

EHR-Complex: Benchmarking Medical Agents for Complex Clinical Reasoning

Yitong Qiao^{1*†}, Lei Liu^{1,2*}, Yue Shen², Jian Wang², Jinjie Gu², Zhixuan Chu^{1‡}, Kui Ren¹

¹The State Key Laboratory of Blockchain and Data Security, Zhejiang University

²Ant Healthcare, Ant Group

Corresponding author: zhixuanchu@zju.edu.cn

Abstract

Clinical agents promise to democratize access to electronic health records (EHRs), yet existing benchmarks fail to reflect the complexity of practical EHR analysis, *e.g.*, often operating on idealized, clean EHRs via static SQL generation rather than interactive execution. In this work, we introduce **EHR-Complex**, a large-scale benchmark designed for interactive clinical database reasoning. Built on the large MIMIC-IV substrate (365K patients, 31 tables, 500M+ records), EHR-Complex comprises about 52K tasks spanning six clinical intents, supporting both patient-level and population-level queries, where each task requires an agent to interact with a sandboxed environment by executing SQL queries or Python code. Notably, EHR-Complex considers the real-world SQL task complexity for longitudinal multi-table aggregation and compositional reasoning, resulting in 31.93 SQL structural components per query on average. Evaluation results on EHR-Complex reveal the clinical difficulty of these EHR reasoning scenarios, with the top-performing model achieving only 62.3% exact-match accuracy. Pass^k consistency drops below 50% for nearly all evaluated models at $k=4$, exposing broad stochastic fragility. A fine-grained analysis of more than 3,800 failed trajectories for representative LLMs reveals three dominant failure modes: SQL logic errors, medical-code lookup failures, and semantic misunderstandings. EHR-Complex provides a rigorous testbed for clinical agents and highlights remaining gaps in robust reasoning for large-scale EHR analysis.

1 Introduction

Large language models (LLMs) and agents are increasingly viewed as promising interfaces for democratizing access to electronic health records

(EHRs), potentially enabling clinicians and healthcare administrators to retrieve and analyze patient data through natural language rather than specialized database expertise (Wang et al., 2020; Lee et al., 2022; Xu et al., 2026; Jiang et al., 2025; Lee et al., 2026). Such capability is especially valuable in modern clinical environments, where EHR databases are large, heterogeneous, longitudinal, and difficult to query efficiently. Clinical agents could substantially reduce the technical barrier to complex cohort construction, patient-level record retrieval, and retrospective outcome analysis.

Prior work on executable agent benchmarks, such as SWE-bench (Jimenez et al., 2024), WebArena (Zhou et al., 2024), and MLAgentBench (Huang et al., 2024), has shown that grounding agents in environments with execution feedback can expose failure modes that static evaluation may miss. Despite this promise, current evaluation settings for clinical database reasoning still only partially capture the complexity of practical EHR analysis. However, existing benchmarks often simplify the problem into one-shot text-to-SQL generation over clean, idealized schemas, where the model is evaluated only on a final static query string.

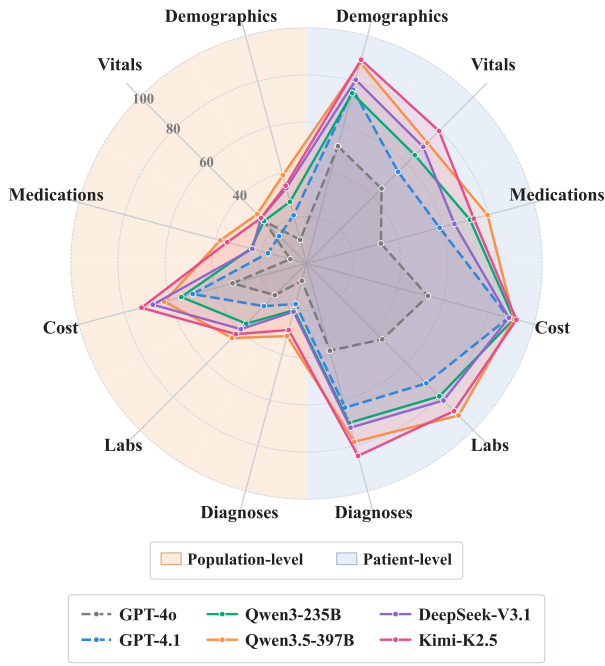
Practical EHR analysis often requires agents to operate in interactive environments rather than merely produce syntactically valid single-shot SQL statements: users must inspect intermediate results, resolve schema ambiguities, identify relevant medical codes, perform longitudinal aggregation across multiple tables, and iteratively refine logic when initial queries fail or return unexpected outputs. Moreover, large EHR databases are noisy, expansive, and semantically challenging, requiring models to reason jointly over temporal structure, coding systems, and compositional constraints.

