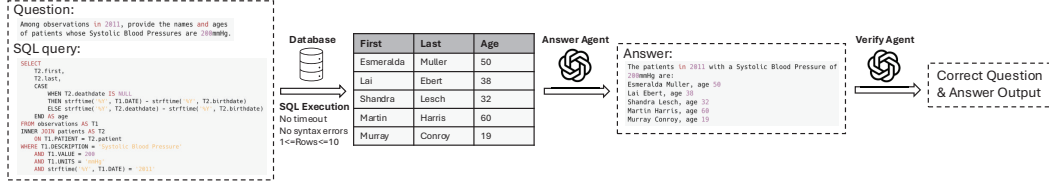


Figure 2: The pipeline filters raw queries based on execution stability and a result cardinality constraint ($1 \leq \text{Rows} \leq 10$). Validated structured data is then synthesized into natural language by the AnswerAgent and rigorously checked by the VerifyAgent to produce factually grounded seeds for multi-hop expansion.

Phase I: Analyst Grounding via SQL Execution. The first phase establishes grounded states by executing SQL queries over structured databases. As shown in Figure 2, we execute queries from BIRD-SQL on their corresponding databases and discard those with execution errors or invalid outputs, retaining only results with 1–10 rows to encourage entity-level reasoning. The validated outputs are then converted into natural language by an AnswerAgent and verified for faithfulness using a VerifyAgent. These grounded intermediate states form the initial components of the reasoning trace for subsequent multi-hop synthesis.

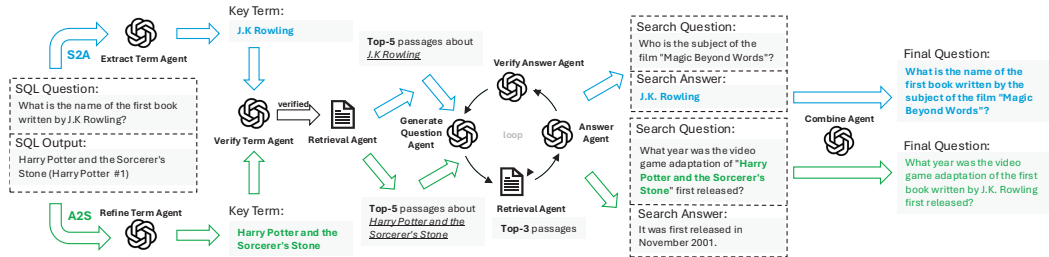


Figure 3: The pipeline branches into two lanes: A2S (bottom) and S2A (top). Both utilize an iterative verification loop involving term extraction, retrieval, and combination, ensuring that the final multi-hop questions solving needs both SQL execution and Wikipedia search.

Phase II: Multi-Hop Trace Construction via Tool Interaction. To enforce cross-tool dependencies, we construct multi-hop reasoning questions through two complementary patterns: Analyst-to-Search (A2S) and Search-to-Analyst (S2A), as illustrated in Figure 3. In A2S, SQL execution produces the entity that anchors the subsequent retrieval step, while in S2A, retrieved evidence determines the entity or constraint for the following SQL query. Both patterns follow a unified synthesis process with iterative verification to ensure consistency across steps, producing structured sequences of executable actions that form multi-hop reasoning traces.

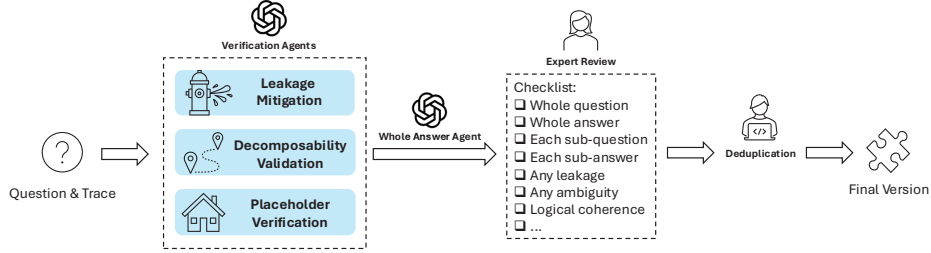


Figure 4: Verification pipeline for synthesized hybrid reasoning data. Each sample undergoes automated checks for leakage, decomposability, and placeholder consistency, followed by gold answer generation, targeted manual audit, and semantic deduplication to ensure data quality and logical integrity.

Phase III: Trace Verification and Quality Control. To ensure data quality and consistency, we apply a verification stage over both the final question-answer pairs and the intermediate states produced (traces) during synthesis (Figure 4). We perform automated checks for leakage, decomposability, and placeholder consistency to ensure logical validity and composability across hops, while masking intermediate bridge entities to prevent shortcut reasoning. We further generate gold-standard answers, conduct targeted manual audits for factual correctness, and apply semantic deduplication to remove redundancy. This process yields high-quality data while preserving intermediate information for subsequent hint construction.

3.3 Trace-to-Hint Construction

From the synthesis pipeline, we obtain not only multi-hop question-answer pairs, but also structured intermediate artifacts such as hop-level questions, tool usage, and intermediate answers, which together form synthesis traces. We convert these traces into a hierarchy of action-level hints for policy learning by exposing different levels of abstraction over the underlying reasoning process. Critically, this conversion happens entirely offline during dataset synthesis. For each instance, the four hint tiers are materialized once alongside the question and gold answer and stored in the dataset.

Hint Tiers. We organize the hints into four levels of increasing information:

- **Empty:** The agent proceeds independently without any external guidance.
- **Low (Decomposition-focused):** This tier provides a high-level logical split of the problem into two dependent sub-tasks, encouraging the agent to recognize the multi-hop structure.
- **Medium (Strategy-focused):** This tier outlines the specific tool usage strategy and provides the ground-truth intermediate answer of the first hop. This directs the agent on what to solve first and what information to carry over.
- **High (Execution-focused):** This tier offers granular, step-by-step instructions. In addition to the strategy, it explicitly provides the required queries for the Analyst hop, removing technical ambiguity in execution.

