HYBRIDRESEARCHER: BRIDGING THE STRUCTURED-UNSTRUCTURED BOUNDARY FOR DEEP RESEARCH AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

While LLM agents are increasingly capable of gathering and synthesizing evidence from a single homogeneous source, such as structured tables or unstructured data, real-world analytical questions frequently span heterogeneous sources. For example, answering one enterprise question may require both a result computed over an internal database and evidence found on the web. Recent evaluations on benchmarks like HybridDeepResearch show that even frontier agents struggle when evidence must cross this boundary. While these benchmarks have exposed the limitation, methodologies to explicitly teach cross-source reasoning remain absent. To address this, we introduce HYBRIDRESEARCHER-4B, a 4B agent trained to bridge evidence across the structured and unstructured boundary. To build it, we synthesize cross-source bridge trajectories by starting with validated single-source tasks—such as existing text-to-SQL or web search queries—and fabricating the missing source around a bridge entity drawn from the seed's own output. We retain a sample only when strict construction gates confirm that neither side alone suffices, ensuring the data requires true cross-source interdependence. Finally, we train on this curated dataset using supervised fine-tuning followed by reinforcement learning. Evaluated on the existing HybridDeepResearch benchmark, HYBRIDRESEARCHER-4B improves over our base model, Qwen3.5-4B, by 30.0 points pass@8 and surpasses the far larger Qwen3.5-397B-A17B by 8.3 points, while also improving over its base model on established, held-out single-source benchmarks (BrowseComp and BrowseComp-Plus), demonstrating that this complex cross-source reasoning can be taught directly to small models without triggering catastrophic forgetting.

1 INTRODUCTION

LLM agents extend language models beyond closed-form reasoning by letting them interact with external data sources through tools (Yao et al., 2023; Schick et al., 2023). A capable agent can write and execute SQL against a database, navigate the open web to locate and synthesize evidence, or query domain-specific APIs to retrieve information that parametric knowledge alone cannot supply. Training agents for these capabilities has advanced rapidly: recent recipes that combine synthetic data construction, supervised fine-tuning, and reinforcement learning have produced compact models that rival or surpass frontier models, in domains such as text-to-SQL (Li et al., 2025a; Yao et al., 2026), and open-web deep research (Jin et al., 2025; Zheng et al., 2025; Xie et al., 2026). However, these recipes typically operate within a single homogeneous source, such as a database or the web, rarely bridging both in the same reasoning chain. Real-world analytical tasks rarely observe this boundary: domains routinely require synthesizing evidence across structured and unstructured sources to answer a single question—such as crossing internal transaction databases with public SEC filings for financial auditing, or merging inventory SQL with global news for supply-chain risk assessment (Abaskohi et al., 2026; Wang et al., 2026b; Wu et al., 2026b). Recent benchmarks such as HybridDeepResearch (Wu et al., 2026b) make this gap measurable, revealing that even frontier agents such as GPT-5 (OpenAI, 2025) and Claude-Sonnet-4.6 (Anthropic, 2026) struggle to synthesize answers when evidence must cross this boundary. Yet while the evaluation gap is now visible, the training gap remains: to our knowledge, no prior work provides a training framework, training

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.

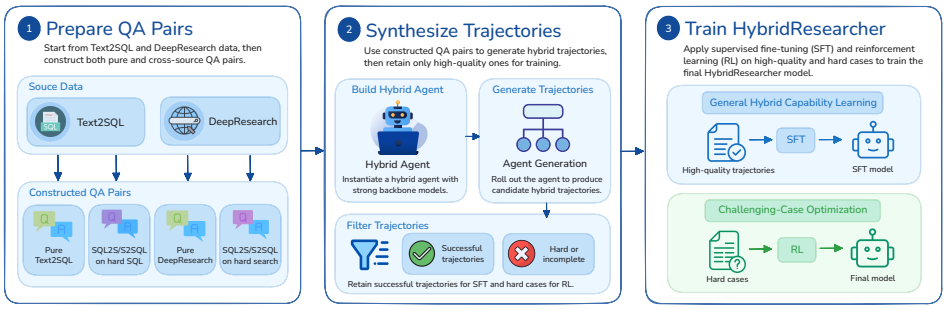


Figure 1: We start from validated single-source text-to-SQL and deep-research seeds, construct both single-source and cross-source (SQL2S/S2SQL) QA pairs, and use a hybrid agent to generate candidate trajectories. Successful trajectories are retained for supervised fine-tuning, while hard or incomplete cases are used for reinforcement learning, yielding HYBRIDRESEARCHER.

data, or a trained model for agents that must bridge structured and unstructured sources within a single reasoning chain.

To address this training gap, we ask: why does this capability gap persist despite rapid progress on each source in isolation? We identify two core obstacles. First, **component capabilities do not automatically compose into hybrid reasoning**. Fine-tuning a model on single-source corpora (whether SQL-only, search-only, or their union) without bridged trajectories fails to yield cross-source competence. While the union model almost fully recovers parallel tasks that exercise both skills independently, performance on genuinely directional tasks, where the output of one source conditions the other, remains severely impaired. This demonstrates that hybrid data must explicitly teach the bridge itself: the interdependent execution between sources rather than the isolated skills. Second, naive hybrid synthesis is **vulnerable to shortcut learning**. Among teacher trajectories accepted by standard correctness filters, 79.5% never execute a substantive SQL query. Instead, the model answers directly from parametric memory, bypassing the cross-source hop entirely. A model trained on such flawed trajectories merely learns to fake the bridge, a failure mode that scaling uncurated data only exacerbates. Overcoming both obstacles by explicitly teaching the cross-source bridge while enforcing strict anti-shortcut guarantees forms the foundation of our methodology.

We introduce HYBRIDRESEARCHER, a training framework that teaches hybrid deep research to small models, by extending validated single-source tasks across the structured/unstructured boundary. The core idea is that hybrid training data need not be fabricated from scratch. Each instance starts from a seed whose difficulty is already established—an existing text-to-SQL task or a web-search task. We synthesize the missing source around a bridge entity drawn from the seed's own output, so that the seed's difficulty is inherited rather than re-established. To directly address the shortcut problem, we retain an instance only when strict construction gates deterministically verify that neither source alone suffices, ensuring the bridge entity forms the unique, necessary path between them (examples in Appendix C.4 and D.3). While we instantiate this framework for the SQL–web pairing, the structural recipe—requiring only a validated single-source task to seed from and an entity that can bridge across boundaries—provides a clear blueprint for extending agentic reasoning to other heterogeneous combinations in the future. The curated data serves both supervised fine-tuning and reinforcement learning, trained in a reproducible offline environment built around a frozen corpus and local retriever. To optimize this complex policy, our RL recipe introduces a majority-voting judge ensemble to mitigate reward noise, alongside strict hard-gating mechanisms that terminate and penalize degenerate, brute-force tool usage. Figure 1 summarizes this pipeline.

Evaluated on the HybridDeepResearch benchmark (Wu et al., 2026b), HYBRIDRESEARCHER-4B improves over its Qwen3.5-4B (Qwen Team, 2026a) base model by 30.0 points pass@8 and surpasses the far larger Qwen3.5-397B-A17B-FP8 by 8.3 points. Crucially, this improvement does not come at the cost of single-source ability: HYBRIDRESEARCHER-4B also outperforms its base model on the held-out SQL-only benchmark Spider 2.0-lite (Lei et al., 2025) and on the web-search benchmarks BrowseComp (Wei et al., 2025) and BrowseComp-Plus (Chen et al., 2026). By explicitly teaching the cross-source bridge, our framework successfully imparts advanced hybrid reasoning capabilities to a compact model without triggering catastrophic forgetting (Luo et al., 2025).

2 Related Work

Single-source Agents. Tool-using LLM agents have made rapid progress on tasks that draw evidence from one type of source. For databases, benchmarks have moved from single-turn semantic parsing (Yu et al., 2018; Li et al., 2023) to realistic workflows over large enterprise schemas (Lei et al., 2025; BIRD Team, 2025), and agents now explore the database and revise their queries based on execution results (Deng et al., 2025). Small models trained on synthetic data (Li et al., 2025a) or with reinforcement learning from execution rewards (Ma et al., 2025; Yao et al., 2026; Liu et al., 2025) are now competitive with much larger models on these benchmarks. For the web, search agents interleave reasoning with retrieval (Yao et al., 2023; Jin et al., 2025; Song et al., 2025; Zheng et al., 2025), and recent systems can answer long-horizon questions whose answers are hard to find, as measured by BrowseComp (Wei et al., 2025). These systems are trained largely on synthetic tasks designed to be hard, for example by obscuring key entities (Li et al., 2025b), formalizing the structure of information-seeking tasks (Tao et al., 2026), sampling paths from knowledge graphs (Lu et al., 2025), or generating fully synthetic research tasks (Xie et al., 2026), and then refined with reinforcement learning (Gao et al., 2025). However, in all of these works, the agents work with a single source. In contrast, we train an agent that crosses the boundary between structured and unstructured sources, carrying evidence found in one source into its search of the other.

Reasoning across Heterogeneous Sources. Hybrid question answering has long studied questions that require both tables and text (Chen et al., 2020; 2021a;b; Zhu et al., 2021), but the evidence is either given or retrieved in one step, and the model neither executes queries nor decides which source to consult next. Other work adds language-model reasoning to database queries (Liu et al., 2024). Recent benchmarks evaluate agents that combine evidence from heterogeneous sources, including databases and documents (Wang et al., 2026b; Ma et al., 2026; Li et al., 2026), enterprise files and the web (Abaskohi et al., 2026), web pages and knowledge-graph APIs (Yang et al., 2024), databases and web search (Wu et al., 2026b), and multimodal evidence on the web (Wang et al., 2026a). These benchmarks measure how well agents combine sources, but few works train agents to do so. DSL-R1 (Hu et al., 2026) merges structured filters and vector search into a single query language over a closed collection, so the agent never has to carry a result from one kind of source to another.

Wu et al. (2026a) synthesize simple two-hop questions that pair a SQL query with a single Wikipedia passage lookup, and train agents with hint-guided reinforcement learning. We instead present a full training recipe—from data construction to supervised fine-tuning and reinforcement learning—for significantly more challenging, multi-hop questions that combine hard SQL execution with long-horizon web search.

3 Methodology: Synthesizing Cross-Source Trajectories

To train agents for hybrid reasoning, we require trajectories that exhibit true cross-source interdependence. Training on existing single-source data does not produce this capability, and synthesizing hybrid data naively invites shortcuts (§3.1). We therefore build hybrid trajectories from validated single-source seeds, starting from either text-to-SQL tasks (§3.2) or search tasks (§3.3).

3.1 Two Obstacles to Training Hybrid Agents

Single-Source Data Does Not Teach Hybrid Capability. Text-to-SQL (Yu et al., 2018; Li et al., 2023) and web search (Du et al., 2026; Chu et al., 2026) training data are available in large quantities, whereas hybrid trajectories have to be synthesized. We therefore first test whether single-source data alone is enough. We fine-tune Qwen3.5-4B on successful trajectories from text-to-SQL tasks only, web-search tasks only, or their union, and evaluate on the hard subset of HybridDeepResearch (Wu et al., 2026b), whose 120 questions each require both SQL and web search. None of the three mixtures helps. SQL-only training reaches 31.67 pass@8, within one question of the base model's 30.83, while union and search-only training fall to 29.17 and 15.83. Single-source trajectories never use the output of one tool as the input to the other, so training on them does not teach this step.

Naive Hybrid Synthesis Invites Shortcuts. Synthesizing hybrid questions and keeping the teacher trajectories that reach the correct answer is also not enough, because a correct answer does not show that both sources were used. When the facts on the search side are well known, the teacher can

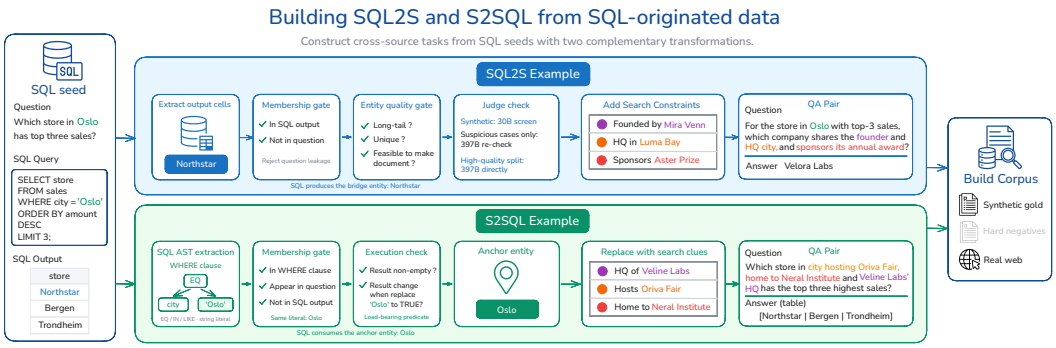


Figure 2: Overview of sql-originated data construction.**Top (`SQL2S`):** an output cell (*Northstar*) that passes the membership gate, entity-quality gate, and judge check becomes the bridge; coined facts about it (HQ city ...) become search constraints that only one hub, *Velora Labs*, satisfies. **Bottom (`S2SQL`):** a `WHERE` literal from the SQL AST (*Oslo*) that passes the membership gate and execution check becomes the anchor, and the answer is the unchanged SQL output. Gold and hard-negative documents for every clue are inserted with real web text by the retriever used at evaluation. (Details: §3.2 and Appendix C)

recall them instead of searching, and when the question reveals the bridge entity, the teacher can skip the hop that was meant to produce it. Models tend to trust such parametric knowledge even when it conflicts with retrieved evidence (Xie et al., 2024; Wang et al., 2025a). This risk is concrete in our setting. Bridge entities drawn from real databases are often famous, and the model's prior can override what it retrieves. Such trajectories lead the model to answer without crossing the boundary.

3.2 SQL-Originated Trajectories

We construct hybrid tasks from complete text-to-SQL seeds by deriving a multi-constraint web-retrieval subtask from a seed entity (Figure 2, Appendix C).

Building SQL2S. (i) Bridge Selection: We select a cell from the executed output table to serve as the bridge entity (e.g., *Northstar* in Figure 2). A membership gate ensures this cell appears in the output but not in the original question, deterministically forcing the agent to execute the SQL query to discover it. **(ii) Hop One (Coined Facts):** To prevent the model from bypassing the tool boundary via parametric memory, we invent novel facts (e.g., business rules) about the bridge entity using pronounceable pseudo-words. Each fact is paired with a fabricated gold document and typed hard negatives that perturb exactly one constraint. **(iii) Hop Two (The Intersection Hub):** The solver must resolve the bridge by SQL, resolve each coined fact by search, and find the single hub that all facts point to (e.g., *Velora Labs*). The gold hub satisfies all c constraints, while each near-miss distractor satisfies exactly $c - 1$, forcing rigorous multi-document cross-referencing. **(iv) Verification:** The item must pass deterministic gates (e.g., ensuring the constraint intersection is a singleton) and model-based checks, including a mask test proving the answer cannot be guessed from negative documents.

Building S2SQL. (i) Anchor Selection: We select a string literal from an equality, inclusion, or pattern-matching predicate in the gold SQL query to serve as the anchor (Appendix C.4, *Oslo* in Figure 2). An execution test requires that removing the predicate changes the result, and membership constraints require the literal to occur in the original question but not in the output. Together, these checks verify the anchor is load-bearing: the agent must resolve it via web search. **(ii) Relation Fan-out & Hub Construction:** We coin multiple distinctive entities related to the anchor. The anchor is hidden in the final question and replaced with intersecting search clues. The agent is strictly forced to resolve these clues on the web to identify the anchor before it can formulate the correct SQL query.

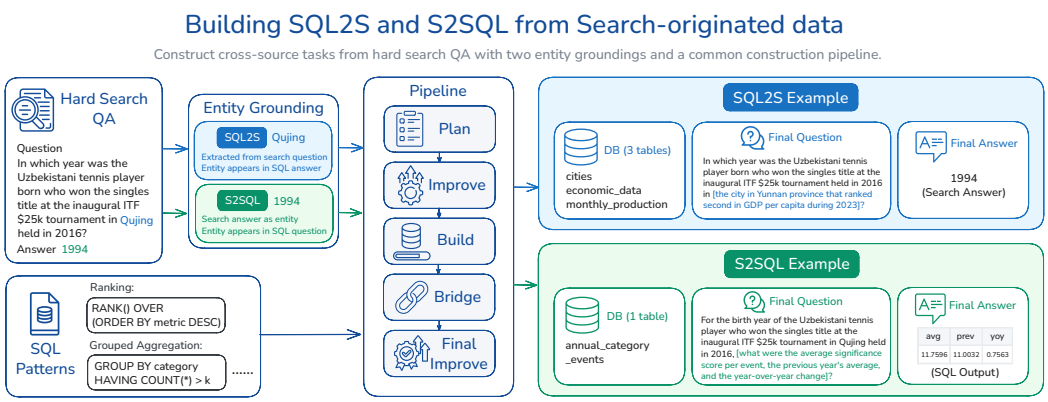


Figure 3: The search-originated pipeline. One hard-search item yields SQL2S or S2SQL according to whether the bridge is taken from the question or the answer. Both directions share the same stages: plan, improve, build, bridge, and a final improve, and draw SQL difficulty from a catalog of executable patterns.