To this end, we introduce **EHR-Complex**, a large-scale benchmark for interactive clinical database reasoning. Built on the full MIMIC-IV v3.1 substrate (Johnson et al., 2024), containing

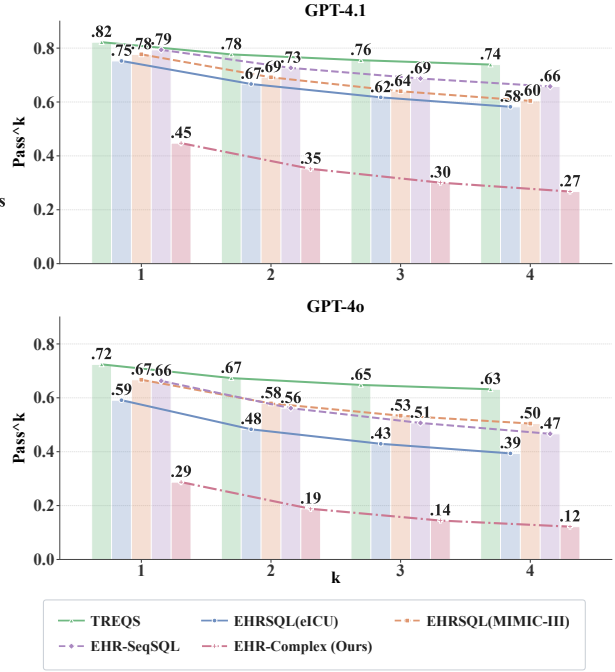
*Equal contribution.

†Work done during an internship at Ant Group.

‡Corresponding author.



(a) Success rates across 12 (6 intent $\times$ 2 scope) dimensions.



(b) Pass^k across different benchmarks.

Figure 1: **Overall Performance of LLMs on EHR-Complex Test Set.** (a) Performance across 12 (6 intent $\times$ 2 scope) dimensions. Patient-level queries typically follow one clinical evidence path, whereas population-level queries require aligning and aggregating evidence paths across patients. Any misaligned path may lead to an incorrect result. (b) Evaluated models achieve near-perfect Pass^k on legacy benchmarks but struggle on EHR-Complex.

365K patients, **31** tables, and more than **500 million** records, EHR-Complex consists of approximately **52K** tasks spanning six major clinical intents.

The benchmark supports both patient-level and population-level information needs and requires agents to operate in a sandboxed environment by executing SQL queries or Python code, rather than merely generating static answers. Importantly, EHR-Complex is designed to reflect the compositional and longitudinal nature of large-scale EHR analysis: benchmark tasks require substantial multi-table reasoning and aggregation, with queries averaging 31.93 SQL structural components per query. Our empirical evaluation across a range of state-of-the-art LLMs shows that EHR-Complex remains highly challenging. Even the best-performing model achieves 62.3% exact-match accuracy, indicating a substantial gap between current capabilities and the robustness needed for dependable EHR analysis. Overall contributions are as follows:

- We introduce a large-scale EHR-Complex benchmark for interactive clinical database reasoning. Built on the full MIMIC-IV substrate, EHR-

Complex comprises approximately 52K tasks across six clinical intents, covering both patient-level and population-level scopes.

- We propose to incorporate real-world SQL task complexity into EHR-Complex, *e.g.*, longitudinal multi-table aggregation and compositional reasoning. Each task is paired with an execution-validated SQL program and a verified answer, with queries averaging 31.93 SQL structural components per query.
- We evaluate 12 LLM agents and analyze their robustness and failure modes, showing persistent challenges in population-level reasoning, medical-code grounding, and SQL logic. Trajectory SFT improves in-domain performance and transfers to unseen templates and clinical intents.

2 Related Work

Static Text-to-SQL. Early efforts to enable natural-language access to EHRs primarily focused on semantic parsing, inheriting the cross-schema evaluation established by general-domain