3.4 Dataset Format and Statistics

Each sample in HyDRA-Bench is a multi-hop hybrid reasoning instance consisting of a natural language question, the associated structured database and unstructured retrieval context, structured intermediate artifacts that form the reasoning trace, and a four-level hint hierarchy derived from that trace. Crucially, for both A2S and S2A flows, the gold final

Flow	Train	Dev	Total
A2S	5,036	849	5,885
S2A	10,480	1,678	12,158
Total	15,516	2,527	18,043

Table 1: Dataset split by reasoning flow.

answer is a **natural-language explanation** (mean length ~ 220 characters) generated and verified during synthesis, rather than a raw SQL cell, entity span, or short label. This format matches our evaluation protocol, where an LLM judge assesses semantic correctness against these reference explanations. The overall dataset split is summarized in Table 1, while fine-grained database and SQL statistics alongside a concrete example are provided in Appendix 7.2 and Appendix 7.5.

4 Data–Algorithm Co-design via Hint-Guided RL

In this section, we instantiate our data–algorithm co-design principle. Instead of treating dataset synthesis (see Section 3) and policy optimization as separate stages, we reuse the intermediate reasoning traces generated during synthesis as progressively weaker forms of supervision for reinforcement learning. This frames a goal-conditioned learning problem in which the agent is initially trained with explicit hints and gradually transitioned toward the fully autonomous target.

4.1 Theoretical Framework: Goal Conditioning and Task Transfer

Let $\mathcal{G} = \{g_0, g_1, \dots, g_L\}$ denote an ordered set of goals, which correspond directly to the hint tiers defined in Section 3 (i.e., $g_k \equiv H_k$, with $L = 3$). Here, g_0 corresponds to the autonomous target objective, and larger indices indicate stronger external guidance (i.e., denser synthesis traces). For an input instance x , the policy is conditioned on both the instance and the current goal $\pi_\theta(z \mid x, g)$, where z denotes a response trajectory.

Why Hints Help. Standard GRPO-style optimization (Shao et al., 2024) can fail on difficult tool-use tasks because the reward signal is extremely sparse: when successful trajectories are rare, most sampled rollouts receive the same return (typically zero). In that regime, intra-group advantage normalization produces little or no learning signal, and the policy gradient collapses (Yu et al., 2025; Liu et al., 2025c; Zhang & Zuo, 2025; Liu et al., 2026). The primary function of hints is to systematically shift the policy onto a proxy task that lies closer to the feasible region, ensuring that successful rollouts become more frequent and the group-relative training signal remains dense and informative.

A Proxy-to-Target View. Let $P^*(z \mid x)$ denote the oracle solution distribution for the ground-truth reasoning trajectories (e.g., successful SQL-Search chains) that satisfy the query x . Our co-design assumes that P^* is an invariant structural property of the task: while the hint level g alters the conditional information, the optimal reasoning path remains consistent. We thus define the goal-conditioned loss as the expected divergence from this oracle:

$$\mathcal{L}_g(\theta) = \mathbb{E}_x [\text{KL}(P^*(z \mid x) \parallel \pi_\theta(z \mid x, g))].$$

Under this formulation, learning with a guided goal g_k optimizes a proxy objective that facilitates the discovery of P^* (Zhao & Li, 2025). The relationship between the guided proxy and the autonomous target can be characterized by the following alignment bound:

$$\mathcal{L}_0(\theta) \leq \mathcal{L}_k(\theta) + \epsilon_{\text{align}}(\theta), \quad \epsilon_{\text{align}}(\theta) = \mathbb{E}_x [\text{KL}(\pi_\theta(z \mid x, g_k) \parallel \pi_\theta(z \mid x, g_0))].$$

This decomposition treats ϵ_{align} as the representation gap induced by the external scaffolding. In the GRPO framework, while P^* is not explicitly provided during RL, the binary reward $R(z)$ acts as a functional filter for P^* . By maximizing expected rewards under g_k , the model effectively minimizes $\mathcal{L}_k$. Because all goals share the same transformer backbone ϕ_θ , minimizing the proxy loss $\mathcal{L}_k$ while regularizing the alignment gap forces the model to internalize the structural invariants of P^* into its weights, enabling autonomous mastery at g_0 .

Curriculum as Constrained Optimization. The critical operational question is how to schedule the amount of guidance. For each sample x_i at training step t , let $c(g)$ be the semantic information cost of goal g , with larger indices corresponding to higher costs. We seek to assign the least informative goal that still remains reachable under the current policy:

$$g_{i,t+1} = \underset{g \in \mathcal{G}_{\text{valid}}}{\operatorname{argmin}} c(g) \quad \text{s.t.} \quad V^{\pi_t}(x_i, g) \geq \tau_s,$$

where $V^{\pi_t}(x_i, g) = \mathbb{E}_{z \sim \pi_t(\cdot | x_i, g)}[R(z)]$ is the expected return, and τ_s is the minimum success threshold required to prevent advantage vanishing and keep the training signal reliable.

4.2 Dynamic Curriculum Learning in TRACE-GRPO

To practically solve the constrained optimization problem defined above, we introduce an adaptive training framework (Algorithm 1). This dynamic curriculum monitors the real-time competence of the policy, continually adjusting the hint level to ensure training strictly occupies the active learning frontier between solved and unsolved instances.

Adaptive Hint Scheduling. For each sample x_i at epoch t , we estimate the empirical success rate over n rollouts: $\mu_{i,t} = (\sum_{j=1}^n \mathbb{I}(\text{success}_{i,j}))/n$, which serves as an unbiased Monte Carlo proxy for $V^{\pi_t}(x_i, g_{i,t})$. If the policy falls below the desired success threshold, the sample receives stronger guidance to restore reachability; otherwise, the hint is weakened to minimize $c(g)$. To govern this, we define an adaptive update rule alongside a dynamic upper bound $l_{i,t}$ (initialized as g_L):

$$g_{i,t+1} = \begin{cases} \min(g_{i,t} + 1, l_{i,t}), & \text{if } \mu_{i,t} < \tau_s \\ \max(g_{i,t} - 1, g_0), & \text{if } \mu_{i,t} \geq \tau_s \end{cases}$$

Hint Lock. To prevent oscillatory scheduling and enforce monotonic skill consolidation, the feasible set $\mathcal{G}_{\text{valid}}$ must be restricted. Once a sample demonstrates competence at a weaker level of support, we strictly bound its ability to regress to heavier hints. We formalize this via a unidirectional hint lock that updates the upper bound $l_{i,t+1} = \min(l_{i,t}, g_{i,t+1})$ exclusively downward. By enforcing this lock, the scheduling policy remains stable. The model is mathematically prevented from permanently resetting to an easier conditioning regime after it has already absorbed the structural invariants necessary for a more autonomous state.