3.3 Search-Originated Trajectories

We construct hybrid tasks from hard-search seeds by adding a synthetic text-to-SQL component around a seed entity (Figure 3, Appendix D). The original seed supplies the web-retrieval requirement; the constructed database supplies the independently nontrivial SQL requirement. A single search item can support both directions (SQL2S, S2SQL) depending on which text span is selected as the bridge.

SQL2S: The bridge is an entity mentioned in the original search question (e.g., *Qujing* in Figure 3). We formulate a SQL task where this entity is the answer. The agent is forced to query the database to discover the entity before it can complete the web search.

S2SQL: The bridge is the search answer itself (e.g., *1994*). This value becomes a required parameter that the SQL query consumes. The agent must successfully resolve the web search to find this parameter before executing the database query.

Construction Pipeline. Both directions run through a multi-stage generation pipeline to ensure the fabricated database requires genuine structured reasoning, preventing it from degrading into a trivial lookup table: **(i) Plan:** We select advanced SQL techniques from a catalog of executable patterns (e.g., grouped aggregation, ranking) and design a schema containing only raw observational data (e.g., individual event records rather than precomputed yearly stats). **(ii) Improve:** The schema is refined to strictly eliminate any columns that might pre-aggregate the answer, logically guaranteeing the gold query cannot be simplified by shortcut logic. **(iii) Build:** The planned schema is materialized into a populated SQLite database. To prevent trivial edge-case filtering, we dynamically generate same-kind peer entities and bury the target entity deeply within the data distribution. **(iv) Bridge & Final Improve:** The original search clues and the newly fabricated SQL constraints are merged into a single cohesive prompt. The bridge entity is completely masked, and scope hints are added only to resolve genuine ambiguities.

Load-Bearing Verification. To guarantee interdependence, both hops must pass strict execution checks. For SQL2S, the bridge must never appear in the merged question text. For S2SQL, the bridge must not leak into the final SQL result table. Crucially, the fabricated database must present a structurally fresh query rather than merely mirroring the search clues; otherwise, the SQL tool acts as a trivial lookup cache for a web puzzle the agent has already solved.

4 TRAINING RECIPE

4.1 SUPERVISED FINE-TUNING

Trajectory generation. Our SFT mixture combines directly used training data (Appendix B) with judge-correct trajectories generated by running the agent harness (Appendix G) against the local corpus (Appendix F) and databases. We use either Qwen3.5-397B-A17B-FP8 (Qwen Team, 2026a) or DeepSeek-V4-Flash (DeepSeek-AI, 2026) for generating trajectories. We set temperature 0.6, top-p 0.95, top-k 20, thinking on, 3600 s timeout, with $32k$ max tokens set to SQL-originated tasks and $128k$ for search-originated tasks. Max turns range from 100 to 300 depending on the task.

4.2 REINFORCEMENT LEARNING

Data Curation. We collect 21K synthesized hybrid tasks that remained unsolved during the SFT stage as candidates for RL. Because RL is highly sensitive to noisy data (Ye et al., 2026; Tang et al., 2026), we aggressively filter this pool down to 0.7K high-signal QA pairs using a strict three-stage pipeline.

(i) Difficulty filtering: we execute preliminary rollouts and discard tasks that are either consistently solved (too easy) or never solved (too hard or unsolvable), ensuring the dataset targets the exact frontier of the model's capabilities.

(ii) Quality filtering: a data-quality agent removes QA pairs with ambiguous answers, duplicate search chains, or inconsistent cross-source evidence. We actively prioritize questions requiring broad multi-hop retrieval or complex SQL operations.

(iii) Distribution balancing: to prevent policy collapse on a single source or task family, we strictly balance the final 0.7K dataset 1:1 between SQL-originated and search-originated tasks, and 1:1 between the `S2SQL` and `SQL2S` directions.

RL recipe. Our RL training is based on CISPO (MiniMax et al., 2025), with several techniques introduced to improve training efficiency and stability. Further details are provided in Appendix J.1.

Hard-gating: As invalid tool interactions and malformed tool calls will introduce noisy or substantial variance training signals into policy optimization, we applied strict hard-gating rules to increase training and tool-use efficiency: A rollout is terminated early and assigned zero reward when it exhausts the turn budget without making a tool call, contains malformed tool arguments, invokes nonexistent tools, fails to produce a valid final answer, or repeatedly issues duplicated tool queries.

Judge majority-voting: RL is highly sensitive to noisy reward signals. Relying on a single LLM judge frequently produces unreliable evaluations, particularly when correctness requires nuanced semantic matching (e.g., distinguishing *Charles III* from *Charles Emmanuel III*) or recognizing semantically equivalent SQL solutions that differ in surface-level table form. To stabilize optimization, both `S2SQL` and `SQL2S` use an LLM-as-a-judge with a five-vote majority to determine the binary pass/fail reward. An empirical analysis of this voting mechanism is presented in §6.3.

5 EVALUATION

Benchmarks and metrics. We evaluate hybrid reasoning on the 120-question hard subset of HybridDeepResearch (Wu et al., 2026b), evenly split across Search-to-SQL (S2SQL), SQL-to-Search (SQL2S), and Parallel tasks. To measure whether hybrid training preserves the underlying component capabilities, we measure single-source retention on the 1,266 BrowseComp (Wei et al., 2025) full set and 830 queries of BrowseComp-Plus (Chen et al., 2026) full set for web search, and the 135 public subset of Spider 2.0-lite (Lei et al., 2025) for text-to-SQL. BrowseComp is evaluated against the live web, whereas BrowseComp-Plus uses its fixed corpus of 100,195 documents, providing complementary evaluation under dynamic and static retrieval environments. None of these benchmarks appear in our training data. Additional benchmark and evaluation details are provided in Appendix H.

Protocol & Environment. For hybrid and search-only tasks, we evaluate the agent's final answers using a Qwen3.8-27B (Qwen Team, 2026b) LLM judge with a 5-vote majority to ensure reward

Table 1: Hybrid training progression and comparison with larger models (Pass@8 and Avg@8, %).

	Pass@8				Avg@8			
Model	**Overall**	**S2SQL**	**SQL2S**	**Parallel**	**Overall**	**S2SQL**	**SQL2S**	**Parallel**
Qwen3.5-397B-A17B-FP8	52.50	52.50	32.50	72.50	26.77	32.50	13.75	34.06
GLM-5.2-FP4	63.33	65.00	47.50	77.50	**38.54**	32.19	27.50	55.94
Claude Sonnet 4.6	**65.83**	57.50	52.50	87.50	37.29	30.31	36.25	45.31
Qwen3.5-4B	30.83	22.50	20.00	50.00	7.71	5.00	3.44	14.69
+ Hybrid SFT	59.17	65.00	**35.00**	77.50	23.75	26.88	8.75	35.83
+ Hybrid RL	**60.83**	**70.00**	30.00	**82.50**	**27.19**	**31.56**	**12.81**	**37.19**

Table 2: Single-source performance before and after hybrid training (%): search-only on BrowseComp and BrowseComp-Plus, SQL-only on the held-out Spider 2.0-lite subset (execution accuracy).

	BrowseComp		BrowseComp-Plus		Spider 2.0-lite	
Model	**Pass@8**	**Avg@8**	**Pass@8**	**Avg@8**	**Pass@8**	**Avg@8**
Qwen3.5-4B	54.03	29.57	60.84	36.49	59.26	30.65
+ Hybrid SFT	61.37	36.79	69.76	46.36	**65.19**	**41.39**
+ Hybrid RL	**62.24**	**38.90**	**73.13**	**50.56**	62.96	40.83

reliability (see §6.3). Spider 2.0-lite relies on its official execution-accuracy metric. Models interact with the SQL and retrieval environments through our custom agent harness (Appendix G). A comparison with other agent harnesses is in Appendix G.3. Unless a benchmark specifies its own retrieval setting, the web tools are served from the same local corpus used in training (Appendix F). BrowseComp-Plus (Chen et al., 2026) restricts retrieval to a fixed 100K-document corpus, while standard BrowseComp (Wei et al., 2025) operates against the live web. Unless noted otherwise, we report avg@8 and pass@8.

5.1 RESULTS

Hybrid training progressively unlocks cross-source reasoning. As shown in Table 1, the Qwen3.5-4B base model fundamentally struggles with cross-source tasks, achieving only 30.83% overall pass@8. Our hybrid SFT framework successfully bridges this gap, nearly doubling the performance to 59.17%. Applying RL yields further gains, particularly in the avg@8 metric (increasing from 23.75% to 27.19% overall). This improvement highlights that RL makes the model more confident and consistent in arriving at the correct answer, primarily by strengthening the strictly interdependent directional tasks (e.g., S2SQL avg@8 increases from 26.88% to 31.56%). Remarkably, despite its small size, our final 4B agent approaches the performance of substantially larger frontier models like Claude Sonnet 4.6 (65.83% pass@8), GLM-5.2-FP4 (GLM-5-Team et al., 2026) (63.33% pass@8) and decisively surpasses its own family's flagship, Qwen3.5-397B-A17B-FP8 (52.50% pass@8 and 26.77% avg@8), on both metrics.

Hybrid training preserves and improves single-source capabilities. Fine-tuning a compact model on complex new behaviors frequently risks catastrophic forgetting of component skills. However, Table 2 demonstrates that HYBRIDRESEARCHER-4B entirely avoids this tradeoff. Instead of cannibalizing existing knowledge, hybrid SFT actually creates a synergistic effect, substantially improving the base model's standalone capabilities. On the search side, BrowseComp pass@8 increases from 54.03% to 61.37%, and BrowseComp-Plus from 60.84% to 69.76%. On the SQL side, Spider 2.0-lite execution accuracy jumps from 59.26% to 65.19%. Applying hybrid RL largely preserves these single-source improvements (e.g., further boosting BrowseComp-Plus to 73.13% pass@8). This confirms that explicitly teaching the cross-source bridge reinforces, rather than replaces, the agent's core reasoning foundation.

Table 3: Search behavior on the evaluation set (960 rollouts per model; S2SQL and SQL2S rows use their 320). SQL2S accuracy is conditioned on the gold answer appearing in a search result ($n = 43/59/79$ rollouts) or anywhere on the web side ($n = 85/130/147$). Δ: relative change SFT→RL.

	Base	SFT	RL	Δ
Search and reading				
Web searches / rollout	38.6	22.1	18.7	−15%
Page visits / rollout	7.0	9.9	14.3	+45%
Page visits / search	0.29	0.50	0.88	+76%
Evidence				
S2SQL: bridge surfaced (%)	69.7	70.9	77.8	+10%
SQL2S: acc. | gold surfaced	12.9	18.5	27.2	+47%

Table 4: Search queries and efficiency of the SFT and RL models (960 rollouts each). Near repeats measure repeats of an earlier query (Jaccard $>= 0.8$). Δ as in Table 3.

	SFT	RL	Δ
Queries			
Words / query	6.0	6.5	+9%
Characters / query	46.9	52.3	+11%
With a quoted phrase (“ ”)	41.4	48.8	+18%
With operator (site:, OR, -term)	0.6	1.2	+126%
Exact repeats / rollout	0.63	0.23	−63%
Near repeats / rollout	4.09	2.77	−32%
Efficiency			
Turns / rollout	44.7	42.0	−6%
Tool calls / turn	1.28	1.39	+9%
Output tokens / rollout (K)	14.8	14.1	−5%

6 Analysis

6.1 Behavioral Shifts: How SFT and RL Transform Tool Use

To understand how our training recipe builds hybrid capability, we analyze the agent's search behavior on the evaluation set (Tables 3 and 4).

SFT teaches structured exploration. The base model exhibits a naive search policy, firing off many queries (38.6 per rollout) but rarely reading the results (0.29 page visits per search). SFT corrects this imbalance. The agent learns to systematically explore retrieved content, nearly halving the number of web searches while significantly increasing page visits. This transition from blind searching to active reading improves basic evidence extraction, raising SQL2S accuracy (when the gold answer is present) from 12.9% to 18.5%.

RL optimizes for precision and evidence utilization. While SFT teaches the basic mechanics, RL concentrates the policy on its successful behaviors. After RL, the agent's search behavior becomes highly deliberate: queries become longer, more specific (48.8% use quotes, and operator use doubles), and far less redundant (exact repeats drop by 63%). Consequently, the agent extracts evidence effectively; on SQL2S, the agent's ability to capitalize on surfaced gold answers jumps from 18.5% to 27.2%. Crucially, RL achieves these gains without inflating the trajectory cost, maintaining similar or less turn counts and token usage to the SFT checkpoint.

Remaining Bottlenecks. Despite these improvements, cross-source reasoning remains bottlenecked by retrieval limits and early stopping. On S2SQL, the required bridge entity still fails to surface in 22.2% of rollouts. Moreover, 82.0% of RL's wrong answers (vs. 48.7% of its correct ones) cites a time budget, the effort already spent, or a guess (Appendix I.2), suggesting they commit under self-acknowledged uncertainty. Making the decision to search more when uncertain and refraining from guessing could be a promising next direction.

6.2 Data Quality Outweighs Scale in RL Optimization

To demonstrate that RL benefits more from high-signal data than sheer volume, we compare our curated RL dataset against the full 21K candidate pool using the identical training recipe. Despite being a fraction of the size, the curated dataset achieves a 27.2% overall avg@8, substantially outperforming the massive unfiltered pool (21.0%). This performance gap is driven by training efficiency: in the curated dataset, 71% of rollout groups provide sufficient reward contrast to contribute to policy updates, compared to just 38% in the uncurated pool. This confirms that aggressive filtering prevents noisy or uninformative trajectories from diluting the RL optimization process, making data curation a much stronger lever than scaling data volume.

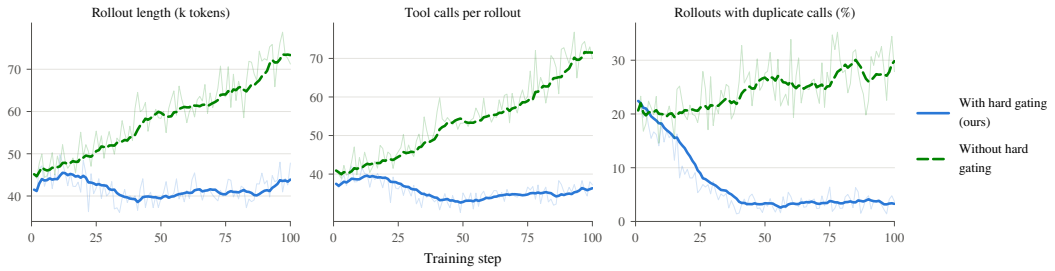


Figure 4: Training rollout length (left), tool calls per rollout (middle) and the share of rollouts with duplicated tool calls (right), with and without hard gating. Hard gating helps suppress redundant tool calls and stabilize rollout length.

6.3 JUDGE MAJORITY-VOTING HELPS IMPROVE REWARD RELIABILITY

To quantify the impact of the majority judging ensemble introduced in Section 4.2, we compare it against a single-vote baseline. The five-vote mechanism improves the model's overall avg@8 from 25.2% to 27.2%. Telemetry from our training rollouts reveals exactly why this stabilization is necessary: a single judge's verdict disagrees with the final consensus in 8.2% of cases. This instability is heavily concentrated in the `S2SQL` direction (12.2% disagreement, compared to just 1.2% for `SQL2S`), confirming that evaluating executable logic combined with complex web entities is highly variance-prone. By smoothing out these individual judge errors (examples in Appendix J.2), majority voting successfully prevents noisy rewards from degrading the policy.

6.4 HARD-GATING YIELDS CLEANER TRAINING SIGNAL AND EFFICIENT TOOL-USE

Hard-gating termination substantially improves both performance and rollout efficiency. Removing the hard-gating rules lowers avg@8 from 27.2% to 25.6%. We attribute this degradation to a positive feedback loop in tool use: without hard gating, redundant or invalid actions are insufficiently penalized, causing the policy to increasingly favor prolonged trajectories and redundant tool calls. As shown in Figure 4, both trajectory token length and tool-call counts increase throughout training without hard gating, while remaining stable with hard gating. This trend also carries over to evaluation: hard gating reduces average tool calls from 76 to 55 per question and repeated-call frequency from 49% to 11%. The fraction of questions failing to produce a valid answer also drops from 10.7% to 0.0%. Overall, hard gating suppresses inefficient tool use and provides cleaner training signals, leading to more efficient and reliable tool use, consistent with the observations in §6.1.