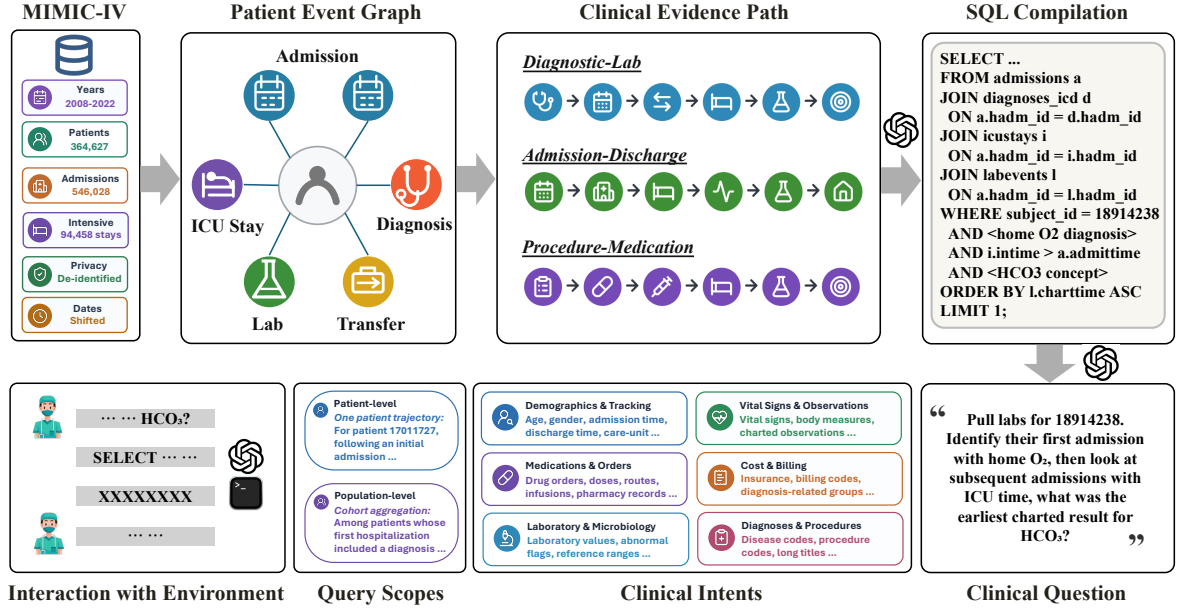


Figure 2: EHR-Complex Construction Pipeline. **Step-1:** Based on the full MIMIC-IV substrate, patient event graphs are derived for the subsequent compositional reasoning. **Step-2:** Clinical evidence paths are extracted from event graphs as SQL query candidates. **Step-3:** Clinical evidence paths are compiled into execution-validated SQL templates. These templates are materialized into patient- and population-level questions across six clinical intents. EHR-Complex provides the environment sandboxes for interactive SQL/Python execution.

benchmarks like Spider (Yu et al., 2018) while adapting it for domain-specific medical grounding and temporal logic. Foundational clinical datasets such as MIMICSQL (Wang et al., 2020) and emrKBQA (Raghavan et al., 2021) mapped medical concepts to schemas, while later works like EHRSQL (Lee et al., 2022) and EHR-SeqSQL (Ryu et al., 2024) introduced hospital-derived questions with time-sensitive templates. Although these resources advanced clinically grounded text-to-SQL evaluation, they remain constrained by a static generation paradigm that lacks execution feedback. Furthermore, to ensure tractability, they typically rely on heavily filtered patient subsets ($<1K$) and fewer schemas (5-17 tables).

Medical Agent Benchmarks. Inspired by general-domain interactive benchmarks like SWE-bench (Jimenez et al., 2024), WebArena (Zhou et al., 2024), and MLAGentBench (Huang et al., 2024), recent efforts have reframed EHR access as a multi-turn agentic task. Frameworks such as EHRAgent (Shi et al., 2024), FHIR-AgentBench (Lee et al., 2026), MedAgentGym (Xu et al., 2026), and MedAgentBench (Jiang et al., 2025) now provide executable environments for iterative querying, marking a pivotal shift

from static generation to interactive problem-solving. Related non-medical benchmarks (e.g., τ -bench (Yao et al., 2025), WorkArena (Drouin et al., 2024)) similarly emphasize grounded interaction, whereas quantitative medical benchmarks such as MedCalc-Bench (Khandekar et al., 2024) evaluate the complementary capability of clinical calculation and quantitative reasoning. While these benchmarks mark important progress toward grounded interaction, many interactive medical benchmarks build on tasks adapted from legacy static datasets, such as EHRSQL (Lee et al., 2022) and TREQS (Wang et al., 2020), and thus center on iterative revision of a single SQL query over smaller database substrates. EHR-Complex complements these settings with full-database reasoning in which agents decompose clinical information needs into dependent SQL/Python steps over longitudinal patient records and population-level cohorts. It also provides consistent coverage across clinical intents and query scopes, enabling trajectory-level analysis of EHR-specific failures.

3 EHR-Complex Benchmark

Task Formulation. As illustrated in Figure 2, given a natural-language query $x \in \mathcal{X}$ over a

Table 1: Comparison with Existing EHR Benchmarks. EHR-Complex reflects real-world clinical complexity, *e.g.*, longitudinal multi-table aggregation and compositional reasoning.

Dataset	Agentic	Database Scale			Task Instances			Task
	Mode ²	#Patients	#Tables	#Elements	Category	#Test	#Train	Complexity ³
MedAgentBench ¹ (Jiang et al., 2025)	✓	100	–	700K	10	300	–	–
TREQS (Wang et al., 2020)	✗	100	5	2.5M	4	996	8,988	3.06
EHRSQL (eICU) (Lee et al., 2022)	✗	<1K	10	1.5M	9	611	6,213	11.21
EHRSQL (MIMIC-III) (Lee et al., 2022)	✗	<1K	17	1.4M	9	1,122	9,318	11.30
EHR-SeqSQL (Ryu et al., 2024)	✗	<1K	17	1.4M	4	7,913	18,950	17.03
EHR-Complex (Ours)	✓	365K	31	>500M	12	3,915	48,092	31.93

¹ MedAgentBench operates in an API environment, making it incompatible with SQL complexity metrics.

² Agentic mode indicates whether a benchmark supports executable interaction with an EHR environment.

³ Average total number of SQL structural components per query. Figure 3 shows the breakdown.