Mastery-Based Sample Pruning. Finally, once a sample can be solved autonomously with high confidence, it no longer provides useful variance for optimization. We define a stricter mastery threshold $\tau_r > \tau_s$ and formalize a pruning indicator $m_{i,t} = \mathbb{I}[g_{i,t} = g_0, \mu_{i,t} > \tau_r]$. Samples evaluating to $m_{i,t} = 1$ are permanently removed from the active training set $\mathcal{D}$. This mastery-based pruning concentrates the computational budget and gradient updates strictly on the remaining unsolved instances, maximizing throughput while keeping the learning signal focused on the hardest reasoning dependencies.

5 Evaluation Results

5.1 Experimental Setup

We evaluate our method on HyDRA-Bench dev set and additionally report results on MuSiQue to assess generalization to longer multi-hop reasoning chains. Agents interact with HyDRA-Bench through a two-tool runtime environment (Appendix 7.3). For RL fine-tuning, we train Qwen3-4B and Qwen3-8B backbones under identical settings and compare TRACE-GRPO against standard GRPO. Both methods use $n = 8$ rollouts per example in each epoch. During evaluation, all models are tested under the strict Zero-hint (H_0) setting, where no external guidance or reasoning traces are provided, ensuring the performance reflects fully autonomous reasoning. We perform $k = 8$ independent runs and report Acc_{avg} , the average accuracy across all runs, and $\text{Best}@k$, the percentage of questions answered correctly in at least one run. To ensure rigorous assessment of complex hybrid reasoning, we employ GPT-o4-mini as an LLM-as-a-judge to evaluate final answer correctness against the gold natural-language explanations. Because final answers require semantic equivalence rather than exact string match, we further validate this judge against human labels on a stratified sample of evaluation trajectories (Appendix 7.4). All experiments are conducted on a cluster with 8 NVIDIA B200 GPUs.

5.2 Model Performance and RL Fine-tuning

Table 2 reports the accuracy of both off-the-shelf models and RL fine-tuned models. We compare TRACE-GRPO with standard GRPO to isolate the impact of trace-derived hints. For TRACE-GRPO, training starts from the empty hint level (H_0), with a success threshold $\tau_s = 0.5$ and removal threshold $\tau_r = 0.875$, enabling a transition from unguided exploration to stable autonomous reasoning.

Existing Model Performance. Among proprietary models, Claude-Opus-4.5 demonstrates the strongest performance. In contrast, open-weight models (Qwen3 series) lag significantly behind even at larger scales (32B), highlighting the inherent difficulty of hybrid reasoning and motivating the need for more effective training strategies.

Reinforcement Learning Results. We evaluate TRACE-GRPO on Qwen3-4B and Qwen3-8B using the same curriculum hyperparameters ($\tau_s = 0.5$, $\tau_r = 0.875$). On both backbones, TRACE-GRPO consistently outperforms GRPO across all metrics in Table 2, confirming that trace-derived hints improve policy optimization beyond unguided exploration.

On Qwen3-4B, TRACE-GRPO raises Acc_{avg} from 43.91% (GRPO) to 46.33%, a substantial gain over the 13.96% off-the-shelf baseline despite the model’s limited capacity. We use this setting for mechanistic analysis in Figure 5: the hint distribution shifts from high-density guidance (H2/H3) toward H_0 , and TRACE-GRPO converges faster than GRPO, peaking around 1,600 steps before mild overfitting.

To assess scalability, we further fine-tune Qwen3-8B under the same protocol. Increasing model size alone provides little benefit (14.02% vs. 13.96% Acc_{avg} for 4B), underscoring that hybrid reasoning requires targeted training rather than parameter count alone. After RL, however, the larger backbone yields a stronger policy: GRPO reaches 48.51% Acc_{avg} , already surpassing 4B TRACE-GRPO. With trace-guided curriculum, Qwen3-8B TRACE-GRPO achieves 53.91% Acc_{avg} and 72.89% $\text{Best}@8$, exceeding Claude-Opus-4.5 (50.56% / 62.84%) and establishing the best performance among all evaluated models. The absolute improvement of TRACE-GRPO over GRPO is also larger at 8B (+5.40 vs. +2.42 Acc_{avg}), suggesting that stronger base models can more effectively internalize curriculum-provided structural invariants once exploration becomes tractable.

We further inspect evaluation trajectories from the 8B runs to verify that gains reflect genuine hybrid tool use rather than shortcut behaviors. No TRACE-GRPO trajectory ends after a single tool call, and the call-count distribution (3 / 4 / 5 / 6 / 7+) is 6.5% / 41.6% / 32.6% / 13.4% / 5.7%, indicating consistent multi-step execution. The distribution is also non-degenerate: TRACE-GRPO peaks at 4 calls (spanning 3–8+) while GRPO peaks at 6

(spanning 4–8+), and TRACE-GRPO averages 1.42 fewer calls per question (4.72 vs. 6.14) despite higher accuracy, suggesting shorter and more decisive plans rather than a fixed call budget. Finally, 94.1% of TRACE-GRPO trajectories invoke both `run.sql` and `cortex.search`, compared with 90.7% for GRPO, confirming that the agent composes both tools rather than collapsing to one.

Together, these results validate both the effectiveness and scalability of TRACE-GRPO: trace-derived hints alleviate sparse rewards at small scale, and the same co-design principle enables an 8B open-weights agent to surpass top-tier proprietary systems on HyDRA-Bench.