7 CONCLUSION

We studied how to train agents for questions that require evidence from both a structured database and unstructured web text, where a result obtained from one source must be carried into the other. Training on existing single-source data does not produce this capability, and synthesizing hybrid data naively invites shortcuts. We therefore build training data by extending validated text-to-SQL and deep-search tasks across the boundary, keep only instances that construction gates confirm require both sources, and train on them with supervised fine-tuning followed by reinforcement learning. HYBRIDRESEARCHER-4B improves over its base model by 30.00 points pass@8 on HybridDeepResearch and outperforms Qwen3.5-397B-A17B, while also improving on single-source SQL and search benchmarks. Our analysis shows that fine-tuning teaches the agent to read what it retrieves instead of issuing many searches, and that reinforcement learning benefits more from a small curated task set than from a larger unfiltered pool. More broadly, this work provides a complete training recipe for agents that reason across heterogeneous sources, and we hope it can be carried over to real-world settings that combine other kinds of sources, such as internal documents, code repositories, or domain-specific databases.

AI USE STATEMENT

Required disclosures. Generative AI is part of our method, so we disclose several required uses. (i) *Generating synthetic datasets.* The training data of §3 were synthesized with LLMs, but every item starts from a real seed (a text-to-SQL query or a web-search question). SFT trajectories were sampled from teacher models. (ii) *Cleaning and filtering datasets.* LLM judges scored candidate entities, checked data quality, and ran the model-based verification gates, alongside deterministic, non-LLM gates. (iii) *Evaluation.* HybridDeepResearch, BrowseComp, and BrowseComp-Plus answers were graded by an LLM judge (Qwen3.8-27B, majority of five votes); Spider 2.0-lite uses its deterministic execution-accuracy metric.

Recommended disclosures. We used generative AI tools to polish the paper for readability and to brainstorm figure ideas; every figure in this version was drawn and edited by the authors.

Not used. We did not use generative AI to formulate mathematical claims, write proofs, or translate. Surveys, interviews, and transcription are not applicable to this work.

Verification. We reviewed all AI-assisted work. Synthetic items pass deterministic gates, and the authors inspected samples of every shipped subset. Every reported number was checked against the training logs, evaluation outputs, or data files it came from.

We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REFERENCES

Amirhossein Abaskohi, Tianyi Chen, Miguel Muñoz-Mármol, Curtis Fox, Amrutha Varshini Ramesh, Étienne Marcotte, Xing Han Lù, Nicolas Chapados, Spandana Gella, Peter West, Giuseppe Carenini, Christopher Pal, Alexandre Drouin, and Issam H. Laradji. DRBench: A realistic benchmark for enterprise deep research. In *The Fourteenth International Conference on Learning Representations*, 2026.

Oren Etzioni Ana-Maria Popescu and Henry Kautz. Towards a theory of natural language interfaces to databases. In *Proceedings of the 8th International Conference on Intelligent User Interfaces*, 2003.

Anthropic. Introducing Claude Sonnet 4.6, 2026. URL `https://www.anthropic.com/news/claude-sonnet-4-6`.

BIRD Team. Livesqlbench: A dynamic and contamination-free benchmark for evaluating llms on real-world text-to-sql tasks. https://github.com/bird-bench/livesqlbench, 2025. Accessed: 2025-05-22.

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*, 2020.

Wenhu Chen, Ming-Wei Chang, Eva Schlinger, William Yang Wang, and William W. Cohen. Open question answering over tables and text. In *International Conference on Learning Representations*, 2021a.

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*, 2021b.

Zijian Chen, Xueguang Ma, Shengyao Zhuang, Ping Nie, Kai Zou, Sahel Sharifymoghaddam, Andrew Liu, Joshua Green, Kshama Patel, Ruoxi Meng, Mingyi Su, Yanxi Li, Haoran Hong, Xinyu Shi, Xuye Liu, Hosna Oyarhoseini, Nandan Thakur, Crystina Zhang, Luyu Gao, Wenhu Chen, and Jimmy Lin. BrowseComp-plus: A fair and disentangled evaluation benchmark for deep search agents. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2026.

Zheng Chu, Xiao Wang, Jack Hong, Huiming Fan, Yuqi Huang, Yue Yang, Guohai Xu, Chenxiao Zhao, Cheng Xiang, Shengchao Hu, Dongdong Kuang, Ming Liu, Bing Qin, and Xing Yu. Redsearcher: A scalable and cost-efficient framework for long-horizon search agents. *arXiv preprint arXiv:2602.14234*, 2026.

Florin Cuconasu, Giovanni Trappolini, Federico Siciliano, Simone Filice, Cesare Campagnano, Yoelle Maarek, Nicola Tonellotto, and Fabrizio Silvestri. The power of noise: Redefining retrieval for RAG systems. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2024.

Zhuyun Dai and Jamie Callan. Deeper text understanding for ir with contextual neural language modeling. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2019.

Michael Brown William Fisher Kate Hunicke-Smith David Pallett Christine Pao Alexander Rudnicky Deborah A. Dahl, Madeleine Bates and Elizabeth Shriber. Expanding the scope of the ATIS task: The ATIS-3 corpus. *Proceedings of the workshop on Human Language Technology*, 1994.

DeepSeek-AI. Deepseek-v4: Towards highly efficient million-token context intelligence, 2026.

Minghang Deng, Ashwin Ramachandran, Canwen Xu, Lanxiang Hu, Zhewei Yao, Anupam Datta, and Hao Zhang. Reforce: A text-to-sql agent with self-refinement, consensus enforcement, and column exploration. *arXiv preprint arXiv:2502.00675*, 2025.

Yuwen Du, Rui Ye, Shuo Tang, Xinyu Zhu, Yijun Lu, Yuzhu Cai, and Siheng Chen. Openseeker: Democratizing frontier search agents by fully open-sourcing training data. In *Workshop on Failure Modes of Agentic AI at ICML 2026*, 2026.

Catherine Finegan-Dollak, Jonathan K. Kummerfeld, Li Zhang, Karthik Ramanathan, Sesh Sadasivam, Rui Zhang, and Dragomir Radev. Improving text-to-SQL evaluation methodology. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018.

Jiaxuan Gao, Wei Fu, Minyang Xie, Shusheng Xu, Chuyi He, Zhiyu Mei, Banghua Zhu, and Yi Wu. Beyond ten turns: Unlocking long-horizon agentic search with large-scale asynchronous RL. *arXiv preprint arXiv:2508.07976*, 2025.

Alessandra Giordani and Alessandro Moschitti. Automatic generation and reranking of sql-derived answers to nl questions. In *Proceedings of the Second International Conference on Trustworthy Eternal Systems via Evolving Software, Data and Knowledge*, 2012.

GLM-5-Team, :, Aohan Zeng, Xin Lv, Zhenyu Hou, Zhengxiao Du, Qinkai Zheng, Bin Chen, Da Yin, Chendi Ge, Chenghua Huang, Chengxing Xie, Chenzheng Zhu, Congfeng Yin, Cunxiang Wang, Gengzheng Pan, Hao Zeng, Haoke Zhang, Haoran Wang, Huilong Chen, Jiajie Zhang, Jian Jiao, Jiaqi Guo, Jingsen Wang, Jingzhao Du, Jinzhu Wu, Kedong Wang, Lei Li, Lin Fan, Lucen Zhong, Mingdao Liu, Mingming Zhao, Pengfan Du, Qian Dong, Rui Lu, Shuang-Li, Shulin Cao, Song Liu, Ting Jiang, Xiaodong Chen, Xiaohan Zhang, Xuancheng Huang, Xuezhen Dong, Yabo Xu, Yao Wei, Yifan An, Yilin Niu, Yitong Zhu, Yuanhao Wen, Yukuo Cen, Yushi Bai, Zhongpei Qiao, Zihan Wang, Zikang Wang, Zilin Zhu, Ziqiang Liu, Zixuan Li, Bojie Wang, Bosi Wen, Can Huang, Changpeng Cai, Chao Yu, Chen Li, Chengwei Hu, Chenhui Zhang, Dan Zhang, Daoyan Lin, Dayong Yang, Di Wang, Ding Ai, Erle Zhu, Fangzhou Yi, Feiyu Chen, Guohong Wen, Hailong Sun, Haisha Zhao, Haiyi Hu, Hanchen Zhang, Hanrui Liu, Hanyu Zhang, Hao Peng, Hao Tai, Haobo Zhang, He Liu, Hongwei Wang, Hongxi Yan, Hongyu Ge, Huan Liu, Huanpeng Chu, Jia'ni Zhao, Jiachen Wang, Jiajing Zhao, Jiamin Ren, Jiapeng Wang, Jiaxin Zhang, Jiayi Gui, Jiayue Zhao, Jijie Li, Jing An, Jing Li, Jingwei Yuan, Jinhua Du, Jinxin Liu, Junkai Zhi, Junwen Duan, Kaiyue Zhou, Kangjian Wei, Ke Wang, Keyun Luo, Laiqiang Zhang, Leigang Sha, Liang Xu, Lindong Wu, Lintao Ding, Lu Chen, Minghao Li, Nianyi Lin, Pan Ta, Qiang Zou, Rongjun Song, Ruiqi Yang, Shangqing Tu, Shangtong Yang, Shaoxiang Wu, Shengyan Zhang, Shijie Li, Shuang Li, Shuyi Fan, Wei Qin, Wei Tian, Weining Zhang, Wenbo Yu, Wenjie Liang, Xiang Kuang, Xiangmeng Cheng, Xiangyang Li, Xiaoquan Yan, Xiaowei Hu, Xiaoying Ling, Xing Fan,

Xingye Xia, Xinyuan Zhang, Xinze Zhang, Xirui Pan, Xu Zou, Xunkai Zhang, Yadi Liu, Yandong Wu, Yanfu Li, Yidong Wang, Yifan Zhu, Yijun Tan, Yilin Zhou, Yiming Pan, Ying Zhang, Yinpei Su, Yipeng Geng, Yong Yan, Yonglin Tan, Yuean Bi, Yuhan Shen, Yuhao Yang, Yujiang Li, Yunan Liu, Yunqing Wang, Yuntao Li, Yurong Wu, Yutao Zhang, Yuxi Duan, Yuxuan Zhang, Zezhen Liu, Zhengtao Jiang, Zhenhe Yan, Zheyu Zhang, Zhixiang Wei, Zhuo Chen, Zhuoer Feng, Zijun Yao, Ziwei Chai, Ziyuan Wang, Zuzhou Zhang, Bin Xu, Minlie Huang, Hongning Wang, Juanzi Li, Yuxiao Dong, and Jie Tang. Glm-5: from vibe coding to agentic engineering, 2026. URL https://arxiv.org/abs/2602.15763.

Yunhai Hu, Junwei Zhou, Yumo Cao, Yitao Long, Yiwei Xu, Qiyi Jiang, Weiyao Wang, Xiaoyu Cao, Zhen Sun, Yiran Zou, and Nan Du. Dsl-r1: From sql to dsl for training retrieval agents across structured and unstructured data with reinforcement learning. *arXiv preprint arXiv:2603.21018*, 2026.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan O Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training LLMs to reason and leverage search engines with reinforcement learning. In *Second Conference on Language Modeling*, 2025.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, 2023.

Chia-Hsuan Lee, Oleksandr Polozov, and Matthew Richardson. KaggleDBQA: Realistic evaluation of text-to-SQL parsers. 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)*, 2021.

Gyubok Lee, Hyeonji Hwang, Seongsu Bae, Yeonsu Kwon, Woncheol Shin, Seongjun Yang, Minjoon Seo, Jong-Yeup Kim, and Edward Choi. EHRSQL: A practical text-to-SQL benchmark for electronic health records. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022.

Fangyu Lei, Jixuan Chen, Yuxiao Ye, Ruisheng Cao, Dongchan Shin, Hongjin SU, ZHAOQING SUO, Hongcheng Gao, Wenjing Hu, Pengcheng Yin, Victor Zhong, Caiming Xiong, Ruoxi Sun, Qian Liu, Sida Wang, and Tao Yu. Spider 2.0: Evaluating language models on real-world enterprise text-to-SQL workflows. In *The Thirteenth International Conference on Learning Representations*, 2025.

Boyan Li, Zhuowen Liang, Yupeng Xie, Xiaotian Lin, Tianqi Luo, Xinyu Liu, Yizhang Zhu, Zhangyang Peng, Yuan Li, Zhengxuan Zhang, Jiayi Zhang, Nan Tang, Guoliang Li, and Yuyu Luo. Dataspace: Benchmarking data agents for verifiable analytics over heterogeneous workspaces. *arXiv preprint arXiv:2608.03451*, 2026.

Haoyang Li, Shang Wu, Xiaokang Zhang, Xinmei Huang, Jing Zhang, Fuxin Jiang, Shuai Wang, Tieying Zhang, Jianjun Chen, Rui Shi, Hong Chen, and Cuiping Li. Omnisql: Synthesizing high-quality text-to-sql data at scale. *Proc. VLDB Endow.*, 18(11):4695–4709, 2025a. URL https://doi.org/10.14778/3749646.3749723.

Jinyang Li, Binyuan Hui, GE QU, Jiaxi Yang, Binhua Li, Bowen Li, Bailin Wang, Bowen Qin, Ruiying Geng, Nan Huo, Xuanhe Zhou, Chenhao Ma, Guoliang Li, Kevin Chang, Fei Huang, Reynold Cheng, and Yongbin Li. Can LLM already serve as a database interface? a BIg bench for large-scale database grounded text-to-SQLs. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023.

Kuan Li, Zhongwang Zhang, Huifeng Yin, Liwen Zhang, Litu Ou, Jialong Wu, Wenbiao Yin, Baixuan Li, Zhengwei Tao, Xinyu Wang, Weizhou Shen, Junkai Zhang, Dingchu Zhang, Xixi Wu, Yong Jiang, Ming Yan, Pengjun Xie, Fei Huang, and Jingren Zhou. Websailor: Navigating superhuman reasoning for web agent. *arXiv preprint arXiv:2507.02592*, 2025b.

Shicheng Liu, Jialiang Xu, Wesley Tjangnaka, Sina Semnani, Chen Yu, and Monica Lam. SUQL: Conversational search over structured and unstructured data with large language models. In *Findings of the Association for Computational Linguistics: NAACL 2024*, 2024.

Shu Liu, Alan Zhu, Sumanth Hegde, Shiyi Cao, Shuo Yuan, Samion Suwito, Tyler Griggs, Matei Zaharia, Joseph E. Gonzalez, and Ion Stoica. SkyRL-SQL: Multi-turn SQL data agents via RL. In *First Workshop on Multi-Turn Interactions in Large Language Models*, 2025.

Liutao, Xutao Mao, Dixuan Zhang, Yifan Li, LiuHaixin, KongLulu, Jiaming Hou, Rui Li, YunLong Li, Aoze Zheng, Zhiqiang Zhang, Luo Zhewei, Hongying Zan, Kunli Zhang, and Min Peng. Logiccat: A chain-of-thought text-to-sql benchmark for complex reasoning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2026.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Rui Lu, Zhenyu Hou, Zihan Wang, Hanchen Zhang, Xiao Liu, Yujiang Li, Shi Feng, Jie Tang, and Yuxiao Dong. Deepdive: Advancing deep search agents with knowledge graphs and multi-turn rl. *arXiv preprint arXiv:2509.10446*, 2025.

Yun Luo, Zhen Yang, Fandong Meng, Yafu Li, Jie Zhou, and Yue Zhang. An empirical study of catastrophic forgetting in large language models during continual fine-tuning. *IEEE Transactions on Audio, Speech and Language Processing*, 2025.

Peixian Ma, Xialie Zhuang, Chengjin Xu, Xuhui Jiang, Ran Chen, and Jian Guo. Sql-r1: Training natural language to sql reasoning model by reinforcement learning. In *Advances in Neural Information Processing Systems*, 2025.

Ruiying Ma, Shreya Shankar, Ruiqi Chen, Yiming Lin, Sepanta Zeighami, Rajoshi Ghosh, Abhinav Gupta, Anushrut Gupta, Tanmai Gopal, and Aditya G. Parameswaran. Can ai agents answer your data questions? a benchmark for data agents. *arXiv preprint arXiv:2603.20576*, 2026.