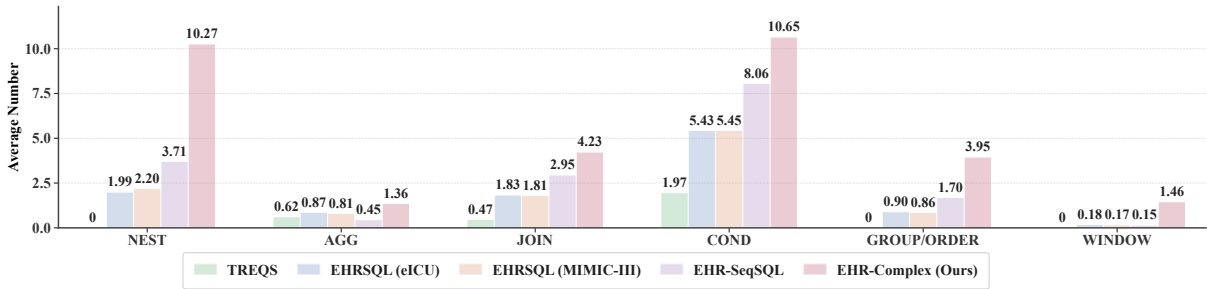


Figure 3: Average Number of SQL Structural Components Per Query Across Different Benchmarks. EHR-Complex exhibits higher task complexity for multi-table aggregation and compositional reasoning.

database $\mathcal{D}$ with schema Σ , an agent produces a multi-turn trajectory $\tau \in \mathcal{T}$ by executing SQL queries or Python code and observing environment feedback. The objective is to derive the target output $y \in \mathcal{Y}$ that matches the verified answer y^* .

For each trajectory, we define correctness as $\mathcal{E}(\tau, y^*) = \mathbb{I}(\text{Norm}(y) = \text{Norm}(y^*))$, where y denotes the final answer extracted from trajectory τ , y^* denotes the ground-truth output, and $\text{Norm}(\cdot)$ normalizes superficial variations such as capitalization. For ranked outputs, the verified answer follows deterministic tie-breaking during reference-answer extraction.

Unlike existing EHR benchmarks (Wang et al., 2020; Lee et al., 2022; Ryu et al., 2024) with static question-answer pairs (x, y^*) or question-SQL pairs (x, q^*) , EHR-Complex derives the desired answer via multi-round interaction between the agent and the environment, *i.e.*, EHR-Complex facilitates the scalable generation and sampling of diverse SQL trajectories $\{\tau^{(0)}, \dots, \tau^{(k)}\}$ with corresponding outputs $\{y^{(0)}, \dots, y^{(k)}\}$ via repeated independent agent rollouts.

3.1 Benchmark Statistics

Intent and Scope Taxonomy. EHR-Complex focuses on verifiable structured EHR tasks grounded in clinically motivated information needs and executable database evidence. Building on query types established in prior benchmarks (Wang et al., 2020; Lee et al., 2022; Ryu et al., 2024; Jiang et al., 2025), we extend the scope to the full MIMIC-IV substrate across six clinical intents and two query scopes:

- **Intent:** Demographics & Tracking (Demographics), Vital Signs & Observations (Vitals), Medications & Orders (Medications), Cost & Billing (Cost), Labs & Microbiology (Labs), and Diagnoses & Procedures (Diagnoses).
- **Scope:** Patient-level (Pat.) and population-level (Pop.), corresponding to single-patient trajectory reasoning and cohort-level aggregation across patients, respectively.

Table 2 summarizes the EHR-Complex test-set distribution across these intents and query scopes. The full benchmark contains 48,092 training tasks and 3,915 test tasks. All reported evaluations are conducted on the test set.

SQL Task Complexity. We quantify SQL task complexity using the average number of SQL struc-

Table 2: Task Distribution of EHR-Complex Test Set.

Clinical intent	Query scope		Total
	Pop.	Pat.	
Labs	835	670	1,505
Demographics	491	460	951
Diagnoses	410	240	650
Medications	324	260	584
Cost	40	90	130
Vitals	55	40	95
Total	2,155	1,760	3,915

tural components in queries. Component definitions and details are provided in Appendix C. As shown in Table 1 and Figure 3, EHR-Complex averages 31.93 components per query, reflecting the real-world SQL task complexity required by longitudinal multi-table aggregation and compositional reasoning.

3.2 Data Curation Pipeline

We formulate task construction as a grounded program synthesis problem over longitudinal EHR data. Let the underlying EHR database be denoted by $\mathcal{D}$. Our goal is to construct a synthesis mapping $f : \mathcal{D} \rightarrow (\mathcal{Q}, \mathcal{A}_{\text{SQL}})$ that transforms raw patient records into a set of executable clinical queries $\mathcal{Q}$ and corresponding verifiable answers $\mathcal{A}_{\text{SQL}}$.