Model Group / Name	Best@1	Best@2	Best@4	Best@8	Acc _{avg}
<i>Proprietary Models</i>					
GPT-4o	0.3039	0.3989	0.4824	0.5457	0.3045
GPT-4.1	0.3779	0.4638	0.5390	0.6015	0.3734
GPT-5	0.3870	0.4614	0.5275	0.5786	0.3842
Claude-Sonnet-4.5	0.3478	0.4143	0.4689	0.5061	0.3510
Claude-Opus-4.5	0.5097	0.5615	0.5991	0.6284	0.5056
<i>Open-Weights Models</i>					
Qwen3-4B	0.1338	0.2026	0.2794	0.3399	0.1396
Qwen3-8B	0.1393	0.2165	0.2913	0.3704	0.1402
Qwen3-14B	0.2070	0.2992	0.3878	0.4674	0.2056
Qwen3-32B	0.2742	0.3609	0.4523	0.5267	0.2703
<i>Qwen3-4B RL Fine-tuning</i>					
GRPO	0.4341	0.5295	0.6003	0.6537	0.4391
TRACE-GRPO	0.4634	0.5588	0.6245	0.6814	0.4633
<i>Qwen3-8B RL Fine-tuning</i>					
GRPO	0.4812	0.5710	0.6383	0.6874	0.4851
TRACE-GRPO	0.5374	0.6189	0.6830	0.7289	0.5391

Table 2: Main results on HyDRA-Bench. We compare existing proprietary and open-weights models against Qwen3-4B and Qwen3-8B fine-tuned with standard GRPO and our proposed TRACE-GRPO. All RL models are evaluated under the zero-hint (H_0) setting.

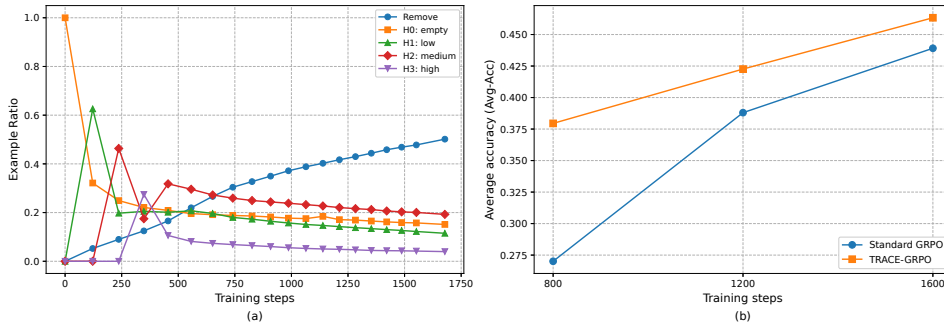


Figure 5: Mechanistic and Performance Analysis of TRACE-GRPO on Qwen3-4B. (a) Adaptive Curriculum Evolution: The shift in Hint Distribution Ratio illustrates the “in-context to in-weights” transition. This reveals how the agent gradually transitions from relying on high-density hints to successfully resolving complex queries independently, effectively internalizing the guidance traces into its own parameters. (b) Convergence Acceleration: Comparison of Average Accuracy on HyDRA-Bench between standard GRPO and TRACE-GRPO. TRACE-GRPO achieves superior performance and faster convergence, peaking around 1,600 steps. Beyond this point, we observe performance degradation for both methods due to overfitting.

5.3 Generalization to Longer Multi-Hop Reasoning

To evaluate the scalability and robustness of our framework, we conduct a generalization study on MuSiQue (Trivedi et al., 2022) by fine-tuning the Qwen3-4B model on its 3-hop and 4-hop training subsets for 400 steps. This study specifically focuses on these complex examples to test the agent’s performance as reasoning depth increases. Utilizing a three-level hierarchical hint structure, we set a relatively conservative success threshold $\tau_s = 0.75$ and a removal threshold $\tau_r = 1.0$. This higher removal threshold is essential given the scarcity of multi-hop training samples, as it ensures the agent thoroughly masters the foundational reasoning patterns at initial hint levels (H_0) before attempting unguided exploration, thereby preventing premature curriculum shifts that could destabilize policy optimization. As reported in Table 3, TRACE-GRPO consistently outperforms both the unfine-tuned baseline and standard GRPO, achieving a significant lead in Best@8. These results demonstrate that TRACE-GRPO effectively generalizes to longer multi-hop reasoning tasks beyond the SQL-Search modality, showcasing robust stability even as the composition of cross-document dependencies becomes increasingly complex.

Method (Qwen3-4B)	Best@1	Best@2	Best@4	Best@8	Acc _{avg}
Unfine-tuned Baseline	0.1339	0.2017	0.2644	0.3288	0.1277
Standard GRPO	0.2266	0.2987	0.3794	0.4464	0.2329
TRACE-GRPO	0.2695	0.3605	0.4352	0.4953	0.2731

Table 3: Generalization results on MuSiQue (Mixed 3-hop and 4-hop). We compare an unfine-tuned Qwen3-4B baseline against standard GRPO and our TRACE-GRPO.

6 Conclusion

In this paper, we addressed the fundamental challenge of optimizing LLM agents for complex hybrid reasoning tasks. These tasks require the tightly coupled interleaving of structured database execution and unstructured semantic retrieval, presenting a severe sparse-reward bottleneck for standard reinforcement learning. To overcome this, we proposed a novel data-algorithm co-design paradigm.

Specifically, we introduced HyDRA-Bench, a rigorous benchmark and synthesis pipeline designed to simulate multi-tool enterprise dependencies. Central to our approach is the TRACE-GRPO framework, which repurposes intermediate reasoning traces from data synthesis into a goal-conditioned, adaptive curriculum. By dynamically scheduling these explicit hints based on real-time policy reachability, the agent can effectively internalize complex structural invariants, including deterministic SQL syntax, and transfer them to an autonomous policy via shared transformer weights.

Empirical evaluations demonstrate that TRACE-GRPO significantly accelerates training convergence and exploration efficiency. Our results show that this framework not only enables 4B models to approach proprietary performance, but also allows an 8B open-weights agent to surpass Claude-Opus-4.5 on hybrid reasoning, while generalizing to extended multi-hop environments such as MuSiQue. Future work will explore expanding this co-design philosophy to open-ended environments where an independent critic model can dynamically verify and generate reasoning traces on-the-fly, further reducing the reliance on static synthesis pipelines.