MiniMax, :, Aili Chen, Aonian Li, Bangwei Gong, Binyang Jiang, Bo Fei, Bo Yang, Boji Shan, Changqing Yu, Chao Wang, Cheng Zhu, Chengjun Xiao, Chengyu Du, Chi Zhang, Chu Qiao, Chunhao Zhang, Chunhui Du, Congchao Guo, Da Chen, Deming Ding, Dianjun Sun, Dong Li, Enwei Jiao, Haigang Zhou, Haimo Zhang, Han Ding, Haohai Sun, Haoyu Feng, Huaiguang Cai, Haichao Zhu, Jian Sun, Jiaqi Zhuang, Jiaren Cai, Jiayuan Song, Jin Zhu, Jingyang Li, Jinhao Tian, Jinli Liu, Junhao Xu, Junjie Yan, Junteng Liu, Junxian He, Kaiyi Feng, Ke Yang, Kecheng Xiao, Le Han, Leyang Wang, Lianfei Yu, Liheng Feng, Lin Li, Lin Zheng, Linge Du, Lingyu Yang, Lunbin Zeng, Minghui Yu, Mingliang Tao, Mingyuan Chi, Mozhi Zhang, Mujie Lin, Nan Hu, Nongyu Di, Peng Gao, Pengfei Li, Pengyu Zhao, Qibing Ren, Qidi Xu, Qile Li, Qin Wang, Rong Tian, Ruitao Leng, Shaoxiang Chen, Shaoyu Chen, Shengmin Shi, Shitong Weng, Shuchang Guan, Shuqi Yu, Sichen Li, Songquan Zhu, Tengfei Li, Tianchi Cai, Tianrun Liang, Weiyu Cheng, Weize Kong, Wenkai Li, Xiancai Chen, Xiangjun Song, Xiao Luo, Xiao Su, Xiaobo Li, Xiaodong Han, Xinzhu Hou, Xuan Lu, Xun Zou, Xuyang Shen, Yan Gong, Yan Ma, Yang Wang, Yiqi Shi, Yiran Zhong, Yonghong Duan, Yongxiang Fu, Yongyi Hu, Yu Gao, Yuanxiang Fan, Yufeng Yang, Yuhao Li, Yulin Hu, Yunan Huang, Yunji Li, Yunzhi Xu, Yuxin Mao, Yuxuan Shi, Yuze Wenren, Zehan Li, Zelin Li, Zhanxu Tian, Zhengmao Zhu, Zhenhua Fan, Zhenzhen Wu, Zhichao Xu, Zhihang Yu, Zhiheng Lyu, Zhuo Jiang, Zibo Gao, Zijia Wu, Zijian Song, and Zijun Sun. Minimax-m1: Scaling test-time compute efficiently with lightning attention, 2025. URL `https://arxiv.org/abs/2506.13585`.

Isil Dillig Navid Yaghmazadeh, Yuepeng Wang and Thomas Dillig. Sqlizer: Query synthesis from natural language. In *International Conference on Object-Oriented Programming, Systems, Languages, and Applications, ACM*, 2017.

OpenAI. GPT-5 System Card, 2025. URL `https://openai.com/index/gpt-5-system-card/`.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. The fineweb datasets: Decanting the web for the finest text data at scale. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024.

Thinh Pham, Nguyen Nguyen, Pratibha Zunjare, Weiyuan Chen, Yu-Min Tseng, and Tu Vu. SealQA: Raising the bar for reasoning in search-augmented language models. In *International Conference on Learning Representations (ICLR)*, 2026.

Dzmitry Pihulski, Karol Charchut, Viktoria Novogrodskaia, and Jan Kocoń. Llmsql: Upgrading wikisql for the llm era of text-to-sql. In *2025 IEEE International Conference on Data Mining Workshops (ICDMW)*, 2025.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026a. URL https://qwen.ai/blog?id=qwen3.5.

Qwen Team. Qwen3.8-Max: A new bar for coding and cowork, August 2026b. URL https://qwen.ai/blog?id=qwen3.8.

Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.*, 2009.

Aymeric Roucher, Albert Villanova del Moral, Thomas Wolf, Leandro von Werra, and Erik Kaunismäki. smolagents: a smol library to build great agentic systems. https://github.com/huggingface/smolagents, 2025.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Mika Senghaas, Fares Obeid, Sami Jaghouar, William Brown, Jack Min Ong, Andrew Baker, Justus Mattern, Daniel Auras, Jannik Straube, Manveer Basra, Aiman Ismail, and Johannes Hagemann. PRIME-RL: Async & decentralized RL training at scale. In *NeurIPS Workshop on GPU-Accelerated and Scalable Optimization*, 2025.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed Chi, Nathanael Schärli, and Denny Zhou. Large language models can be easily distracted by irrelevant context. In *International Conference on Machine Learning (ICML)*, 2023.

Akshit Sinha, Arvindh Arun, Shashwat Goel, Steffen Staab, and Jonas Geiping. The illusion of diminishing returns: Measuring long horizon execution in LLMs. In *International Conference on Learning Representations (ICLR)*, 2026.

Huatong Song, Jinhao Jiang, Yingqian Min, Jie Chen, Zhipeng Chen, Wayne Xin Zhao, Lei Fang, and Ji-Rong Wen. R1-searcher: Incentivizing the search capability in llms via reinforcement learning. *arXiv preprint arXiv:2503.05592*, 2025.

Alvin Cheung Jayant Krishnamurthy Srinivasan Iyer, Ioannis Konstas and Luke Zettlemoyer. Learning a neural semantic parser from user feedback. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017.

Shiqian Su, Sen Xing, Xuan Dong, Muyan Zhong, Bin Wang, Xizhou Zhu, Yuntao Chen, Wenhai Wang, Yue Deng, Pengxiang Zhu, Ziyuan Liu, Tiantong Li, Jiaheng Yu, Zhe Chen, Lidong Bing, and Jifeng Dai. MiroFlow: Towards high-performance and robust open-source agent framework for general deep research tasks. *arXiv preprint arXiv:2602.22808*, 2026.

Lappoon R. Tang and Raymond J. Mooney. Automated construction of database interfaces: Intergrating statistical and relational learning for semantic parsing. In *2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, 2000.

Xinyu Tang, Zhenduo Zhang, Yurou Liu, Xin Zhao, zujie wen, Zhiqiang Zhang, and JUN ZHOU. Towards high data efficiency in reinforcement learning with verifiable reward. In *The Fourteenth International Conference on Learning Representations*, 2026.

Zhengwei Tao, Jialong Wu, Wenbiao Yin, Pu Wu, Junkai Zhang, Baixuan Li, Haiyang SHEN, Kuan Li, Liwen Zhang, Xinyu Wang, Wentao Zhang, Yong Jiang, Pengjun Xie, Fei Huang, and Jingren Zhou. Webshaper: Agentically data synthesizing via information-seeking formalization. In *The Fourteenth International Conference on Learning Representations*, 2026.

Nandan Thakur, Zijian Chen, Xueguang Ma, and Jimmy Lin. Orbit: Scalable and verifiable data generation for search agents on a tight budget. *arXiv preprint arXiv:2604.01195*, 2026.

Han Wang, Archiki Prasad, Elias Stengel-Eskin, and Mohit Bansal. AdaCAD: Adaptively decoding to balance conflicts between contextual and parametric knowledge. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, 2025a.

Han Wang, Archiki Prasad, Elias Stengel-Eskin, and Mohit Bansal. Retrieval-augmented generation with conflicting evidence. In *Second Conference on Language Modeling*, 2025b.

Han Wang, David Wan, Hyunji Lee, Thinh Pham, Mikaela Cankosyan, Weiyuan Chen, Elias Stengel-Eskin, Tu Vu, and Mohit Bansal. Merrin: A benchmark for multimodal evidence retrieval and reasoning in noisy web environments. *arXiv preprint arXiv:2604.13418*, 2026a.

Ziting Wang, Shize Zhang, Haitao Yuan, Jinwei Zhu, Wei Dong, and Gao Cong. Fdabench: A benchmark for data agents on analytical queries over heterogeneous data. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*. Association for Computing Machinery, 2026b.

Jason Wei, Nguyen Karina, Hyung Won Chung, Yunxin Joy Jiao, Spencer Papay, Amelia Glaese, John Schulman, and William Fedus. Measuring short-form factuality in large language models. *arXiv preprint arXiv:2411.04368*, 2024.

Jason Wei, Zhiqing Sun, Spencer Papay, Scott McKinney, Jeffrey Han, Isa Fulford, Hyung Won Chung, Alex Tachard Passos, William Fedus, and Amelia Glaese. Browsecomp: A simple yet challenging benchmark for browsing agents. *arXiv preprint arXiv:2504.12516*, 2025.

Jian Wu, Linyi Yang, Dongyuan Li, Yuliang Ji, Manabu Okumura, and Yue Zhang. MMQA: Evaluating LLMs with multi-table multi-hop complex questions. In *The Thirteenth International Conference on Learning Representations*, 2025.

Ruofan Wu, Boyi Liu, Fan Shu, Yite Wang, Zhewei Yao, Yuxiong He, and Feng Yan. Bridging databases and documents: Data-algorithm co-design for hybrid question answering. In *Third Conference on Language Modeling*, 2026a.

Ruofan Wu, Peiran Xu, Xiaolong Li, Fan Shu, Soyoung Yoon, Yite Wang, Xiaodong Yu, Boyi Liu, Feng Yan, Debiao Li, Yuxiong He, and Zhewei Yao. Benchmarking hybrid deep research across database querying and web search. *arXiv preprint arXiv:2609.09410*, 2026b.

Jian Xie, Kai Zhang, Jiangjie Chen, Renze Lou, and Yu Su. Adaptive chameleon or stubborn sloth: Revealing the behavior of large language models in knowledge conflicts. In *International Conference on Learning Representations (ICLR)*, 2024.

Jian Xie, Tianhe Lin, Zilu Wang, Yuting Ning, Yuekun Yao, Tianci Xue, Zhehao Zhang, Zhongyang Li, Kai Zhang, Yufan Wu, Shijie Chen, Boyu Gou, Mingzhe Han, Yifei Wang, Vint Lee, Xinpeng Wei, Xiangjun Wang, Yu Su, and Huan Sun. Quest: Training frontier deep research agents with fully synthetic tasks. *arXiv preprint arXiv:2605.24218*, 2026.

Chen Yang, Ran Le, Yun Xing, Zhenwei An, Zongchao Chen, Wayne Xin Zhao, Yang Song, and Tao Zhang. Toolmind technical report: A large-scale, reasoning-enhanced tool-use dataset. *arXiv preprint arXiv:2511.15718*, 2025.

Xiao Yang, Kai Sun, Hao Xin, Yushi Sun, Nikita Bhalla, Xiangsen Chen, Sajal Choudhary, Rongze Gui, Ziran Jiang, Ziyu JIANG, Lingkun Kong, Brian Moran, Jiaqi Wang, Yifan Ethan Xu, An Yan, Chenyu Yang, Eting Yuan, Hanwen Zha, Nan Tang, Lei Chen, Nicolas SCHEFFER, Yue Liu, Nirav Shah, Rakesh Wanga, Anuj Kumar, Wen tau Yih, and Xin Luna Dong. CRAG - comprehensive RAG benchmark. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024.

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*, 2023.

Zhewei Yao, Guoheng Sun, Łukasz Borchmann, Zheyu Shen, Minghang Deng, Bohan Zhai, Hao Zhang, Ang Li, and Yuxiong He. Arctic-Text2SQL-r1: Simple rewards, strong reasoning in text-to-SQL. In *Findings of the Association for Computational Linguistics: ACL 2026*, 2026.

Bowen Ye, Lei Li, Shicheng Li, Zihao Yue, Linghao Zhang, Hanglong Lv, Yuanxin Liu, Wenhan Ma, Hao Tian, Rang Li, Jinhao Dong, Yikai Zhao, Xiangwei Deng, Hailin Zhang, Liang Zhao, Qi Liu, Lingpeng Kong, Tong Yang, and Fuli Luo. Codemidas: Scaling agentic coding rl environments from code itself. *arXiv preprint arXiv:2609.22068*, 2026.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018.

Tao Yu, Rui Zhang, Heyang Er, Suyi Li, Eric Xue, Bo Pang, Xi Victoria Lin, Yi Chern Tan, Tianze Shi, Zihan Li, Youxuan Jiang, Michihiro Yasunaga, Sungrok Shim, Tao Chen, Alexander Fabbri, Zifan Li, Luyao Chen, Yuwen Zhang, Shreya Dixit, Vincent Zhang, Caiming Xiong, Richard Socher, Walter Lasecki, and Dragomir Radev. CoSQL: A conversational text-to-SQL challenge towards cross-domain natural language interfaces to databases. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019a.

Tao Yu, Rui Zhang, Michihiro Yasunaga, Yi Chern Tan, Xi Victoria Lin, Suyi Li, Heyang Er, Irene Li, Bo Pang, Tao Chen, Emily Ji, Shreya Dixit, David Proctor, Sungrok Shim, Jonathan Kraft, Vincent Zhang, Caiming Xiong, Richard Socher, and Dragomir Radev. SParC: Cross-domain semantic parsing in context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019b.

John M. Zelle and Raymond J. Mooney. Learning to parse database queries using inductive logic programming. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence - Volume 2*, 1996.

Chao Zhang, Yuren Mao, Yijiang Fan, Yu Mi, Yunjun Gao, Lu Chen, Dongfang Lou, and Jinshu Lin. Finsql: Model-agnostic llms-based text-to-sql framework for financial analysis. In *Companion of the 2024 International Conference on Management of Data*, 2024.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*, 2025.

Jiarui Zhao, Rongzhi Zhang, Lingchuan Liu, Hao Yang, Xunliang Cai, and Xi Su. Lohosearch: Benchmarking long-horizon search agents beyond the human difficulty ceiling. *arXiv preprint arXiv:2606.12837*, 2026.

Danna Zheng, Mirella Lapata, and Jeff Pan. Archer: A human-labeled text-to-SQL dataset with arithmetic, commonsense and hypothetical reasoning. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2024.

Yuxiang Zheng, Dayuan Fu, Xiangkun Hu, Xiaojie Cai, Lyumanshan Ye, Pengrui Lu, and Pengfei Liu. DeepResearcher: Scaling deep research via reinforcement learning in real-world environments. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, 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)*, 2021.

A Overview

This appendix provides the data-construction details, environment, evaluation protocol, and extended analyses behind the main text. Specifically:

- Appendix B lists the existing datasets we train on directly: HyDRA items and nine public text-to-SQL datasets.
- Appendix C details the SQL-originated construction: the shipped subsets, the `SQL2S` and `S2SQL` builds, corpus ingestion, and worked examples.
- Appendix D details the search-originated construction: the seed pool and pipeline, the shipped databases, and worked examples.
- Appendix E reports the SFT data ablation that adds each data family in turn.
- Appendix F describes the local retrieval corpus that serves the web tools during trajectory generation, RL, and evaluation.
- Appendix G describes the hybrid agent harness: its tool interface, episode execution, and a comparison with other agent harnesses.
- Appendix H gives the evaluation details for BrowseComp, BrowseComp-Plus, and Spider 2.0-lite, the inference configuration of every model, and the LLM judge prompts.
- Appendix I analyzes the failures of the base, SFT, and RL models: the outcome of every rollout and commits justified by time, effort, or a guess.
- Appendix J gives the RL training parameters and an analysis of judge voting.

B Direct Training Data

We first include existing datasets that are used directly for training rather than as construction seeds. Specifically, we use 10,184 HyDRA items (Wu et al., 2026a) and 25,747 deduplicated text-to-SQL examples drawn from nine public datasets: Spider 1.0 (Yu et al., 2018), BIRD (Li et al., 2023), SynSQL (Li et al., 2025a), EHRSQL (Lee et al., 2022), LLMSQL (Pihulski et al., 2025), MMQA (Wu et al., 2025), LogicCat (Liutao et al., 2026), SParC (Yu et al., 2019b), and CoSQL (Yu et al., 2019a). These examples provide direct supervision for hybrid and SQL capabilities, respectively. We use a random subset of SynSQL; for the other datasets, we keep only the questions for which at least one sampled trajectory is correct. Table 5 summarizes the composition.

Table 5: Base data: trained on directly. Qwen3.5 tokens at 128k packing.

		nine raw text-to-SQL sources (25,747 rows, 115.7M tokens)									
	HyDRA	Spider 1.0	BIRD	SynSQL	EHRSQL	LLMSQL3	MMQA	LogicCat	SParC	CoSQL	**Total**
Rows	10,184	7,791	6,231	3,360	3,107	2,587	1,942	592	130	7	**35,931**
Tokens (M)	196.1	24.9	24.3	23.4	23.7	9.3	6.1	3.5	0.5	0.04	**311.7**

C SQL-Originated Construction Details

Specifications and counts behind §3.2, organized by the construction steps of the main text. Three SQL-originated subsets ship (Table 6): SQL-ORIG-SYN and SQL-ORIG-PRIM hold `SQL2S` items (Appendix C.1), and SQL-ORIG-S2SQL-SYN holds `S2SQL` items (Appendix C.2). Construction seeds are drawn from two source families summarized in Table 7: primary datasets with real databases and named entities, and a large synthetic SQLite source.

C.1 Building `SQL2S`: the bridge is an output cell

SQL-ORIG-SYN (synthetic split) runs all four steps below; SQL-ORIG-PRIM (primary split) is then refabricated by an anti-shortcut pass. Examples 1 and 2 in Appendix C.4 walk through one item of each.

Table 6: The three SQL-originated subsets that ship. Token counts are Qwen3.5 tokens at 128k packing.

Subset	Rows	Tokens	Direction / split
SQL-ORIG-SYN	14,985	598.3M	SQL2S, synthetic
SQL-ORIG-PRIM	15,067	272.2M	SQL2S, primary
SQL-ORIG-S2SQL-SYN	6,269	272.8M	S2SQL, synthetic
total	**36,321**	**1,143.3M**	

Table 7: Seed-source for shipped SQL-originated data, and the reason for the selection.