Step-1: Patient Event Graph. To provide a unified substrate for compositional reasoning over patient trajectories, we induce a patient event graph from MIMIC-IV, serving as an intermediate structured representation for task generation. Formally, for each patient p , we construct a heterogeneous temporal graph $G_p = (V_p, E_p, \phi_V, \phi_E)$, where V_p denotes clinically meaningful event nodes, $E_p \subseteq V_p \times V_p$ denotes typed relations between events, and ϕ_V, ϕ_E are node-type and edge-type mappings, respectively. In this graph, heterogeneous records from admissions, ICU stays, laboratory events, diagnoses, prescriptions, and charted observations are normalized into event nodes $v = (c, t, a)$, where c denotes the clinical concept, t the timestamp, and a the associated attributes. Temporal and semantic dependencies are then represented as edges $e = (v_i, r, v_j)$, where $r \in \mathcal{R}$ and $\mathcal{R}$ is a predefined set of relation types such as precedes, indicates, or treated_by.

Step-2: Clinical Evidence Path. This step focuses on sampling concrete clinical evidence paths

from the patient event graph, referring to a temporally ordered chain of clinically related events that supports a candidate query. An evidence path is defined as an ordered sequence

$$P = (v_1, v_2, \dots, v_k), \quad (v_i, v_{i+1}) \in E_p,$$

where $t(v_1) \leq t(v_2) \leq \dots \leq t(v_k)$ and the path satisfies both temporal consistency and clinical plausibility constraints. This path constructs a reasoning chain that adheres to clinical logic, encompassing diverse clinical events within the patient event graph, such as ICU stays, laboratory events, and charted observations. Hence, the system can not only track the timeline but also flexibly query clinical details at any moment. This formulation induces: (1) groundedness, every task is tied to an explicit chain of evidence in the source EHR, and (2) compositionality, complex tasks are generated by composing reusable reasoning operators over event paths instead of enumerating surface-form question patterns.

Step-3: Evidence-to-SQL Template. The clinical evidence paths are compiled into executable SQL templates using strong LLMs via a mapping $g : P \xrightarrow{\text{LLM}} s$, where P is an evidence path and $s \in \mathcal{S}$ is a SQL program that operationalizes the intended reasoning process over $\mathcal{D}$. A patient-level query typically follows an evidence path for a single patient, whereas a population-level query is derived by aligning the same evidence-path pattern across multiple patients and aggregating the resulting evidence. The synthesized query is then checked by database execution $\text{Exec}(s, \mathcal{D})$. Specifically, we keep queries that execute successfully, return non-empty results, and admit deterministic answer extraction from the underlying records. This compilation step grounds high-level reasoning operators in database-level retrieval logic, while the subsequent execution check ensures that only well-formed tasks are retained.

Template-based Scaling Up Data curation pipeline produces 796 structurally unique templates that are all validated by execution. We expand each template along two dimensions. First, we instantiate it in multiple concrete database contexts by resampling eligible patients and task-specific parameters such as time windows, diagnosis codes, laboratory items, medications, and target admissions, while keeping the underlying reasoning pattern fixed. Second, for each executable instance,

Table 3: Evaluation Results on the EHR-Complex Test Set.

Model	Demographics		Vitals		Medications		Cost		Labs		Diagnoses		Avg.
	Pat.	Pop.	Pat.	Pop.	Pat.	Pop.	Pat.	Pop.	Pat.	Pop.	Pat.	Pop.	
Proprietary API Models													
GPT-4o	0.52	0.1	0.45	0.25	0.33	0.07	0.53	0.33	0.45	0.19	0.38	0.08	0.31
GPT-4.1 mini	0.75	0.27	0.6	0.22	0.67	0.21	0.88	0.6	0.65	0.3	0.59	0.17	0.49
GPT-4.1	0.76	0.21	0.55	0.16	0.58	0.17	0.89	0.5	0.72	0.26	0.63	0.18	0.47
Gemini 2.5 Pro	0.44	0.19	0.17	0.2	0.43	0.15	0.59	0.38	0.38	0.27	0.34	0.15	0.31
Open-Weight Models ≥100B													
Qwen3-235B	0.75	0.27	0.65	0.25	0.72	0.25	0.91	0.55	0.8	0.36	0.7	0.2	0.53
Qwen3.5-397B	0.88	0.39	0.72	0.3	0.8	0.38	0.91	0.63	0.91	0.45	0.78	0.32	0.62
DeepSeek-V3.1	0.81	0.33	0.7	0.27	0.65	0.24	0.89	0.67	0.82	0.39	0.72	0.21	0.56
DeepSeek-V3.2-Exp	0.83	0.36	0.75	0.27	0.7	0.3	0.88	0.65	0.89	0.44	0.77	0.26	0.59
Kimi-K2.5	0.89	0.34	0.8	0.27	0.73	0.35	0.92	0.72	0.89	0.42	0.84	0.29	0.62
Open-Weight Models <100B													
Qwen3-4B	0.24	0.07	0.28	0.11	0.18	0.01	0.48	0.12	0.17	0.06	0.16	0.03	0.16
Qwen3-14B	0.44	0.14	0.2	0.18	0.43	0.15	0.63	0.33	0.4	0.19	0.39	0.1	0.3
Qwen3-14B-SFT	0.69	0.22	0.42	0.14	0.6	0.19	0.84	0.51	0.74	0.22	0.63	0.18	0.45
Qwen3-32B	0.51	0.2	0.3	0.24	0.47	0.2	0.73	0.35	0.5	0.21	0.42	0.15	0.36
Qwen3-32B-SFT	0.83	0.36	0.55	0.29	0.72	0.3	0.86	0.55	0.8	0.34	0.69	0.26	0.55

Green: Models are fine-tuned on trajectories distilled from EHR-Complex train set (Section 4.4).

we generate multiple independent natural language rewritings. This combination of structural reuse, patient-grounded variation, and linguistic diversity yields approximately 52K tasks, and every instance remains paired with an executable query and a verified exact answer.