Acknowledgments

This work is supported by the US National Science Foundation under Grant CAREER-2305491.

References

- Ankush Agarwal, Chaitanya Devaguptapu, et al. Hybrid graphs for table-and-text based question answering using llms. *arXiv preprint arXiv:2501.17767*, 2025.
- Jayetri Bardhan, Bushi Xiao, and Daisy Zhe Wang. Ttqa-rs-a break-down prompting approach for multi-hop table-text question answering with reasoning and summarization. *arXiv preprint arXiv:2406.14732*, 2024.
- Kianté Brantley, Mingyu Chen, Zhaolin Gao, Jason Lee, Wen Sun, Wenhao Zhan, and Xuezhou Zhang. Accelerating rl for llm reasoning with optimal advantage regression. *Advances in Neural Information Processing Systems*, 38:151492–151531, 2026.
- Wenhu Chen, Ming-Wei Chang, Eva Schlinger, William Wang, and William W Cohen. Open question answering over tables and text. *arXiv preprint arXiv:2010.10439*, 2020a.
- Wenhu Chen, Hanwen Zha, Zhiyu Chen, Wenhan Xiong, Hong Wang, and William Yang Wang. Hybridqa: A dataset of multi-hop question answering over tabular and textual data. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 1026–1036, 2020b.
- Yuqian Fu, Tinghong Chen, Jiajun Chai, Xihuai Wang, Songjun Tu, Guojun Yin, Wei Lin, Qichao Zhang, Yuanheng Zhu, and Dongbin Zhao. Srft: A single-stage method with supervised and reinforcement fine-tuning for reasoning. *arXiv preprint arXiv:2506.19767*, 2025.
- Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yixin Dai, Jiawei Sun, Haofen Wang, Haofen Wang, et al. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2(1):32, 2023.
- Mingqian He, Yongliang Shen, Wenqi Zhang, Qiuying Peng, Jun Wang, and Weiming Lu. Star-sql: Self-taught reasoner for text-to-sql. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 24365–24375, 2025.
- Zeyu Huang, Tianhao Cheng, Zihan Qiu, Zili Wang, Yinghui Xu, Edoardo M Ponti, and Ivan Titov. Blending supervised and reinforcement fine-tuning with prefix sampling. *arXiv preprint arXiv:2507.01679*, 2025.
- Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong C Park. Adaptive-rag: Learning to adapt retrieval-augmented large language models through question complexity. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 7036–7050, 2024.
- Fangyu Lei, Xiang Li, Yifan Wei, Shizhu He, Yiming Huang, Jun Zhao, and Kang Liu. S3 hqa: A three-stage approach for multi-hop text-table hybrid question answering. *arXiv preprint arXiv:2305.11725*, 2023.
- Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474, 2020.
- Jinyang Li, Binyuan Hui, Ge Qu, Jiayi Yang, Binhua Li, Bowen Li, Bailin Wang, Bowen Qin, Ruiying Geng, Nan Huo, et al. Can llm already serve as a database interface? a big bench for large-scale database grounded text-to-sqls. *Advances in Neural Information Processing Systems*, 36, 2024.
- Xiaoxi Li, Wenxiang Jiao, Jiarui Jin, Guanting Dong, Jiajie Jin, Yinuo Wang, Hao Wang, Yutao Zhu, Ji-Rong Wen, Yuan Lu, et al. Deepagent: A general reasoning agent with scalable toolsets. *arXiv preprint arXiv:2510.21618*, 2025.

- Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let’s verify step by step. In *The Twelfth International Conference on Learning Representations*, 2023.
- Huanyu Liu, Jia Li, Chang Yu, Taozhi Chen, Yihong Dong, Lecheng Wang, XiaoLong Hu, and Ge Li. Evocot: Overcoming the exploration bottleneck in reinforcement learning. *arXiv preprint arXiv:2508.07809*, 2025a.
- Mingyang Liu, Gabriele Farina, and Asuman Ozdaglar. Uft: Unifying supervised and reinforcement fine-tuning. *arXiv preprint arXiv:2505.16984*, 2025b.
- Shih-Yang Liu, Xin Dong, Ximing Lu, Shizhe Diao, Peter Belcak, Mingjie Liu, Min-Hung Chen, Hongxu Yin, Yu-Chiang Frank Wang, Kwang-Ting Cheng, et al. Gdpo: Group reward-decoupled normalization policy optimization for multi-reward rl optimization. *arXiv preprint arXiv:2601.05242*, 2026.
- Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025c.
- Lu Ma, Hao Liang, Meiyl Qiang, Lexiang Tang, Xiaochen Ma, Zhen Hao Wong, Junbo Niu, Chengyu Shen, Runming He, Yanhao Li, et al. Learning what reinforcement learning can’t: Interleaved online fine-tuning for hardest questions. *arXiv preprint arXiv:2506.07527*, 2025.
- Shishir G Patil, Tianjun Zhang, Xin Wang, and Joseph E Gonzalez. Gorilla: Large language model connected with massive apis. *Advances in Neural Information Processing Systems*, 37: 126544–126565, 2024.
- Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, et al. Kilt: a benchmark for knowledge intensive language tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 2523–2544, 2021.
- Shengmin Piao, Jieun Lee, and Sanghyun Park. Lite-sql: A lightweight and efficient text-to-sql framework with vector-based schema linking and execution-guided self-correction. In *Findings of the Association for Computational Linguistics: EACL 2026*, pp. 3593–3608, 2026.
- Mohammadreza Pourreza and Davood Rafiei. Dts-sql: Decomposed text-to-sql with small large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 8212–8220, 2024.
- Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, et al. Toolllm: Facilitating large language models to master 16000+ real-world apis. *arXiv preprint arXiv:2307.16789*, 2023.
- Yuxiao Qu, Amrith Setlur, Virginia Smith, Ruslan Salakhutdinov, and Aviral Kumar. Pope: Learning to reason on hard problems via privileged on-policy exploration. *arXiv preprint arXiv:2601.18779*, 2026.
- Daniel Ritter, Owen Oertell, Bradley Guo, Jonathan Chang, Kianté Brantley, and Wen Sun. Llm can learn to reason via off-policy rl. *arXiv preprint arXiv:2602.19362*, 2026.
- John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.
- Qi Shi, Han Cui, Haofeng Wang, Qingfu Zhu, Wanxiang Che, and Ting Liu. Exploring hybrid question answering via program-based prompting. *arXiv preprint arXiv:2402.10812*, 2024.