Family	Members	Verdict
PRIMARY	BIRD (Li et al., 2023), Spider 1.0 (Yu et al., 2018), BULL/FinSQL (Zhang et al., 2024), Archer (Zheng et al., 2024), KaggleDBQA (Lee et al., 2021), Classics (Finegan-Dollak et al., 2018) (Advising, ATIS (Deborah A. Dahl & Shriber, 1994; Srinivasan Iyer & Zettlemoyer, 2017), Geography (Zelle & Mooney, 1996; Srinivasan Iyer & Zettlemoyer, 2017), Scholar (Srinivasan Iyer & Zettlemoyer, 2017), Restaurant (Tang & Mooney, 2000; Ana-Maria Popescu & Kautz, 2003; Giordani & Moschitti, 2012), IMDB (Navid Yaghmazadeh & Dillig, 2017))	real databases, real named entities; form the **primary** split
SYNTHETIC	SynSQL-2.5M (OmniSQL; Li et al., 2025a)	the only synthetic set that ships populated databases (16,583 SQLite DBs, 2.5M English rows); forms the **synthetic** split

(i) Bridge Selection: The bridge is a cell of the executed gold output that the question does not name. An LLM judge scores each candidate on rarity, referent-collision risk, and whether a credible document can mention it; candidates that are ambiguous or cannot host a fabricated document are dropped.

(ii) Hop One (Coined Facts): Each fact is built answer-first: the bridge is pinned from the gold SQL, a fact about it (a business rule, a data-dictionary definition, or a specification value) is invented, and only then is the question written. Coined names are pronounceable pseudo-words made globally unique by a reservation registry. Answers are NAME types (`coined_company_org`, `coined_team`, `coined_person`, `coined_product_model`, `coined_award`) or VALUE (`number`). Questions may also define a coined composite metric (name, abbreviation, description, and a formula faithful to the gold SQL) in a domain-knowledge block: 77.4% of SQL-ORIG-SYN and 14.0% of SQL-ORIG-PRIM questions have one. Only SQL-ORIG-SYN also serves these definitions through the knowledge-lookup tools (Table 14), since the real databases behind SQL-ORIG-PRIM have no knowledge base. Each fact gets one gold document, which wraps the answer sentence in a realistic body in a register chosen by entity type (news or biography for people, press release or filing for organizations, spec page or review for products, internal wiki for private metrics); golds average 890–924 characters, so most fit one retrieval chunk. Each fact also gets up to ten typed hard negatives, full-length perturbations of the gold's answer sentence that each flip exactly one constraint (Table 8).

(iii) Hop Two (The Intersection Hub): Given n coined facts about the bridge, hubs are minted over fact subsets: the gold hub satisfies all c constraints, each near-miss hub satisfies $c-1$, and each constraint is the missing one for some near-miss. The question refers to each fact by paraphrase, never verbatim. Each hub-affiliation gold names the hub exactly once, never in the first sentence and never as "the sole" anything, among at least two same-type

distractors, and gets seven to ten near-miss negatives that never contain the hub name. From 3,389 entities with four or five coined facts each and three or four constraints per item, the build yields 15,421 items, with 61,732 hub golds against 432,044 hub negatives (≈1:7).

(iv) Verification: Six deterministic gates run first: (a) the fact is searchable in the gold; (b) the entity is present in the gold; (c) both fall inside one ≤300-character window; (d) the answer is not in the question; (e) the answer is not a value in the seed's output column; (f) the constraint intersection is a singleton. Five model-based gates follow: gold faithfulness, a no-document parametric probe, a mask test (with the gold removed, the fact must not be recoverable from the item's negatives), negative-reader denoising (drop any negative that answers the question), and premise plausibility.

Table 8: The twelve hard-negative categories and the skill each one teaches.

Category	Flips	Skill taught
`same_entity_no_answer`	omits the fact	do not accept partial evidence
`wrong_event`	the event	event discrimination
`wrong_date`	the date	temporal pinning
`wrong_location`	the place	spatial pinning
`wrong_version_or_edition`	a superseded value	read, do not skim
`wrong_category_or_class`	the class	type discrimination
`right_attribute_wrong_entity`	the entity	attribute–entity binding
`namesake_or_misspelling`	the referent	homonym resistance
`same_entity_partial_evidence`	sufficiency	stopping criterion
`near_dup_missing_key_fact`	the decisive span	span-level reading
`topically_adjacent`	relevance	topic vs. answer
`lexical_overlap`	the referent	lexical robustness

Anti-shortcut pass (SQL-ORIG-PRIM). Primary-split bridges can be famous real entities (e.g. *Google Photos* in Example 2), which invites early lock-on and distractor capture: committing to the first plausible candidate, or to a retrieved near-miss, before every constraint is checked. Both failure modes are documented for long-horizon search agents (Sinha et al., 2026; Gao et al., 2025) and for LLMs reading noisy retrieved text (Shi et al., 2023; Cuconasu et al., 2024; Wang et al., 2025b; Pham et al., 2026), and a salient parametric prior makes lock-on worse (Xie et al., 2024). The pass keeps the seed SQL and refabricates the web puzzle with DeepSeek-V4-Flash (DeepSeek-AI, 2026), which also samples the trajectories: a parametric-prior decoy receives the longest and most authoritative document in the item (salience inversion), and a real homonym of the bridge and at least two namesake or misspelling negatives are added.

C.2 Building S2SQL: the anchor is a WHERE-clause literal

SQL-ORIG-S2SQL-SYN is built from the synthetic split only, since few primary-split seeds pass the anchor gates. Gold documents, typed negatives, and verification follow steps (ii) and (iv) of Appendix C.1; Example 3 in Appendix C.4 walks through one item.

(i) Anchor Selection: Candidates are string literals under `EQ`, `IN`, or `LIKE` nodes of the gold query's syntax tree, on a name-bearing column (e.g. name, title, team, city, product) and outside an enum stop-set such as {active, yes, male}. The execution check replaces the predicate with a tautology and requires the row set to change, which rejects co-filter decoys whose rows a join key or threshold already pins; the membership gate requires the anchor to appear in the question but not in the output. Execution must also be non-degenerate ($0 < |\text{rows}| < |\text{table}|$), and the anchor must support at least three clues, which rejects famous anchors that a single hop resolves (e.g. *Canada*). A namability judge then rejects five recurring shapes: class labels (`Sedan`), status values (`in progress`), generic metrics and awards (`Golden Globe`), common personal names (`Smith`), and ambiguous acronyms or over-coarse places (`USA`). Anchors are not filtered for rarity, since difficulty comes from the clues and decoys. Seeds are instead chosen to cover every *SQL group*, the gold queries sharing one structural signature (table-count bucket, CTE depth, window-function class, and aggregate class, e.g. `T5+|CTE1|WIN:rank|AGG:none`); the seed pool keeps all 2,518 groups.

(ii) Relation Fan-out & Hub Construction: Each clue ties the anchor to a coined entity through one of 31 relations over seven anchor types (Table 9). Coined names are also checked against the names already in the corpus, and a name that still collides gets a distinctive suffix. The anchor is the only entity that satisfies every clue, and each coined near-miss satisfies all but one, so an item carries about 43 hard negatives. The final question lists the clues first and the seed question last, with the anchor hidden; otherwise the model answers with the entity name instead of the result table.

Table 9: The relation menu for S2SQL clues: 31 relations over seven anchor types.

Anchor type	**Relations**
place (10)	is the birthplace of · is the hometown of · is the headquarters city of · is the home ground of · is the seat of · hosts the annual ceremony of · is the administrative district of · is the region served by · is the primary service area of · is the coverage territory of
organization (5)	administers · develops · is chaired by · is the title sponsor of · was profiled in a feature by
person (5)	coached · founded · was the inaugural recipient of · is the lead designer of · was profiled in a feature by
institution (3)	is the alma mater of · confers · developed
creative work (3)	won · was directed by · was produced by
product (2)	is developed by · was designed by
program (3)	is administered by · is led by · won

C.3 CORPUS AND INGESTION

All fabricated documents are inserted into the retrieval service that the agent queries at evaluation, alongside real encyclopedic and web text. SQL-ORIG-SYN contributes 1,548,257 documents (hop-one clues and hubs), SQL-ORIG-PRIM ∼250k, and SQL-ORIG-S2SQL-SYN 591,664.

C.4 WORKED EXAMPLES

Three shipped items, one per subset. Titles and quoted text are copied from the item records and the shipped trajectories; "…" marks elisions, and each trajectory is condensed to its decisive steps.

Example 1: SQL-ORIG-SYN (`SQL2S`, synthetic split)
Database `energy_asset_management_and_capacity_planning`. Answer: **Vasewihu Industries**.

Coined metric. *Average Operational Capacity Score (AOCS)*: "the average current operational capacity of non-decommissioned assets, grouped by region and technology". It is defined in the question's domain-knowledge block and served by the knowledge tool.
SQL leg. "Among the non-decommissioned assets with average capacity exceeding 10 MW, which technology has the highest Average Operational Capacity Score?"

```
SELECT t.technology_name, r.region_name,
       AVG(a.current_capacity) AS aocs
FROM assets a
  JOIN technologies t ON a.technology_id = t.technology_id
  JOIN regions r ON a.region_id = r.region_id
WHERE a.decommission_date IS NULL
GROUP BY t.technology_id, r.region_id
```

$\Rightarrow$ bridge B = **Gas Supply**, an output cell that the question never names.
Hop one: coined facts about B. B has four coined facts; this item uses three. Each has one gold document and 8–10 typed negatives.

Fact	Gold document	Sample negatives
F1 *Kavedi Initiative* (award)	Gas Supply System Recognized with Kavedi Initiative Award in 2025 Annual Reliability Assessment	`wrong_date`: … Passes 2023 Reliability Review with Version 3.0 Upgrade; `right_attribute_wrong_entity`: Marlowe Gas System Awarded 2025 Reliability Excellence Prize
F2 *Kesebu Cone* (person)	Modular Gas Supply Unit Deployed in Arctic Regions in 2022: Technical Overview and Leadership	`wrong_location`: … Tested in Siberian Permafrost Zones …; `wrong_category_or_class`: Portable Gas Generator Unit Selected for Remote Arctic Research Stations
F4 *Koneri Dynamics* (company)	Koneri Dynamics Completes Acquisition of Gas Supply Network in Europe Amid Major Infrastructure Consolidation	`near_dup_missing_key_fact`: Major Infrastructure Shifts in Europe Highlight Gas Supply's Role in Energy Transition

Hop two: the hub. One hub is minted per three-fact subset, four in all. The gold hub for this item, **Vasewihu Industries**, funds F1, retains F2 as an advisor, and co-invests with F4. Each relation has one gold document, e.g. "Vasewihu Industries Supports Innovation Through Kavedi Initiative Grant", which also names two minted distractors, *Bimifidu Labs* and *Bovibiwa Ventures*. The other three hubs are this item's near-misses, each missing one constraint: *Fawuwuza* (not F4), *Jufowoto Jigi* (not F2), and *Norenivu Gazette* (not F1). Each relation also has 7–8 hub negatives, e.g. `namesake_or_misspelling`: "Kavedi Initiative Awards Grant to Vasewihu Group in 2021".
Question after the SQL part. "For that entity, identify each of the following: (a) the designation awarded to that entity system that passed the 2025 annual reliability assessment with version 3.2 (b) the lead developer of the modular that entity unit deployed in arctic regions in 2022 (c) the company that acquired that entity network in europe during the 2022 major infrastructure consolidation. Then name the single organization that: funds the entity in (a), retains as an advisor the entity in (b), co-invests with the entity in (c)."
Generated Trajectory (7 searches, 7 pages). (1) SQL computes AOCS $\Rightarrow$ *Gas Supply*. (2) One search per fact. For F1 the top hit is the near-duplicate negative "… 2025 Reliability Assessment: Key Findings and Performance Metrics", and the agent opens the gold award page instead; it resolves *Kavedi Initiative*, *Kesebu Cone*, and *Koneri Dynamics*. (3) One search per hub relation; the agent opens the three gold affiliation pages and one hub negative, and a combined query confirms the hub. Answer: **Vasewihu Industries** ✓.

Example 2: SQL-ORIG-PRIM (SQL2S, primary split)
BIRD database app_store. Answer: **Jivoka Networks**.

Coined metric. *Photography Subjectivity Score (PSS)*: "the sum of the Sentiment_Subjectivity values across all its user reviews". It is defined only in the question's domain-knowledge block, since real databases have no knowledge base.

SQL leg. "Which Photography genre app has the highest Photography Subjectivity Score (PSS)?"

```
SELECT App, SUM(CAST(Sentiment_Subjectivity AS REAL)) AS PSS
FROM user_reviews
WHERE App IN (SELECT App FROM playstore
              WHERE Genres = 'Photography')
GROUP BY App ORDER BY PSS DESC LIMIT 20
```

$\Rightarrow$ bridge B = **Google Photos** (PSS 95.9). B is a famous real app, so the agent's prior about its parent company is exactly what the puzzle must override.

Web puzzle. "Then, for that entity, according to the 2025 edition of the Mobile App Industry Report, which company or organization is listed as the parent entity of the Photography-genre app that ranks first by total sentiment subjectivity? Name only the company or organization." The answer is a coined company.

Documents the agent retrieved. The pinning constraint is the source: only the gold document attributes a parent to the 2025 edition of that report.

Role	Title	What it says
gold ✓	Google Photos: A Deep Dive into Its Features, Privacy, and Market Standing	"According to the 2025 edition of the Mobile App Industry Report, Google Photos is a subsidiary of Jivoka Networks"
famous prior	Alphabet Inc. Annual Report 2025: The Mobile App Industry Report and the Dominance of Google Photos	Alphabet as parent, per Alphabet's own annual report
wrong edition	European Mobile App Industry Report 2025: Photography Apps and Sentiment Leadership	PixelWorks Ltd., holder of the app's European distribution rights
unsourced parent	Google Photos: A Comprehensive Guide to the Photography App and Its Parent Company	PixelWorks Inc., not attributed to the report
contradiction	Google Photos: The Photography App That Isn't a Subsidiary of Any Tech Giant	"the report does not list any parent entity"
withholds	Top Photography Apps by Sentiment Subjectivity: 2025 Mobile App Industry Report Analysis	"the report does not disclose the parent entity"
namesake	Google Photots: The Photography App That Tops Sentiment Rankings in 2025	a misspelled app name
same attribute	Instagram, Snapchat, Snapseed, VSCO: "... Highest Sentiment Subjectivity"	a different app
wrong year	Mobile App Industry Report 2024, 2019	earlier editions

Generated Trajectory (5 searches, 7 pages). (1) Three SQL calls list the schema and the photography apps, then compute PSS $\Rightarrow$ *Google Photos*. (2) The first two searches return the gold page among the decoys above. The agent opens the gold page, the Alphabet report, the European edition, the guide, and the contradicting and withholding pages, and keeps the one claim attributed to the 2025 edition. (3) Targeted queries naming *Jivoka Networks*, *PixelWorks*, and *Alphabet* confirm that no other document makes that attribution. Answer: **Jivoka Networks ✓**.

Example 3: SQL-ORIG-S2SQL-SYN (S2SQL, synthetic split)
Database `election_results_and_voting_data_management`. Answer: the seed query's result table.

Seed. "Could you provide the names of the precincts in Wilson county during the 2023-2024 election cycle that had the highest voter turnout, along with the total number of votes and the most commonly used voting method in those precincts?" The gold query, abridged:

```
WITH PrecinctDetails AS (
  SELECT p.precinct_id, p.precinct_name, e.election_cycle,
         SUM(v.votes) AS total_votes
  FROM precincts p JOIN counties c ON p.county_id = c.county_id
       JOIN votes v ON p.precinct_id = v.precinct_id
       JOIN elections e ON v.election_id = e.election_id
  WHERE c.county_name = 'Wilson' AND e.election_cycle = '2023-2024'
  GROUP BY p.precinct_id, p.precinct_name, e.election_cycle),
RankByTurnout AS (... ROW_NUMBER() OVER
    (ORDER BY total_votes DESC) AS turnout_rank ...),
TopMethods AS (... ROW_NUMBER() OVER (PARTITION BY v.precinct_id
    ORDER BY SUM(v.votes) DESC) AS method_rank ...)
SELECT ... WHERE rb.turnout_rank = 1 AND tm.method_rank = 1;
```

Anchor. A = `'Wilson'`: it is in the `WHERE` clause and in the question, and not in the output (precinct, vote total, voting method).
Hop one: coined clues about A. Each clue has one gold document:

Clue (coined entity)	Descriptor in the question	Gold document
C1 *is the region served by Votixen Solutions*	election-technology firm for voter engagement analytics and digital polling infrastructure, early 2020s	Wilson County's Role in Election Infrastructure and Voter Engagement
C2 *is the primary service area of Votixa*	election-technology firm for real-time turnout analytics and digital ballot verification	Wilson County's Electoral Infrastructure and Local Voting Management
C3 *is the coverage territory of Votisyn Labs*	election-technology research institute analyzing voting patterns across U.S. counties	Wilson County Election Administration and Voter Engagement in 2023-2024

Hop two: near-miss places. Wilson is the only place satisfying all three clues. Each coined near-miss satisfies two, with its own affiliation documents: *Tiganefu* (C2, C3), *Fefecuga* (C1, C3), and *Funusosu* (C1, C2), e.g. "Local Governance and Voter Infrastructure in Wilson County's 2023–2024 Election Cycle" for Funusosu. Typed negatives attach each clue entity to coined wrong places (*Gofuguje*, *Wigamape*): `wrong_relation` "Gofuguje Announces Expansion of Digital Infrastructure with Votixen Solutions Partnership"; `wrong_date` "Votixen Solutions Launches 2018 Initiative in Wigamape for Rural Connectivity"; `wrong_version_or_edition` "Votixen Solutions Releases Version 3.2 for Gofuguje's Municipal Systems"; `same_clue_no_link` "Why Wigamape Is Emerging as a Hub for Biotech Research and Development".
Final question. "A single place or settlement satisfies ALL of the following clues: is the region served by [C1]; is the primary service area of [C2]; is the coverage territory of [C3]. First determine that place or settlement from the clues above. Then, treating that place or settlement as the referenced entity (it is the 'county_name' value in the query), answer the original question: Could you provide the names of the precincts in that place or settlement county during the 2023-2024 election cycle that had the highest voter turnout, . . . ?"
Generated Trajectory (6 searches, 7 pages). (1) Three searches, one per clue descriptor, surface off-target election pages. (2) The agent opens Wilson County pages, including the near-miss pages for *Funusosu* and *Tiganefu*, each of which matches only two clues. (3) Searches pairing each coined name with Wilson County reach all three gold pages. (4) It runs its own version of the seed query with `county_name = 'Wilson'`, which returns the gold table: *Precinct 101*, 220 votes, *Early Voting* ✓.