Human Validation. We conduct a two-stage human validation study on the test set to assess the semantic validity of EHR-Complex tasks. First, we run four strong commercial model configurations on the full test set: GPT-5.4 (OpenAI, 2026) with high and low reasoning effort, Gemini 3.1 Pro (Google Gemini Team, 2026), and Claude Sonnet 4.6 (Anthropic, 2026). Each model independently interprets the natural-language query, interacts with the environment, and submits a final answer. Among the 3,915 test tasks, 954 are not solved by any of these configurations, forming a challenging candidate set for human review.

Expert annotators then review a stratified sample of 500 tasks from this candidate set, focusing on *clinical plausibility* and *semantic alignment* rather than syntax alone. Specifically, they assess whether the natural-language question faithfully reflects the clinical intent encoded by the SQL and whether the verified answer is supported by the retrieved

database evidence. The audit finds that 95.0% of these challenging sampled tasks are valid after adjudication. Because the sample is drawn from tasks unsolved by all four model configurations, this rate provides a conservative stress test of benchmark validity. More details are reported in Appendix E.

4 Experiments

4.1 Experimental Setup

We evaluate 12 frontier LLMs on the EHR-Complex test set of 3,915 tasks. The closed-source APIs are GPT-4o-2024-11-20 (GPT-4o) (OpenAI, 2024), GPT-4.1-mini-2025-04-14 (GPT-4.1 mini), GPT-4.1-2025-04-14 (GPT-4.1) (OpenAI, 2025), and Gemini-2.5-Pro (Gemini 2.5 Pro) (Kavukcuoglu, 2025). The open-weight models ($\geq 100B$) are Qwen3-235B-A22B-Instruct-2507 (Qwen3-235B) (Qwen Team, 2025a), Qwen3.5-397B-A17B (Qwen3.5-397B) (Qwen Team, 2026), DeepSeek-V3.1 (DeepSeek-AI, 2025a), DeepSeek-V3.2-Exp (DeepSeek-AI, 2025b), and Kimi-K2.5 (Kimi Team, 2026). The efficient-scale open-weight models ($< 100B$) are Qwen3-4B, Qwen3-14B, and Qwen3-32B (Qwen Team, 2025b). All runs use a temperature of 0 and a maximum budget of 50 interaction turns, with each turn allowing one attempt to execute

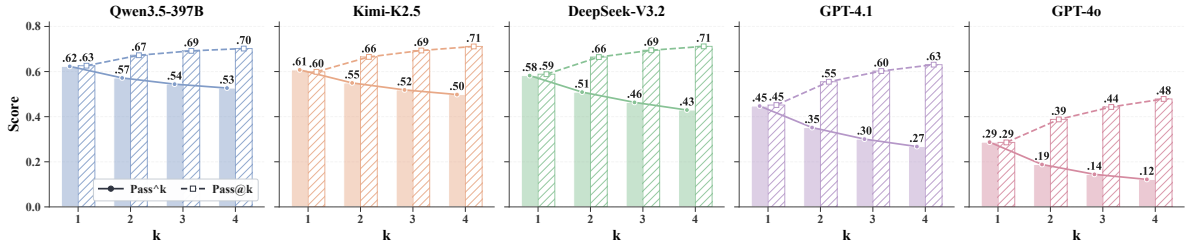
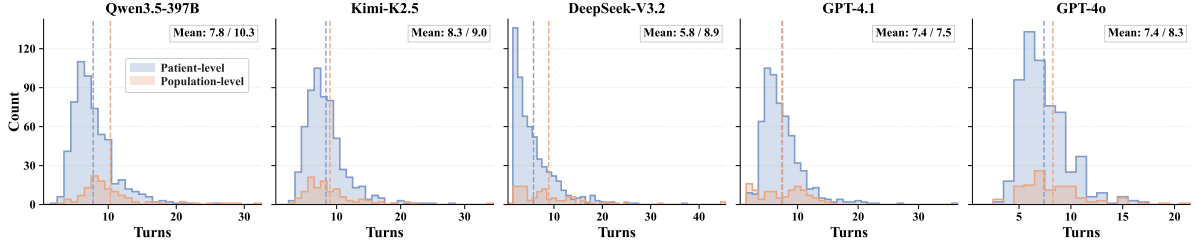
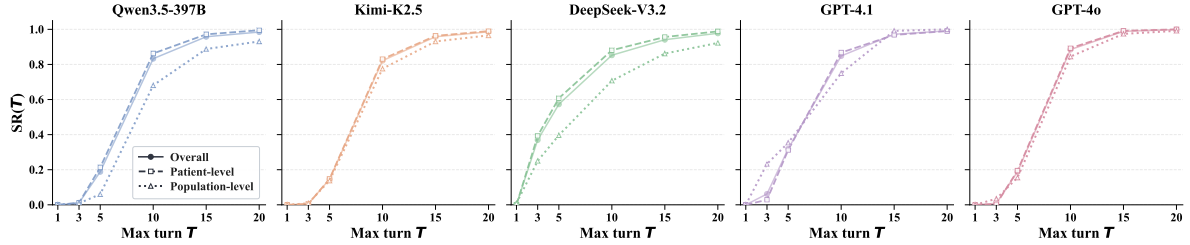


Figure 4: Pass^k and Pass@k at k = 1, 2, 3, 4.