- Kimi Team, Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, et al. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*, 2025.
- Harsh Trivedi, Niranjana Balasubramanian, Tushar Khot, and Ashish Sabharwal. musique: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022.
- Bing Wang, Changyu Ren, Jian Yang, Xinnian Liang, Jiaqi Bai, Linzheng Chai, Zhao Yan, Qian-Wen Zhang, Di Yin, Xing Sun, et al. Mac-sql: A multi-agent collaborative framework for text-to-sql. In *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 540–557, 2025.
- Jian Wu, Yicheng Xu, Yan Gao, Jian-Guang Lou, Börje F Karlsson, and Manabu Okumura. Tacr: A table-alignment-based cell-selection and reasoning model for hybrid question-answering. *arXiv preprint arXiv:2305.14682*, 2023.
- Junde Wu, Jiayuan Zhu, Yuyuan Liu, Min Xu, and Yueming Jin. Agentic reasoning: A streamlined framework for enhancing llm reasoning with agentic tools. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 28489–28503, 2025.
- Jianhao Yan, Yafu Li, Zican Hu, Zhi Wang, Ganqu Cui, Xiaoye Qu, Yu Cheng, and Yue Zhang. Learning to reason under off-policy guidance. *arXiv preprint arXiv:2504.14945*, 2025.
- Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*, 2022.
- Zhewei Yao, Guoheng Sun, Lukasz Borchmann, Gaurav Nuti, Zheyu Shen, Minghang Deng, Bohan Zhai, Hao Zhang, Ang Li, and Yuxiong He. Arctic-text2sql-r1: Simple rewards, strong reasoning in text-to-sql. *arXiv preprint arXiv:2505.20315*, 2025.
- Qiyang Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.
- Aohan Zeng, Xin Lv, Qinkai Zheng, Zhenyu Hou, Bin Chen, Chengxing Xie, Cunxiang Wang, Da Yin, Hao Zeng, Jiajie Zhang, et al. Glm-4.5: Agentic, reasoning, and coding (arc) foundation models. *arXiv preprint arXiv:2508.06471*, 2025.
- Jingyi Zhang, Jiaxing Huang, Huanjin Yao, Shunyu Liu, Xikun Zhang, Shijian Lu, and Dacheng Tao. R1-vl: Learning to reason with multimodal large language models via step-wise group relative policy optimization, 2025a. URL <https://arxiv.org/abs/2503.12937>.
- Jixiao Zhang and Chunsheng Zuo. Grpo-lead: A difficulty-aware reinforcement learning approach for concise mathematical reasoning in language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 5642–5665, 2025.
- Kaiyi Zhang, Ang Lv, Jinpeng Li, Yongbo Wang, Feng Wang, Haoyuan Hu, and Rui Yan. Stephint: Multi-level stepwise hints enhance reinforcement learning to reason. *arXiv preprint arXiv:2507.02841*, 2025b.
- Xuechen Zhang, Zijian Huang, Yingcong Li, Chenshun Ni, Jiasi Chen, and Samet Oymak. Bread: Branched rollouts from expert anchors bridge sft & rl for reasoning. *arXiv preprint arXiv:2506.17211*, 2025c.
- Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models. *arXiv preprint arXiv:2601.18734*, 2026.
- Yilun Zhao, Yunxiang Li, Chenying Li, and Rui Zhang. Multihiertr: Numerical reasoning over multi hierarchical tabular and textual data. *arXiv preprint arXiv:2206.01347*, 2022.

Zihui Zhao and Zechang Li. Reflective preference optimization (rpo): Enhancing on-policy alignment via hint-guided reflection. *arXiv preprint arXiv:2512.13240*, 2025.

Wanjun Zhong, Junjie Huang, Qian Liu, Ming Zhou, Jiahai Wang, Jian Yin, and Nan Duan. Reasoning over hybrid chain for table-and-text open domain qa. *arXiv preprint arXiv:2201.05880*, 2022.

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. *arXiv preprint arXiv:2105.07624*, 2021.

7 Appendix

7.1 Extended Related Work

7.1.1 Hybrid Table-Text Question Answering

Hybrid question answering requires integrating structured data from tables with unstructured text. Benchmarks such as HybridQA (Chen et al., 2020b) and OTT-QA (Chen et al., 2020a) evaluate multi-hop reasoning across linked tables and passages, while domain-specific datasets like TAT-QA (Zhu et al., 2021) and MultiHierTT (Zhao et al., 2022) introduce additional numerical and hierarchical complexity. Early approaches rely on specialized pipelines or structured reasoning frameworks to align table and text information (Lei et al., 2023; Wu et al., 2023; Zhong et al., 2022; Agarwal et al., 2025), while more recent LLM-based methods decompose queries into intermediate reasoning steps or programs (Bardhan et al., 2024; Shi et al., 2024). Despite these advances, existing benchmarks treat tables as static contexts rather than interactive systems. Models operate over pre-retrieved table snippets instead of issuing executable queries over large-scale databases. As a result, they fail to capture realistic hybrid reasoning workflows, where agents must dynamically interleave structured queries with unstructured retrieval.

7.1.2 General Tool-Use Agents

Tool-use agents extend LLMs with the ability to interact with external systems. Early works such as Gorilla (Patil et al., 2024) and ToolBench (Qin et al., 2023) focus on generating accurate function calls, while frameworks like ReAct (Yao et al., 2022) enable iterative reasoning through interleaved thought-action loops. Recent systems further improve long-horizon tool orchestration via large-scale training and architectural innovations (Team et al., 2025; Zeng et al., 2025; Li et al., 2025). However, these approaches are primarily developed for domains with explicit and informative feedback, such as coding or mathematical reasoning. In contrast, hybrid information-seeking tasks involve weak and ambiguous signals. For instance, database interactions often result in “silent failures”, where syntactically valid SQL queries return empty results due to semantic misalignment. Such signals are insufficient for reliable error detection and recovery, making standard agentic loops ineffective in these settings.