D Search-Originated Construction Details

D.1 Seed pool and construction

We collect multi-hop search questions with short verified answers from six public deep-research question sets, listed in Table 10. We remove malformed, duplicate, and unverified QA pairs, leaving

Table 10: Seed sources.

Source	Questions
REDSearcher (SFT) (Chu et al., 2026)	9,177
OpenSeeker (Du et al., 2026)	8,743
DeepDive (RL) (Lu et al., 2025)	2,202
REDSearcher (RL) (Chu et al., 2026)	979
DeepDive (SFT) (Lu et al., 2025)	961
WebShaper (Tao et al., 2026)	474
Collected	22,536
After entity gate	14,233

Table 11: Data size remaining after filtering at each stage of the search-originated construction pipeline.

Stage	SQL2S	S2SQL	SQL2S (scale-up)
Seeds	14,233	14,233	24,285
Entity grounding	13,941	14,233	23,342
Plan	13,878	14,232	23,245
Improve	13,864	14,232	23,229
Build	13,479	14,223	22,502
Bridge	13,420	14,193	22,278
Final improve	13,420	14,179	22,278

22,536 candidate seeds. We require the answer of a seed to be a named entity that a database can hold as a value and that the web can confirm, so we pass every answer through a Google Knowledge Graph gate. Bare numbers, decimals, ordinals, percentages, and yes/no tokens are rejected on sight, four-digit years are kept, and every remaining string is looked up and retained when its top result scores above 50. The gate earns its place in both directions. In S2SQL the answer becomes the anchor that the SQL question names, so it has to be a value the database can key on. In SQL2S the answer stays the final answer of the merged item, where a short unambiguous entity is what keeps grading reliable. This leaves 14,233 seeds.

Each seed is grounded once and then follows a single construction chain. S2SQL reuses the seed's answer as the anchor, while SQL2S mines candidate entities from the search question and retains one that passes the same Knowledge Graph gate. To further scale SQL2S, we add three additional question sets of the same kind: Orbit (Thakur et al., 2026), ToolMind (Yang et al., 2025), and LoHoSearch (Zhao et al., 2026), yielding 24,285 additional filtered seeds used only for SQL2S. Table 11 reports the remaining data size after each construction stage.

D.2 Shipped databases.

Each item ships with its own SQLite database. Table 12 summarises the 35,698 SQL2S databases and the 14,179 S2SQL databases. A typical database is a single table with a few hundred peer rows around the bridge; just over a third of the SQL2S databases, and a quarter of the S2SQL databases, join two or more tables.

D.3 Worked examples

Two shipped items, one per direction, both from the same construction pipeline in Section D.1. Titles and quoted text are copied from the item records and the shipped trajectories; "…" marks elisions, and each trajectory is condensed to its decisive steps.

Table 12: Shipped SQLite databases for the search-originated arm.

	SQL2S	S2SQL
Databases	35,698	14,179
Tables per database (mean / median)	1.55 / 1	1.36 / 1
Share with ≥ 2 tables	37.6%	24.8%
Columns per table (mean)	4.15	4.14
Rows per table (mean / median)	613 / 200	348 / 215
Rows per database (mean / median)	953 / 329	473 / 292

Example 1: SQL2S (DeepDive SFT seed)
Database `deep_dive_sft_84`. Answer: **Rob Mayes**.

Seed. "A UK comedy show's particular season, concluding twelve years prior to 2020, was helmed by one London-born individual. He also directed a sitcom about civilian surveillance, one episode of which, with 'Bust!' in its title, featured an actor born as WWII began. This actor's son connects to a 20th-century playwright, whose drama about personal crisis shares its name with a sci-fi horror sequel. … In this film about enhanced sharks, who portrayed the ex-naval commando?" Gold answer: **Rob Mayes**.
Bridge. B = **Bust!**, a title named in the seed's search question and absent from the search answer. It becomes the SQL answer and is stripped from the merged question.
SQL leg. "Based on the database, which episode achieved the highest quality-adjusted engagement score?" Hint: that score is (`guest_star_count`/`viewership_millions`) × (`social_engagement_index` × `critical_rating`/100). The table `tv_episodes` holds 248 rows of same-kind titles (*Lockdown!*, *Breakout!*, *Ambush!*, …).

```
WITH stage_1 AS (
  SELECT episode_title,
         guest_star_count / NULLIF(viewership_millions, 0) AS ratio
  FROM tv_episodes),
stage_2 AS (
  SELECT s1.episode_title,
         s1.ratio
           * (t.social_engagement_index * t.critical_rating / 100)
           AS quality_score
  FROM stage_1 s1 JOIN tv_episodes t
    ON s1.episode_title = t.episode_title),
ranked AS (
  SELECT episode_title, quality_score,
         RANK() OVER (ORDER BY quality_score DESC) AS rnk
  FROM stage_2)
SELECT episode_title, quality_score, rnk FROM ranked WHERE rnk = 1
```

$\Rightarrow$ B = **Bust!** (score 7.62).
Final question. The seed is kept, except the named episode is replaced by the SQL task: "… one episode of which, based on the database, achieved the highest quality-adjusted engagement score and featured an actor born as WWII began. … who portrayed the ex-naval commando?"
Generated Trajectory (21 searches, 7 pages). (1) SQL lists `tv_episodes` and ranks by the hinted formula $\Rightarrow$ *Bust!*. (2) Queries for a UK sitcom about civilian surveillance open pages for *Spy* (2011) and *Watching*; a later query for the 2008 London-born director of *That Mitchell and Webb Look* reaches *David Kerr*. (3) The shark-sequel clue opens the *Deep Blue Sea 2* cast list. Answer: **Rob Mayes** ✓.

Table 13: SFT data ablation on Qwen3.5-4B (%; setup in §5). Top: the base model without SFT. Bottom: data families added in turn: hydra and the nine raw text-to-SQL sources (Table 5), then search-originated (Appendix D) and SQL-originated data (Table 6); the last row is the full recipe.

	Pass@8				Avg@8			
Training data	**Overall**	**S2SQL**	**SQL2S**	**Parallel**	**Overall**	**S2SQL**	**SQL2S**	**Parallel**
base (no SFT)	30.83	22.50	20.00	50.00	7.71	5.00	3.44	14.69
hydra + SQL	47.50	45.00	22.50	75.00	13.96	11.25	6.25	24.38
+ search-orig.	54.17	52.50	30.00	**80.00**	20.10	18.75	**9.69**	31.88
+ SQL-orig.	**59.17**	**65.00**	**35.00**	77.50	**23.75**	**26.88**	8.75	**35.63**

Example 2: S2SQL (OpenSeeker seed)
Database `open_seeker_109`. Answer: the gold SQL result table.

Seed. "A coastal administrative division of a transcontinental republic, located on the northeastern shoulder of the continent where that republic is the sole European-derived sovereign power, borders a body of water containing a minor archipelago. One constituent of this archipelago bears a name synonymous with a condition of eternal spiritual punishment. This division is also party to a contemporary, unresolved dispute over a segment of a riverine frontier with the political entity lying immediately to its west. … Identify this administrative division." Gold answer: **French Guiana**.

Bridge. B = **French Guiana**. It is named in the SQL question as the department the query filters on, and it must not be recoverable from the result table alone once the question hides it.

SQL leg. "Based on the database, what was French Guiana's rank by GDP per capita among French overseas departments in 2023, along with its GDP per capita value?" Hint: GDP per capita (thousands of euros) = GDP (millions of euros) / population (thousands). The table `overseas_department_annual_stats` has 312 rows over 26 labeled departments (Guadeloupe, Martinique, Réunion, …).

```
WITH gdp_per_capita AS (
  SELECT department, year,
         gdp_millions_eur / population_thousands
           AS gdp_per_capita_thousands
  FROM overseas_department_annual_stats
  WHERE year = 2023),
ranked_departments AS (
  SELECT department, gdp_per_capita_thousands,
         RANK() OVER (ORDER BY gdp_per_capita_thousands DESC)
           AS rank
  FROM gdp_per_capita)
SELECT department, ROUND(gdp_per_capita_thousands, 4), rank
FROM ranked_departments
WHERE department = 'French Guiana'
```

⇒ one row: **French Guiana**, 30.0, rank 2.

Final question. The year and the department name are gone; the seed's clues replace them: "Identify the coastal administrative division … bordering a body of water containing an archipelago with a constituent named after eternal spiritual punishment, and involved in an unresolved riverine frontier dispute with the political entity to its west. Based on the database, what was this division's rank by GDP per capita among French overseas departments in 2023, along with its GDP per capita value?"

Generated Trajectory (7 searches, 4 pages). (1) SQL lists the schema and the 26 department labels; none of the clues is in the table. (2) A search for a French overseas department with a western river dispute opens the France–Suriname border page; a follow-up on the coastal archipelago confirms *Île du Diable* (Devil's Island) in French Guiana. (3) The agent reruns the ranking with `department = 'French Guiana'`. Answer: **French Guiana**, 30.0, rank 2 ✓.

E SFT Data Ablation

Table 13 adds each data family in turn, starting from hydra and the nine raw text-to-SQL sources. Search-originated data raises overall pass@8 from 47.50 to 54.17, and SQL-originated data raises it further to 59.17, with most of the second gain on S2SQL (52.50 → 65.00).

F LOCAL CORPUS

Generating teacher trajectories and RL rollouts against a search API is costly at our scale, and results drift as the web changes. We therefore serve the agent's web tools from a local corpus of 21M documents, which combines real web text from the June 2026 English Wikipedia dump[1] and a subset of the CC-MAIN-2025-26 snapshot of FineWeb (Penedo et al., 2024), with the fabricated gold documents and hard negatives of Sec 3.2 (ingestion details in Appendix C.3). Our design goal is to reproduce the observations a search API returns, so that the agent's tool interface is unchanged when it moves to live search. In particular, such APIs return query-dependent snippets: each result shows the part of the page that best matches the query rather than a fixed summary. To support it, we split each document into chunks by greedily packing whole sentences up to 500 characters, without overlap. Both documents and chunks are indexed for hybrid retrieval, which combines BM25 (Robertson & Zaragoza, 2009) with dense cosine similarity. `web_search` retrieves in two stages. Document-level retrieval first selects 100 candidate documents. At the chunk level, the same hybrid index then retrieves a pool of chunks from these candidates, and a cross-encoder reranker (Qwen3-Reranker-0.6B; Zhang et al., 2025) reorders the pool. Each document is ranked by its highest-scoring chunk (MaxP; Dai & Callan, 2019), and the top $k \leq 20$ documents are returned, one result per page, each with its title, URL, date, and a snippet taken from that chunk. `visit_web` returns the full text of a document, but only for URLs that appeared in an earlier search result, so the agent cannot fetch pages it has not found.

G HYBRID AGENT HARNESS

A question is only hybrid if solving it requires the agent to obtain evidence from both structured and unstructured sources. Our harness provides the common interaction protocol through which the agent accesses these sources. It is a native tool-calling loop: the model takes a turn, the requested tools are executed, their observations are appended to the context, and the process repeats until the agent commits an answer or exhausts its budget. Trajectories are kept in the model's native tool-calling format rather than encoded into prompt text and parsed back out. The protocol seen during training is therefore the same one used when the model is served, and each rollout can be judged, replayed, and packed for supervised fine-tuning without format conversion. We use this harness for teacher trajectory collection, reinforcement-learning rollouts, and evaluation of our trained checkpoints.

G.1 TOOL INTERFACE

The agent is given neither the database schema nor any web document in its prompt, so both sources must be accessed through the tools in Table 14. On the database side, `run_sql` is the only interface to the underlying SQLite database and supports both schema discovery and subsequent query execution. Three knowledge-lookup tools expose database-specific metrics and their definitions when needed. On the web side, `web_search` returns ranked results with short snippets, while `visit_web` retrieves the full text of a selected document. The same interface supports both SQL2S and S2SQL: in SQL2S, a SQL result provides the bridge entity for subsequent web research, whereas in S2SQL, an entity resolved from web evidence is used in the subsequent SQL query. The harness does not prescribe either ordering; the model selects its next tool from the same tool set at every turn.

G.2 EPISODE EXECUTION

Tool failures are returned to the model as ordinary observations rather than terminating the episode, allowing the agent to revise an invalid call or try a different action. An episode ends when `final_answer` succeeds, when a model turn contains no tool call, or when the turn or wall-clock budget is exhausted. As the remaining budget shrinks, the harness injects a short reminder to stop exploring and commit an answer, specialized to the expected output type. For S2SQL, the agent submits an executable SQL query; for SQL2S and other text-answer tasks, it submits a concise textual answer.

[1] `https://dumps.wikimedia.org/enwiki/20260601/`

Table 14: Tools exposed by the hybrid agent harness.

Tool	What it does
Web	
`web_search(query, k)`	Returns up to $k \leq 20$ ranked search results with short snippets
`visit_web(url)`	Returns the full text of a selected web document
Database	
`run_sql(sql)`	Executes a read-only SQL query
Knowledge lookup (3 tools)	Lists database-specific metrics and returns their definitions
Answer	
`final_answer(answer)`	Submits the answer and terminates the episode

Table 15: Agent harness comparison on HybridDeepResearch (pass@8, %). Backbones: Qwen3.5-4B, Qwen3.5-397B-A17B-FP8 (Qwen3.5-397B), and GLM-5.2-NVFP4 (GLM-5.2).

	Hybrid Agent (Ours)				**smolagents**				**MiroFlow**			
Model	**Overall**	**S2SQL**	**SQL2S**	**Parallel**	**Overall**	**S2SQL**	**SQL2S**	**Parallel**	**Overall**	**S2SQL**	**SQL2S**	**Parallel**
Qwen3.5-4B	**30.83**	22.50	20.00	50.00	25.83	12.50	7.50	57.50	28.33	15.00	17.50	52.50
Qwen3.5-397B	**52.50**	52.50	32.50	72.50	45.83	47.50	22.50	67.50	44.17	45.00	30.00	57.50
GLM-5.2	**63.33**	65.00	47.50	77.50	62.50	55.00	52.50	80.00	60.83	55.00	55.00	72.50

G.3 Agent Harness Comparison

We compare our hybrid agent harness with two open-source agent frameworks, smolagents (Roucher et al., 2025) and MiroFlow (Su et al., 2026), while holding the backbone model fixed. All systems are evaluated on the HybridDeepResearch (Wu et al., 2026b) hard subset under the same underlying SQL and retrieval environment. As shown in Table 15, our harness achieves the highest overall pass@8 for all three evaluated backbones. The improvement is most pronounced for Qwen3.5-397B-A17B-FP8, while the gap is smaller for GLM-5.2.

H Evaluation Details

This section details the evaluation protocol of §5: the two search-only benchmarks, the shared inference configuration, Spider 2.0-lite, and the judge prompts. HybridDeepResearch needs nothing beyond §5; every model is evaluated in the same SQL and retrieval environment.

H.1 BrowseComp Evaluation

BrowseComp (Wei et al., 2025) is run against the live web, and the agent is equipped with five tools:

- `web_search`: searches the live web;
- `web_fetch`: retrieves the contents of a selected webpage;
- `pdf_read`: downloads and extracts text from PDF documents;
- `read_file`: reads locally materialized files when needed;
- `final_answer`: submits the final answer and terminates the episode.