(a) Turn Count Distribution across Successful Tasks.



(b) Success Rate (SR) When Limiting Max Turn T .

Figure 5: Analysis on 712 Successful Tasks on EHR-Complex Test Set. These 712 trajectories are obtained by the intersection of the successful trajectories from five models, with the maximum turn budget set to 50.

SQL/Python against the database and receive feedback. Task success is measured by normalized exact match against the verified ground-truth answer. More implementation details are reported in Appendix A.

4.2 Main Results

Overall Performance. Table 3 reports evaluation success rates on EHR-Complex. Among 12 models evaluated without any fine-tuning, Kimi-K2.5 and Qwen3.5-397B achieve the highest macro-averaged scores, both around 0.62, leaving substantial room for improvement. Among proprietary models, GPT-4.1 mini and GPT-4.1 reach 0.49 and 0.47, respectively, while GPT-4o and Gemini 2.5 Pro both remain at 0.31. Performance also drops sharply at a smaller scale, with Qwen3-4B achieving only 0.16. These results suggest that EHR-Complex remains a challenging execution-based benchmark for current clinical agents.

Performance Gap between Patient- and Population-Level. Averaging over clinical intents, Kimi-K2.5 drops from 0.85 to 0.40, Qwen3.5-397B from 0.83 to 0.41, and GPT-4.1 from 0.69 to 0.25 when moving from patient-level to population-level queries. The gap is particularly pronounced for Diagnoses, where temporally grounded constraints are more common: Kimi-K2.5 drops from 0.85 to 0.29, and GPT-4.1 from 0.63 to 0.18. This suggests that population-level queries introduce failures beyond single-patient reasoning: small errors in per-patient alignment, code grounding, or filtering can silently change the cohort before aggregation. Appendix F gives representative examples.

Cross Intent Analysis. At the patient level, Cost is among the highest-performing intents for strong models, with Kimi-K2.5 and GPT-4.1 reaching 0.92 and 0.89, respectively. Many Cost questions involve identifying the relevant admission

or follow-up stay and retrieving admission-level attributes such as insurance. Demographics and Labs also show relatively high success rates, but they require broader schema navigation: models must align admissions over time before accessing transfer, laboratory, or microbiology records. Vitals and Medications show lower success rates, as they involve denser event streams, temporal extrema, ordered measurements or medications, dosage constraints, and caregiver- or order-level aggregation. At the population level, Cost remains comparatively strong, with Kimi-K2.5 at 0.73 and GPT-4.1 at 0.50. By contrast, Vitals, Diagnoses, and Medications are among the lowest-performing intents for Kimi-K2.5, at 0.27, 0.29, and 0.35, respectively.

4.3 Additional Analysis

Pass^k and Pass@k Beyond the full-test evaluation in Table 3, we study whether model success is stable across repeated independent rollouts. Because repeated execution is costly, we compute Pass^k consistency and Pass@k on a stratified 20% subsample of the EHR-Complex test set, comprising 783 tasks across nine representative models. Pass^k is evaluated at temperature 0.0 and requires all k rollouts to be correct, whereas Pass@k is evaluated at temperature 0.6 and requires at least one correct rollout, for k = 1, 2, 3, 4.

Figure 4 shows that multiple rollouts improve best-case success under Pass@k while exposing instability under Pass^k. Pass@k increases from 0.63 to 0.70 for Qwen3.5-397B and from 0.60 to 0.71 for Kimi-K2.5, indicating that some tasks are solved only when multiple trajectories are sampled. By contrast, Pass^k decreases with k: strong models drop from roughly 0.62 at k= 1 to below 0.53 at k= 4. These results suggest that execution-based EHR reasoning remains sensitive to rollout-level variation, where small changes in temporal alignment, cohort definition, or aggregation logic can yield plausible but incorrect final answers. Full model-, scope-, and intent-level results are reported in Appendix H.

Interaction Turn Analysis We analyze the 712 EHR-Complex test tasks solved correctly by all five representative models. Because all five models solve these tasks under the 50-turn limit, this analysis isolates interaction efficiency rather than final solvability.