7.1.3 Hint-Based Reinforcement Learning

Standard Reinforcement Learning (RL) methods such as PPO (Schulman et al., 2017) and GRPO (Shao et al., 2024) suffer from sparse rewards in multi-hop tasks, since early mistakes receive no direct feedback. Process Reward Models (PRMs) partially address this issue by scoring intermediate steps (Lightman et al., 2023), but their reliance on expensive step-wise annotation limits scalability. To improve exploration without manual labeling, recent work introduces hint-based or partial-trajectory guidance. Early methods such as Prefix-RFT (Huang et al., 2025) and SRFT (Fu et al., 2025) use partial solutions to stitch successful trajectories. UFT (Liu et al., 2025b) gradually shortens these hints to encourage a transition from imitation to autonomous reasoning, while LUFFY (Yan et al., 2025) incorporates off-policy teacher traces to mitigate cold-start exploration. BREAD (Zhang et al., 2025c) further densifies learning signals by adaptively inserting expert prefixes when self-generated rollouts fail. Related adaptive approaches, including StepHint (Zhang et al., 2025b) and ReLIFT (Ma et al., 2025), vary the amount of guidance according to model uncertainty, and POPE (Qu et al., 2026) frames this setting as privileged on-policy exploration using oracle solution prefixes. However, these methods are primarily designed for continuous text reasoning and do not readily extend to settings with discrete, executable actions such as SQL queries and retrieval calls. Moreover, they typically rely on externally provided hints, teacher traces, or oracle trajectories, limiting their applicability in scalable and realistic environments.

7.1.4 Hybrid Off-Policy and On-Policy RL

Recent work combines offline guidance with on-policy optimization to improve sample efficiency for LLM reasoning. OAPL (Ritter et al., 2026) trains directly on off-policy rollout data under policy lag. A*-PO (Brantley et al., 2026) first estimates an optimal value function from offline reference-policy samples, then performs on-policy updates. OPSD (Zhao et al., 2026) distills from a privileged-context teacher into a non-privileged student over the model’s own rollouts. We characterize TRACE-GRPO as a hybrid paradigm in which on-policy GRPO rollouts are scaffolded by offline-synthesized hierarchical hints. Hints are produced once during dataset construction and stored with each sample, while training only selects the hint tier and samples rollouts from the current policy with live tool execution.

7.2 Dataset Statistics

This section reports detailed statistics for the structured side of HyDRA-Bench, which is grounded in BIRD-SQL (Li et al., 2024) after trace-aware filtering.

Database Coverage. HyDRA-Bench retains 80 of BIRD-SQL’s 95 databases (69 train / 11 dev), with the official BIRD dev split held out from training. These databases comprise 597 tables and 4,337 columns, with 7.6 tables and 54 columns per database on average (max 65 tables and 455 columns per database). They span 12+ domains including sports, geography, media, education, healthcare, finance, government, codebase, etc.

SQL Output Characteristics. Across all 18,043 examples, SQL execution returns a single-row result in 91.8% of cases (3.9% with 2 to 3 rows, 4.3% with 4 to 10 rows), consistent with our cardinality filter. Multi-column outputs account for 13.0% of SQL hops, and approximately 48% involve numeric aggregations (AVG/SUM/COUNT/MAX/MIN). The remainder cover single entities, categorical labels, dates, or short lists.

SQL Query Complexity. At the query level, SQL hops average 3.3 operations (mode 3, max 10), and only 0.1% reduce to a pure single-value lookup. Table 4 summarizes the multi-label distribution of SQL constructs across all 18,043 queries.

SQL Feature	Freq (%)	SQL Feature	Freq (%)
WHERE	88.5	GROUP BY	10.9
JOIN	80.0	CAST	10.5
Aggregation	49.6	DISTINCT	8.7
Date/Temporal	29.9	LIKE	7.3
LIMIT	25.4	Nested Subquery	6.0
ORDER BY	24.9	HAVING	0.7
CASE WHEN	10.9		

Table 4: Multi-label SQL construct frequencies across 18,043 SQL hops in HyDRA-Bench.

7.3 Runtime Environment

This section describes the runtime environment exposed to the agent during training and evaluation on HyDRA-Bench.

DB Binding. Each question is bound to one database via its `db_id`. HyDRA-Bench therefore evaluates intra-database reasoning rather than cross-database routing.

Agent Tools. The agent has two tools exposed via strict OpenAI-style function calling. `run_sql(code_query)` executes a single read-only SQLite statement under a 10-second wall-clock timeout and returns structured execution results. `cortex_search(search_query)` retrieves passages from the Wiki2019 index described below.

Tool Schemas. The function schemas below are taken from our runtime implementation.

run_sql

Description: A tool for executing SQL queries against a connected database. Use this tool when you need to run, test, or evaluate SQL statements and retrieve structured results.

Parameters:

- `code.query` (string, required): The SQL query to be executed against the database. Must be a valid SQL statement, including any necessary filters, joins, sorting, or aggregation.

cortex_search

Description: An information retrieval system that searches through Wikipedia 2019 and other knowledge sources to find factual information relevant to the query. Use this tool when you need to verify facts, get background information, or access knowledge that might not be in the database. For optimal results, construct specific, focused queries around a single concept or question.

Parameters:

- `search_query` (string, required): A precise, focused search query targeting specific information. Avoid vague or overly broad queries.

DB Discovery. Database schema and sample values are deliberately omitted from the prompt context. The agent instead discovers schema online via `sqlite_master` and `PRAGMA` queries through `run_sql`. This setting is strictly harder than an oracle-schema baseline and avoids prompt bloat on databases with up to 65 tables and 455 columns.