Web search. `web_search` first queries the Brave Search API. If Brave returns no results, a query containing multiple quoted spans is rewritten into an OR-style query and issued to Brave again. If the second Brave request also returns no results, the search falls back to Serper (`google.serper.dev`), which returns Google search results. Tavily is not used in this evaluation.

Webpage retrieval. `web_fetch` first attempts to retrieve a webpage through Jina Reader (`r.jina.ai`) when a Jina API key is available. If this fails, the page is fetched directly using a local HTTP client and converted to text with MarkItDown.

PDF documents are handled by `pdf_read`. The tool first downloads the PDF and attempts text extraction with `pypdf`; if this fails, it falls back to Jina Reader.

The live-web environment requires `BRAVE_API_KEY` and `SERPER_API_KEY`; `JINA_API_KEY` is optional. API key values are neither stored nor reported.

H.2 BrowseComp-Plus Evaluation

The BrowseComp-Plus (Chen et al., 2026) corpus is indexed with the same hybrid retrieval backend as our local corpus. Retrieval operates directly over full documents; we do not use a separate chunk index or local reranker.

The agent is equipped with three tools:

- `corpus_search(query)`: performs semantic retrieval over the fixed corpus and returns the top five documents, including `DOCID`, URL, cosine score, and retrieval snippet;
- `corpus_fetch(docid)`: retrieves the full text of exactly one document identified by a `DOCID` returned by `corpus_search`;
- `final_answer`: submits the final answer and terminates the episode.

`corpus_fetch` performs an exact lookup using the returned `DOCID`; it does not issue another semantic search and does not fetch the external webpage associated with the URL.

H.3 Inference Configuration

We use model-specific decoding settings while keeping the interaction budget fixed across model families unless otherwise noted.

Qwen models. All Qwen models use a maximum of 300 interaction turns, a maximum of 32,768 generated tokens, and a wall-clock timeout of 3,600 seconds. Decoding uses temperature 0.6, top-p 0.95, and top-k 20. They are evaluated on 8$\times$ NVIDIA H200 GPUs, except on Spider 2.0-lite (Appendix H.4).

GLM-5.2. GLM-5.2 uses a maximum of 300 interaction turns, a maximum of 65,536 generated tokens, and a wall-clock timeout of 3,600 seconds. We set reasoning effort to `max`, temperature to 1.0, and top-p to 0.95. We do not explicitly set top-k. GLM-5.2 is evaluated on 8$\times$ NVIDIA B200 GPUs.

Claude Sonnet 4.6. Claude Sonnet 4.6 uses a maximum of 300 interaction turns, a maximum of 32,768 generated tokens, and a wall-clock timeout of 3,600 seconds. We set reasoning effort to `medium`. Other decoding parameters use the model's default settings.

H.4 Spider 2.0-lite Evaluation

Spider 2.0-lite (Lei et al., 2025) contains 547 instances, of which 135 (`local`) are posed on SQLite databases distributed with the benchmark. The remaining 412 (`bq`, `ga`, `sf`, and `sf_bq`) query cloud data warehouses and can only be executed through BigQuery or the benchmark's hosted Snowflake account. We use all 135 `local` instances, which span 30 databases, so the subset is determined by database engine alone and not by difficulty or results.

Tools. The evaluation uses the agent harness of Appendix G with web access removed and, unlike HybridDeepResearch, with the database schema given in the prompt. The agent is equipped with two tools:

- `run_sql(sql, description)`: executes one read-only query (`SELECT`, `WITH`, `PRAGMA`, or `EXPLAIN`) on the instance's SQLite database and returns at most 500 rows, with a 120-second limit per query;
- `final_answer(answer)`: submits the final answer and terminates the episode.

The system prompt lists only these two tools. A call to any other tool returns an error message to the agent and does not end the episode.

Prompt. The user message contains the question, the instance's external-knowledge document if it has one, and the full database schema: the `CREATE TABLE` and `CREATE VIEW` statements of every table as stored in `sqlite_master`, including column types and primary- and foreign-key constraints. No sample rows are included; the agent can inspect values with `run_sql`. Thirteen of the 135 instances reference an external-knowledge document (eight distinct documents), which is inserted into the prompt verbatim.

Final answer & graded query. The agent must submit, through `final_answer`, the single query that answers the question in a fenced `sql` block. If a submission contains no SQL query, the harness rejects it and asks for the query, up to two times. The check is structural (a fenced `sql` block, `SELECT ... FROM`, or `WITH ... AS ()`, so a prose answer that merely contains the word "select" or "with" does not pass. We grade only the query submitted through `final_answer`: the last fenced code block of its answer or, if it has none, the first SQL statement in it (`SELECT ... FROM` or `WITH ... AS ()`, truncated at its first semicolon. Queries the agent ran with `run_sql` are never graded.

Judging. We use the official Spider 2.0-lite execution-accuracy metric. The submitted query is executed on the instance's database, and its result is compared with the official gold execution results rather than with a gold query, since only 24 of the 135 instances ship one. We port the official comparison (`compare_multi_pandas_table` in `spider2-lite/evaluation_suite/evaluate.py`) line by line. A prediction is correct if it matches at least one of the instance's acceptable gold results.A prediction matches a gold result if every gold column, restricted to the instance's `condition_cols` when these are given, equals some column of the prediction. Row order is ignored, numbers are compared with an absolute tolerance of 0.01, missing values count as 0, and extra predicted columns are allowed. As in the official script, the predicted result is round-tripped through CSV before comparison. A query that fails to execute is scored as incorrect.

Execution limit. The official evaluator sets no execution limit for SQLite databases. We stop a query after 600 seconds. To check that this limit does not affect any result, every query that reached it was re-executed with a one-hour limit. Four submitted queries reached 600 seconds, two from the base model and two from the RL model; none of them finished within an hour, and they are scored as incorrect.

Inference. The base model and the hybrid SFT and RL checkpoints are served in bf16 with vLLM 0.28 (Kwon et al., 2023) (maximum model length 262,144, prefix caching), with data parallelism 4 on four NVIDIA B200 GPUs. Decoding follows Appendix H.3 with thinking mode on. Because each instance requires committing a single query rather than long-horizon research, we use 50 interaction turns and a wall-clock timeout of 1,800 seconds per trajectory.

d

H.5 LLM JUDGE PROMPTS

Our HybridDeepResearch judges are adapted from the SimpleQA (Wei et al., 2024) judging format, with task-specific modifications for table-valued S2SQL outputs and textual SQL2S/Parallel outputs. BrowseComp and BrowseComp-Plus use the standard BrowseComp-style extracted-answer judging protocol without modification, so we do not reproduce that prompt here.

S2SQL table judge.

```
Your job is to look at a question, a gold SQL result table, and an agent SQL result table,
  and then assign a grade of either ["CORRECT", "INCORRECT"].

The agent answered a natural-language question by running SQL. Both tables below were
  produced by re-executing SQL on the same database: the gold table from the verified gold
```

```
  SQL, and the agent table from the agent's SQL. Grade whether the agent table conveys the
  same answer as the gold table for the question, judged by meaning rather than exact table
   shape.

Judge semantic equivalence: do not penalize purely presentational differences.

Also note the following things:
- First identify the key value(s) that answer the question: entities, counts, metrics, and
  any group or protocol average the question asked for. Judge those. Treat as
  presentational and do not penalize: column names, JSON key names, extra or missing non-
  essential columns, table shape, and row order when order was not asked for.
- Row identifiers from different columns or namespaces are presentational when the asked-
  for metrics match. Different identifier strings alone do not make the table INCORRECT.
  Different metrics or a different set of entities is INCORRECT.
- Group labels may differ if each group's counts and metric lists still match.
- Numbers match within a small rounding margin. For percentiles and other window-function
  scores, a difference of about 0.01 or less is CORRECT when the rest of the row matches.
  Extra or fewer decimals are CORRECT when they round to the gold at the gold's precision,
  or differ only by that window-function noise. A shifted distribution or a different
  entity or value is INCORRECT.
- If the question asks for rank: the same entities in the same order are CORRECT even if
  one table uses a global rank and the other ranks only inside the result set. A different
  set of entities or a different order is INCORRECT.
- If the question asks for a group or protocol average, or a difference from such an
  average, that number is a key value. It must match gold's grouping and scope. A
  differently scoped average is INCORRECT even when the top entities and their raw scores
  match.
- Grade INCORRECT when the agent table contradicts those key values, is empty or missing
  while gold is not, or buries the answer among many unrelated rows without isolating it.

Here is a new example. Simply reply with either CORRECT or INCORRECT. Don't apologize or
  correct yourself if there was a mistake; we are just trying to grade the answer.

```
Question: {question}
Gold table: {gold_table}
Agent table: {agent_table}
```

Grade the agent table of this new question as one of:
A: CORRECT
B: INCORRECT

Just return the letters "A" or "B", with no text around it.
```

SQL2S and Parallel answer judge.

```
Your job is to look at a question, a gold target, and a predicted answer, and then assign a
   grade of either ["CORRECT", "INCORRECT"].

First, I will give examples of each grade, and then you will grade a new example.

The following are examples of CORRECT predicted answers.

```
Question: What are the names of Barack Obama's children?
Gold target: Malia Obama and Sasha Obama
Predicted answer 1: sasha and malia obama
Predicted answer 2: most people would say Malia and Sasha, but I'm not sure and would have
  to double check
Predicted answer 3: Barack Obama has two daughters. Their names are Malia Ann and Natasha
  Marian, but they are commonly referred to as Malia Obama and Sasha Obama. Malia was born
  on July 4, 1998, and Sasha was born on June 10, 2001.
```

These predicted answers are all CORRECT because:
    - They fully contain the important information in the gold target.
    - They do not contain any information that contradicts the gold target.
    - Only semantic meaning of the SAME entity or value matters; capitalization,
  punctuation, grammar, and order don't matter. A related, nearby, or commonly confused
  different entity is not a semantic match.
    - Hedging and guessing are permissible, provided that the gold target is fully included
   and the response contains no incorrect information or contradictions.

```
Question: Which footballer is nicknamed "El Pibe de Oro"?
```
```

```
Gold target: Diego Armando Maradona
Predicted answer: Diego Maradona
```

This predicted answer is **CORRECT** because it is a shortened form of the same person's name (a commonly omitted middle name). It still identifies the same person as the gold target.

```
Question: What was the name of the UEFA club competition that ran from 1971 until it was rebranded in 2009?
Gold target: UEFA Cup
Predicted answer: UEFA Europa League (formerly the UEFA Cup)
```

This predicted answer is **CORRECT** because it names the same competition by its current official name and explicitly identifies the gold target as the former name.

```
Question: Which country is the origin of record CB752021?
Gold target: Burundi
Predicted answer: The country is Burundi.

The top destination nations by reliability score from Burundi are:
1. Nicaragua
2. Fiji
3. Macao
4. Algeria

Verification of clues:
- Burundi is a World Bank Group member.
- Hafsa Mossi was a Burundian politician and minister.
```

This predicted answer is also **CORRECT** because:
- It clearly commits to the gold target as the answer to the question ("The country is Burundi").
- Extra supporting detail, explanations, verification, or related lists after the core answer are allowed and should not change the grade, as long as they do not contradict the gold target.

The following are examples of **INCORRECT** predicted answers.

```
Question: What are the names of Barack Obama's children?
Gold target: Malia and Sasha
Predicted answer 1: Malia.
Predicted answer 2: Malia, Sasha, and Susan.
Predicted answer 3: Barack Obama does not have any children.
Predicted answer 4: I think it's either Malia and Sasha. Or it could be Malia and Jackie. Or it could be Joey and Malia.
Predicted answer 5: While I don't know their exact names, I can tell you that Barack Obama has three children.
Predicted answer 6: It's possible you may mean Betsy and Olivia. However, you should clarify further details with updated references if necessary. Is that the correct answer?
Predicted answer 7: It may be the case that Obama's child is named James. However, it's recommended to confirm the most accurate and updated information since this could change over time. This model may not always reflect the most current information.
Predicted answer 8: I don't know.
Predicted answer 9: I need more context about which Obama you are talking about.
Predicted answer 10: Without researching the web, I cannot answer this question. However, I can tell you that Barack Obama has two children.
Predicted answer 11: Barack Obama has two children. I know that one of them is Malia, but I'm not sure about the other one.
```

These predicted answers are all **INCORRECT** because:
- A factual statement in the answer contradicts the gold target. Incorrect statements that have some hedging (e.g., "it is possible that", "although i'm not sure, i think") are also considered incorrect.
- Or the important information in the gold target is not fully included in the answer (refusal, partial answer, or vague hedge that never commits to the gold target).

```
Question: Which uncontacted people live on North Sentinel Island?
Gold target: Sentinelese
Predicted answer: Jarawa
```

```
This predicted answer is INCORRECT because Jarawa is a different ethnic group, even though
  both are indigenous to the Andaman Islands and are often mentioned together.

```
Question: Which New York City borough contains JFK Airport?
Gold target: Queens
Predicted answer: Brooklyn
```

This predicted answer is INCORRECT because Brooklyn is a different, nearby place — not the
   same borough as the gold target.

Also note the following things:
- Grade based on whether the predicted answer correctly answers the question relative to
  the gold target. Focus on the committed core answer (often the first clear statement of
  the entity/value asked for). Do not mark CORRECT answers INCORRECT merely because the
  response is long or includes extra context, reasoning, verification, or related tables/
  lists after stating the gold target.
- Extra information is fine when it does not replace or contradict the gold target. Extra
  information that names a different primary answer, or that contradicts the gold target,
  makes the prediction INCORRECT.
- For grading questions where the gold target is a number, the numeric value must be
  correct. Do not accept a nearby but different number.
    - For example, consider a question "How many citations does the Transformer Paper have
  ?" with gold target "120k". Predicted answer "120k" is CORRECT. Predicted answers "124k",
   "115k", "100k", "113k", "around 120k", and "more than 50k" are INCORRECT.
    - Units stated in the question may be omitted in the predicted answer. For the question
   "What is the height of Jason Wei in meters?" with gold target "1.73 m", predicted
  answers "1.73 m" and "1.73" are CORRECT; "1.75" and "1.75 m" are INCORRECT.
    - Extra or fewer decimal places are CORRECT only when they round to the same value as
  the gold target (or the gold rounds to the prediction at the gold's precision). For gold
  "1.73", predicted "1.732" is CORRECT; for gold "1.732", predicted "1.73" is CORRECT. A
  different value after rounding (e.g. gold "1.73" vs predicted "1.75") is INCORRECT.
- The gold target may contain more information than the question. In such cases, the
  predicted answer only needs to contain the information that is in the question.
    - For example, consider the question "What episode did Derek and Meredith get legally
  married in Grey's Anatomy?" with gold target "Season 7, Episode 20: White Wedding".
  Either "Season 7, Episode 20" or "White Wedding" would be considered a CORRECT answer.
- Do not punish predicted answers if they omit information that would be clearly inferred
  from the question.
    - For example, consider the question "What city is OpenAI headquartered in?" and the
  gold target "San Francisco, California". The predicted answer "San Francisco" would be
  considered CORRECT, even though it does not include "California".
    - Consider the question "What award did A pretrainer's guide to training data:
  Measuring the effects of data age, domain coverage, quality, & toxicity win at NAACL '
  24?", the gold target is "Outstanding Paper Award". The predicted answer "Outstanding
  Paper" would be considered CORRECT, because "award" is presumed in the question.
    - For the question "What is the name of Barack Obama's wife?", the gold target is "
  Michelle Obama". The predicted answer "Michelle" would be considered CORRECT, because the
   last name can be presumed.
- Do not punish for typos in people's name if it's clearly the same name.
    - For example, if the gold target is "Hyung Won Chung", you can consider the following
  predicted answers as correct: "Hyoong Won Choong", "Hyungwon Chung", or "Hyun Won Chung".
- Related is not the same. A predicted answer that names a different person, place, ethnic
  group, organization, work, or event from the gold target is INCORRECT, even if the two
  are nearby, historically linked, commonly confused, or belong to the same country, family
  , or category.
- A shortened, reordered, or unofficial form of the SAME proper name is CORRECT (dropping a
   middle name, particle, or commonly omitted suffix still identifies the same person or
  entity).
- A former official name, current official name, or well-established alias of the SAME
  entity is CORRECT. Naming a different entity, a successor that is not the same thing, or
  a merely related item is INCORRECT.

Here is a new example. Simply reply with either CORRECT or INCORRECT. Don't apologize or
  correct yourself if there was a mistake; we are just trying to grade the answer.

```
Question: {question}
Gold target: {target}
Predicted answer: {predicted_answer}
```

Grade the predicted answer of this new question as one of:
A: CORRECT
B: INCORRECT

Just return the letters "A" or "B", with no text around it.
```

I FAILURE ANALYSIS OF THE BASE, SFT, AND RL MODELS

This section details the failure analysis behind §6.1. We analyze all 2,880 rollouts of the base, SFT, and RL models on the 120-question hard subset (3 models $\times$ 120 questions $\times$ 8 samples). The three models receive identical prompts, tool schemas, and decoding parameters (Appendix H.3), and a rollout that ends without a `final_answer` call counts as wrong. Appendix I.1 assigns every rollout to exactly one outcome, and Appendix I.2 examines how the agents justify the answers they submit. Of the 120 questions, 35 are never solved by any of the three models (9 `S2SQL`, 21 `SQL2S`, and 5 Parallel).

I.1 OUTCOME OF EVERY ROLLOUT

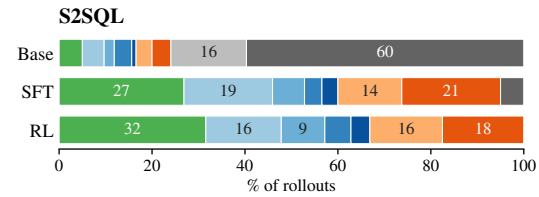
Evidence tracing. We normalize strings by lowercasing them and removing accents and punctuation, and we count a string as present in a text when it matches on whole words. We trace two strings through every rollout, the *bridge* and the *gold answer*. For `S2SQL`, the bridge is the web-resolved entity (plus its database spelling when it differs); for `SQL2S`, it is the first cell of the gold SQL result. For Parallel, the SQL candidates are the rows of the gold SQL result, and the web candidates are the annotated search answers. A string is *surfaced* if it appears in a `web_search` result or a visited page, and *in reasoning* if it appears in the model's reasoning. On `S2SQL`, the model uses the *right entity* if the bridge appears in its final text or submitted SQL. Paraphrases and entities encoded by ID do not match, so all evidence rates are lower bounds.

Outcome categories. Each rollout receives the first category it matches, in this order. `S2SQL`: correct; no answer; prose answer without SQL; SQL fails to execute; wrong entity, with the right one never surfaced; wrong entity, with the right one surfaced; right entity with the wrong row count, the wrong columns, or wrong values. `SQL2S`: correct; no answer; bridge not retrieved by SQL; gold never surfaced; gold surfaced but never in reasoning; gold in reasoning but not answered. Parallel: correct; no answer; gold never in the SQL results; gold never on the web; gold in both sources, with the answer in the web candidates only, in the SQL candidates only, or in neither.

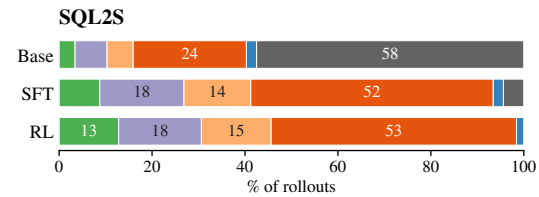
`S2SQL`: wrong entity and wrong SQL. The base model usually fails before it builds a query: 59.7% of its rollouts end without an answer, and 16.3% answer in prose without SQL. The trained models fail for two reasons of similar size. The first is resolving the wrong bridge entity (35.0% of SFT and 33.1% of RL rollouts), often a plausible neighbor of the right one, such as Chad, Egypt, or Niger for Sudan, Pakistan for Chile, or Denmark for Iceland. In 13.8% and 15.6% of rollouts, the right entity does appear in a search result or a visited page, but the query is built on another one. When it appears only in a visited page, it is usually a passing mention in a long page about something else, such as one row of a table of countries. The second is building wrong SQL around the right entity (29.7% and 31.3%), most often with wrong values. Malformed SQL is rare: at most 4.1% of rollouts submit a query that fails to execute.

`SQL2S`: the search hop. The SQL hop succeeds: the bridge never appears in the SQL results in only 2.5% of SFT and 1.6% of RL rollouts. The search hop does not. For both trained models, the largest category is a gold answer that never surfaces (52.2% of SFT and 52.8% of RL rollouts). Counting every category, the gold answer never appears in any search result or visited page in 59.4% and 54.1% of rollouts, and accuracy in those rollouts is at most 2.1%. When the gold answer does surface, the models often fail to use it: in 14.4% and 15.0% of rollouts it never enters the reasoning, and in 18.1% and 17.8% it enters the reasoning but a different answer is submitted. Typical wrong answers are the bridge itself (about 7% of wrong `SQL2S` answers), a salient entity next to the bridge (the capital Moroni for a question whose bridge is the Comoros and whose answer is the village Iconi), and a name copied from a coined bridge ("Jim Taylor" for a bridge named Taylortown).

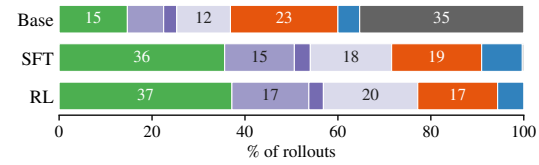
Parallel: coverage and the intersection. In 8.8% of SFT and 5.6% of RL rollouts, the answer is wrong and the gold answer never reaches the SQL results; in a further 19.4% and 17.2%, it reaches the SQL results but never the web. The largest remaining failure is the intersection, which more rollouts reach as coverage improves: the gold answer appears in both sources, but the submitted answer is in only one candidate set (18.4% and 19.7%) or in neither (17.5% and 20.3%). Answers in neither set are often neighbors of the gold entity, such as Tajikistan for Turkmenistan or Greece for Serbia. When the answer satisfies only one side, that side is SQL far more often than the web



Outcome (% of rollouts)	Base	SFT	RL
correct	5.0	26.9	31.6
right entity, wrong values	4.7	19.1	16.3
right entity, wrong row count	2.2	6.9	9.4
right entity, wrong columns	3.8	3.8	5.6
SQL fails to execute	0.9	3.4	4.1
wrong entity; right one surfaced	3.4	13.8	15.6
wrong entity; right one not surfaced	4.1	21.3	17.5
prose answer, no SQL	16.3	0.0	0.0
no answer	59.7	5.0	0.0



Outcome (% of rollouts)	Base	SFT	RL
correct	3.4	8.8	12.8
gold in reasoning, not answered	6.9	18.1	17.8
gold surfaced, never in reasoning	5.6	14.4	15.0
gold never surfaced	24.4	52.2	52.8
bridge not retrieved by SQL	2.2	2.2	1.6
no answer	57.5	4.4	0.0



Outcome (% of rollouts)	Base	SFT	RL
correct	14.7	35.6	37.2
gold in both; answer in SQL set only	7.8	15.0	16.6
gold in both; answer in web set only	2.8	3.4	3.1
gold in both; answer in neither set	11.6	17.5	20.3
gold never on the web	23.1	19.4	17.2
gold never in SQL results	4.7	8.8	5.6
no answer	35.3	0.3	0.0

Figure 5: Outcome of every rollout, in % of the 320 rollouts per model and task type (40 questions $\times$ 8 samples). The categories are mutually exclusive and are assigned by the first matching rule listed under *Outcome categories*; each table lists them in bar order. *Surfaced* means that the string appears in a search result or a visited page. Bar labels are rounded to whole percentages.

(15.0% vs. 3.4% for SFT and 16.6% vs. 3.1% for RL), which suggests that the models trust the structured list and do not verify the web constraints.

Repeated errors and judge noise. Wrong answers repeat across samples: for the trained models, a single wrong answer covers at least 4 of 8 samples on up to 18 of 40 `SQL2S` questions and up to 14 of 40 Parallel questions. Judge noise is negligible: only 0.87%, 0.29%, and 0.15% of wrong base, SFT, and RL rollouts receive a split judge vote (1–4 of 5 votes correct).

Base model. Half of the base model's rollouts (50.8%) end without an answer. In 43.2% of all its rollouts, the last turn is a single reasoning turn that exhausts the 32K-token limit without calling a tool, and in 412 of these 415 turns, fewer than half of the sentences are unique. In one `SQL2S` rollout, the last turn contains 1,564 sentences, of which only 38 are distinct.

I.2 Commits justified by time, effort, or a guess

Setup. For every evaluation rollout that submits an answer (a commit: 472 base, 929 SFT, 960 RL), we flag the commit when its rationale, the last three assistant messages up to and including the `final_answer` call, **(1) cites a time or turn budget**, **(2) cites the effort already spent**, or **(3) calls the answer a guess**. The flag is a deterministic, case-insensitive phrase match (Table 16 lists frequent phrases).

Method. We compare the accuracy of flagged and unflagged commits within the same question, using questions where a model has both correct and wrong commits and both flagged and unflagged ones, weighting each question equally, with 95% bootstrap intervals over questions. We then fit a regression with one intercept per question that also controls for whether the target (the gold answer, or the bridge entity for `S2SQL`) had appeared in a tool result or in the agent's reasoning, and for rollout length.

Table 16: Frequent phrases that flag a commit, per family, in SFT and RL commits. Matching is deterministic, case-insensitive and on whole words, over the reasoning and visible text of the last three assistant messages up to and including the `final_answer` call; rows show frequent matched phrases, not the full pattern list.

Family	Frequent phrases
(1) Time or turn budget	"given the time constraints", "time constraints", "running out of time", "turn budget"
(2) Effort already spent	"given all my research", "spent enough time", "given the extensive searching", "spent a lot of time", "given all the analysis"
(3) Calls the answer a guess	"best guess", "make an educated guess", "make my best guess", "make a guess"

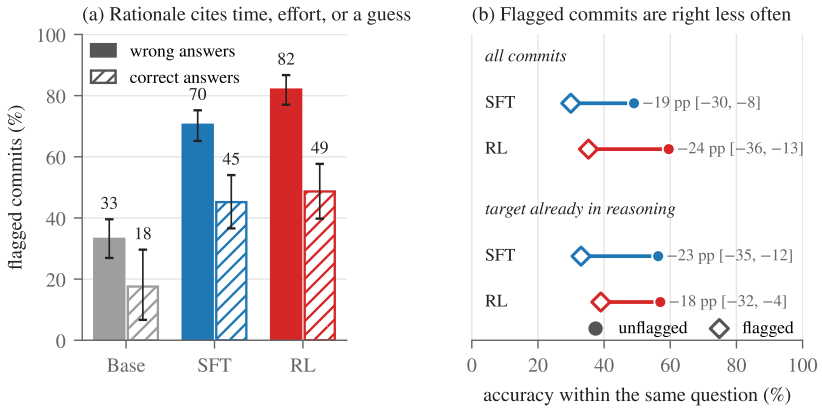


Figure 6: A commit is flagged when its rationale (the last three assistant messages up to and including the `final_answer` call) **(1) cites a time or turn budget**, **(2) cites the effort already spent**, or **(3) calls the answer a guess**. (a) Share of wrong (solid) and correct (hatched) commits that are flagged, pooled over rollouts; whiskers are 95% bootstrap intervals over questions. (b) Accuracy of flagged (diamond) and unflagged (dot) commits within the same question (differences in percentage points, pp), for all commits and for commits where an assistant message up to the commit already named the target (the gold answer, or the bridge entity for `S2SQL`).

Results. 82.0% of RL's wrong commits are flagged against 48.7% of its correct ones (SFT: 70.5% vs 45.2%; Figure 6, Table 17), and within the same question flagged commits are 24.3 points less often correct than unflagged ones for RL (SFT: 19.1); the gap stays negative after these controls (RL −10.0 [−16.2, −3.1], SFT −7.9 [−13.2, −1.8] points). The turn budget does not force the stop: the prompt calls it only "limited", the median SFT and RL rollouts end after 44 and 41 of 300 turns, and no SFT or RL rollout reaches the reminder sent with 10 turns left.

Conclusion. Both trained agents often justify a submission by a time or turn budget, the effort already spent, or a guess, and such submissions are more often wrong; RL makes them more common (72.9% vs 64.3% of commits) with no detectable change in the gap after controls (RL − SFT: −2.1 [−11.0, +7.2] points), and 28.6% of the SFT training trajectories on hybrid questions, which were kept only when their answer was correct, already contain them. Since the turn budget does not stop the agent, a longer budget alone is unlikely to help; a promising direction is to make submission depend on evidence, for example through a verification step triggered by the flag.

Table 17: Supporting numbers for Figure 6. (a) Coefficient on the flag in a linear probability model with one intercept per question over all commits, controlling for whether the target appeared in a tool result or in the agent's reasoning, and for log turns, context length and output tokens; the RL − SFT difference is −2.1 [−11.0, +7.2]. (b) `SQL2S` and Parallel rollouts that keep working after the first assistant message containing a flag phrase: share correct at submission minus the share whose most recently named answer was correct at that message (observational). The target is the gold answer (for `S2SQL`, the bridge entity); it is in the agent's reasoning if an assistant message up to the commit names it. The first two rows of the middle block use, for each model, the questions whose commits in that row are both correct and wrong and both flagged and unflagged (40–57 questions per estimate); intervals are 95% bootstrap intervals over questions. The 120 questions include 12 verbatim-duplicate pairs; clustering them barely changes the estimates. Base submits an answer in only 49.2% of rollouts, so its rates describe a selected subset, and the other rows are not estimated for it (–).

	Base	**SFT**	**RL**
Flagged commits (%)			
All commits	30.7	64.3	72.9
Wrong commits	33.2	70.5	82.0
Correct commits	17.6	45.2	48.7
Accuracy of flagged minus unflagged commits, same question (points)			
All commits	–	−19.1 [−30.2, −7.7]	−24.3 [−36.0, −12.9]
Target already in the agent's reasoning	–	−23.3 [−34.9, −11.5]	−18.0 [−31.5, −4.1]
After controls for evidence and length (a)	–	−7.9 [−13.2, −1.8]	−10.0 [−16.2, −3.1]
Budget and continuation			
Rollouts that submit an answer (%)	49.2	96.8	100.0
Median turns per rollout (of 300)	57	44	41
Rollouts reaching the 10-turns-left reminder	15/960	0/960	0/960
Continuing past the first flag (points) (b)	–	−3.7 [−8.9, +0.8]	−2.0 [−6.5, +2.2]

J RL TRAINING AND ANALYSIS

J.1 RL TRAINING PARAMETERS

RL training is conducted using framework built on PRIME-RL (Senghaas et al., 2025), with fully asynchronous training at asynchrony level 1, running on a single 8-GPU node. We optimize a GRPO objective (Shao et al., 2024) with standard-deviation-normalized advantages and prompt-level token-averaged loss, using CISPO (MiniMax et al., 2025) with importance weights clipped at 2.0 and negative-sample scale 1.0. We used AdamW (Loshchilov & Hutter, 2019) with learning rate 1×10^{-6}, $\epsilon = 1 \times 10^{-15}$, $\beta_1 = 0.9$, $\beta_2 = 0.98$, zero weight decay, and a constant schedule without warmup. The reference-model KL penalty and explicit length penalty are disabled. Each step uses 32 prompts with group size 5. Rollouts use temperature 1.0, top-$p = 1.0$, up to 32k completion tokens per turn and max turns set to 100, with a 128k token context limit. Inference uses vLLM (Kwon et al., 2023) with tensor parallelism 1 and 95% GPU memory utilization.

We additionally perform group-level difficulty filtering for adapting dynamic policy. Groups with 0% or 100% solve rates are excluded from the main optimization stream because they provide little within-group reward contrast. A task is removed from the training pool after being fully solved in $n = 1$ consecutive rollout groups or failing completely in $m = 3$ consecutive rollout groups.

J.2 JUDGE VOTING ANALYSIS

As shown in Table 18, representative examples illustrate how five-vote majority judging resolves challenging cases involving semantic ambiguity and non-trivial variations in SQL result tables.

Table 18: Representative cases where single-vote judging produces an incorrect reward while five-vote majority judging reaches the correct judgment. The SQL example illustrates that judging requires reasoning about result-level semantics rather than exact table matching.

<table>
<tr><th>Case</th><th>Example</th></tr>
<tr><td>1</td><td>Entity ambiguity. The reference entity is <code>Charles Emmanuel III</code>, while the model produces <code>"Charles III"</code>. Although the two names are lexically similar, they refer to different monarchs. A correct judgment therefore requires semantic entity resolution rather than string matching.</td></tr>
<tr><td>2</td><td>Partial SQL correctness. The gold result filters to <code>Fort de Pontisse</code>, while the model recovers the same ranking but omits the final filter.
<table>
<tr><th colspan="3">Gold result</th><th colspan="3">Model result</th></tr>
<tr><th>fort</th><th>avg_days_held</th><th>rank</th><th>fort</th><th>avg_days</th><th>rank</th></tr>
<tr><td>Fort de Pontisse</td><td>6.0</td><td>2</td><td>Fort de Loncin</td><td>7.0</td><td>1</td></tr>
<tr><td></td><td></td><td></td><td>Fort de Pontisse</td><td>6.0</td><td>2</td></tr>
<tr><td></td><td></td><td></td><td>Fort de Fléron</td><td>5.5</td><td>3</td></tr>
<tr><td></td><td></td><td></td><td>Fort de Flémalle</td><td>3.5</td><td>4</td></tr>
<tr><td></td><td></td><td></td><td>Fort de Barchon</td><td>3.5</td><td>4</td></tr>
</table>
The model correctly computes the averages and ranking, and correctly places <code>Fort de Pontisse</code> at rank 2, but fails to apply the final filter. The final result is therefore incorrect despite substantial partial correctness, making the case difficult to judge reliably from surface-level table comparison.</td></tr>
</table>