Retrieval Corpus and Search. For retrieval we use the KILT 2019-08-01 Wikipedia dump (Petroni et al., 2021). Abstracts are kept as-is, and article bodies are split into 4 to 5 sentence passages, yielding approximately 38.5M indexed chunks. Retrieval is served by Cortex Search with vector, keyword, and semantic reranking, using `snowflake-arctic-embed-m-v1.5` as the embedding model. We return $\text{top-}k = 3$ passages per call, allow up to 10 search calls per episode, and cap each passage at 3K characters.

Judge	Acc	Align w/ human
GPT-o4-mini	50.0%	99.5%
GPT-5.4	50.0%	98.5%
Claude Sonnet 4.6	53.5%	97.0%
Claude Opus 4.6	51.5%	98.0%
Human	50.5%	100%

Table 5: Judge validation on 200 stratified trajectories from the Qwen3-8B evaluation. Acc is the fraction judged correct. Align w/ human is agreement with the human annotator.

7.4 LLM Judge Validation

Because HyDRA-Bench gold answers are natural-language explanations, exact string match is insufficient for evaluation. We therefore use GPT-o4-mini as the primary LLM judge throughout our experiments. To validate this choice, we randomly sampled 100 trajectories that GPT-o4-mini judged correct and 100 it judged incorrect from the Qwen3-8B evaluation, and re-labeled each with four LLM judges and one human annotator.

As shown in Table 5, all four LLM judges agree with the human annotator on 97.0 to 99.5% of the 200 samples, and Acc lies within 3 points across all judges. GPT-o4-mini in particular matches the human label on 199 of 200 samples, with fraction-correct within 0.5 points of

the human. These results indicate that LLM-as-a-judge provides a reliable proxy for human assessment on HyDRA-Bench.

7.5 HyDRA-Bench Example

Instance ID: train_s2a_3
Final Question: What is the average height of crew members in the team within a film studio that works on aspects like storyboarding or 3D modeling? Final Answer: The average height of crew members in the Animation Department, which works on aspects like storyboarding or 3D modeling, is approximately 1.798 meters.
Intermediate Reasoning Traces (Structured Artifacts)
• Step 1 (Search): <i>Question:</i> What is the term for the team within a film studio that works on aspects like storyboarding or 3D modeling? <i>Answer:</i> Animation department. • Step 2 (SQL): <i>Question:</i> What is the average heights of crew members from Animation department? <i>Query:</i> <code>SELECT AVG(T1.height_meters) FROM Person AS T1 INNER JOIN Credit AS T2 ON T1.name = T2.person WHERE T2.category = 'Animation Department' ;</code> <i>Output:</i> <code>AVG(T1.height_meters): 1.7975384615384615</code> <i>Answer:</i> The average height of crew members from the Animation Department is approximately 1.798 meters.
Four-Level Hint Hierarchy ($\mathcal{H}$)
• Level 0 (Empty): There are no hints on this problem, please proceed independently. • Level 1 (Low): To solve this complex problem, we must break it down into two dependent steps. We need to first investigate the question: “What is the term for the team within a film studio that works on aspects like storyboarding or 3D modeling?”. Once we have determined that answer, we will be able to fill in the missing information for the second step, allowing us to find: “What is the average heights of crew members from #1?”. • Level 2 (Medium): Our strategy involves breaking this into two steps, where we first utilized the “search_tool” to investigate the question: “What is the term for the team within a film studio that works on aspects like storyboarding or 3D modeling?”. Then we will successfully determine the answer to the first question: “Animation department”. With this crucial piece of information now established, the next logical step is to employ the “run_sql” to answer: “What is the average heights of crew members from Animation Department? ”. • Level 3 (High): To determine the final answer, we need to decompose the problem into two sequential stages. First, we use the “search_tool” to answer the first question: “What is the term for the team within a film studio that works on aspects like storyboarding or 3D modeling?”. Then we will successfully determine the answer to the first question: “Animation department”. With this crucial piece of information now established, the next logical step is to employ the “run_sql” to answer: “What is the average heights of crew members from Animation Department?”. To do this, we need to give the specific sql query: “<code>SELECT AVG(T1.height_meters) FROM Person AS T1 INNER JOIN Credit AS T2 ON T1.name = T2.person WHERE T2.category = 'Animation Department' ;</code>”

To illustrate the structural complexity of HyDRA-Bench, we provide a concrete example of a *Search-to-SQL* (S2A) instance. This example illustrates the structural composition of a typical HyDRA-Bench instance. It demonstrates how intermediate reasoning traces, naturally

generated during the synthesis of hybrid queries, are leveraged to construct a four-level hint hierarchy $\mathcal{H}$ that provides progressively dense guidance for reinforcement learning.

7.6 Detailed Procedure of TRACE-GRPO Curriculum Learning

Algorithm 1 Dynamic curriculum learning via hint-guided rollouts.

Require: Dataset $\mathcal{D}$, success threshold τ_s , removal threshold τ_r , rollouts n

- 1: Initialize $h_{i,0} \leftarrow H_0$ (empty) and $L_i \leftarrow H_3$ (Locking upper bound) for all $i \in \mathcal{D}$
- 2: **for** epoch $t = 1$ **to** T **do**
- 3: **for** sample $i \in \mathcal{D}$ **do**
- 4: Generate n rollouts using hint $h_{i,t}$
- 5: Calculate success rate $\mu_{i,t} = \frac{1}{n} \sum_{j=1}^n \mathbb{I}(\text{success}_{i,j})$
- 6: **if** $\mu_{i,t} < \tau_s$ **then**
- 7: $h_{i,t+1} \leftarrow \min(h_{i,t} + 1, L_i)$
- 8: **else**
- 9: $h_{i,t+1} \leftarrow \max(h_{i,t} - 1, H_0)$
- 10: $L_i \leftarrow \min(L_i, h_{i,t+1})$
- 11: **end if**
- 12: **if** $h_{i,t+1} = H_0$ **and** $\mu_{i,t} > \tau_r$ **then**
- 13: $\mathcal{D} \leftarrow \mathcal{D} \setminus \{i\}$
- 14: **end if**
- 15: **end for**
- 16: **end for**