Figure 5(a) shows that population-level tasks generally require more turns. For Qwen3.5-

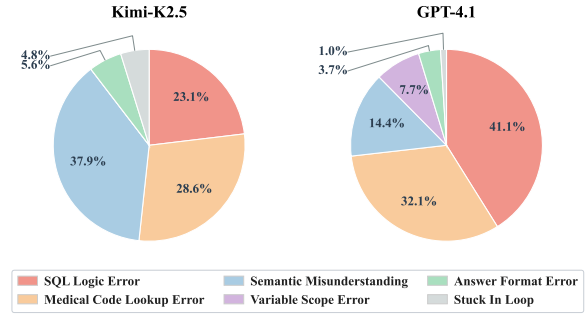


Figure 6: Distribution of Failure Categories.

397B, the mean turn count increases from 7.8 on patient-level tasks to 10.3 on population-level tasks. DeepSeek-V3.2-Exp uses the fewest turns on patient-level tasks, but this advantage narrows on population-level aggregation. For each maximum-turn limit $T \in \{1, 3, 5, 10, 15, 20\}$, we define $SR(T)$ as the fraction of the 712 tasks completed within T turns. The $T = 1$ condition provides a controlled approximation to one-shot generation, while larger budgets allow agents to use execution feedback in subsequent steps. Figure 5(b) shows that small turn limits are insufficient, while SR increases sharply by $T = 10$ and largely plateaus by $T = 15$. Full results are reported in Appendix I.

Failure Case Analysis To understand failure patterns on EHR-Complex, we analyze 3,814 failed trajectories from two representative models: Kimi-K2.5 and GPT-4.1. We first manually inspect more than 50 failures per model to derive a compact taxonomy, and then use Gemini 2.5 Pro to label the remaining failures at scale under the same rubric. The taxonomy contains six categories: *Semantic Misunderstanding*, *Medical-Code Lookup Error*, *SQL Logic Error*, *Variable Scope Error*, *Answer Format Error* and *Stuck-in-Loop*. Figure 6 shows that Kimi-K2.5 is dominated by *Semantic Misunderstanding* (37.9%) and *Medical-Code Lookup Error* (28.6%), indicating that many failures originate in clinical interpretation or grounding. GPT-4.1, by contrast, is dominated by *SQL Logic Error* (41.1%) and *Medical-Code Lookup Error* (32.1%), suggesting that it often captures the broad task intent but loses correctness during query construction. Further details are provided in Appendix G.

4.4 Trajectory Supervision Training

We evaluate whether successful EHR-Complex trajectories offer effective supervision for interactive reasoning. Distilling 6,000 Kimi-K2.5-generated

Table 4: Cross-Template and Cross-Intent Generalization Results.

Model	Scope	Demographics	Vitals	Medications	Cost	Labs	Diagnoses	All Avg.	Held-out Avg.
<i>(a) Cross-template Evaluation</i>									
Qwen3-8B	Patient	17.6	5.9	11.8	35.3	0.0	17.6	14.7	–
	Population	0.0	5.9	0.0	11.8	17.6	0.0	5.9	–
Qwen3-8B-SFT	Patient	64.7	17.6	41.2	82.4	64.7	35.3	51.0	–
	Population	23.5	11.8	17.6	35.3	52.9	0.0	23.5	–
<i>(b) Cross-intent Evaluation</i>									
Qwen3-8B	Patient	29.4	29.4	5.9	41.2	17.6	0.0	20.6	17.6
	Population	5.9	5.9	0.0	11.8	11.8	0.0	5.9	5.9
Qwen3-8B-SFT	Patient	82.4	11.8	52.9	94.1	52.9	58.8	58.8	39.2
	Population	29.4	17.6	35.3	41.2	29.4	11.8	27.5	27.4

Green: Models are fine-tuned on EHR-Complex trajectories. In panel (b), Vitals, Medications, and Labs are excluded from training. Held-out Avg. is computed over these three intents.

trajectories from the training set (with no test overlap), we fine-tune Qwen3-14B and Qwen3-32B via full-parameter SFT for one epoch. As shown in Table 3, this yields consistent in-domain gains: Qwen3-14B improves from 0.30 to 0.45, and Qwen3-32B from 0.36 to 0.55. Notably, the fine-tuned 32B model surpasses GPT-4.1 and becomes competitive with the much larger Qwen3-235B. These results confirm that EHR-Complex trajectories provide learnable procedural supervision for schema navigation and temporal grounding, validating the environment as a scalable source for agent training. Full details are in Appendix J.

Cross-Template and Cross-Intent Generalization. We further fine-tune Qwen3-8B for one epoch on 1,000 trajectories under two controlled settings. Cross-template evaluation uses 204 questions from entirely unseen templates, balanced across six clinical intents and two query scopes. Cross-intent evaluation excludes Vitals, Medications, and Labs from training and uses non-overlapping template splits. As shown in Table 4, SFT improves macro-average accuracy from 10.3 to 37.3 on unseen templates and from 11.8 to 33.3 on held-out intents. The gains indicate that trajectory supervision transfers beyond the query structures and clinical intents observed during training. Partition details are provided in Appendix J.3.

5 Conclusion

We introduced EHR-Complex, a large-scale benchmark for interactive clinical database reasoning over longitudinal EHRs. Built on the full MIMIC-IV substrate, EHR-Complex covers diverse patient-

and population-level queries that require sandboxed SQL/Python execution, multi-table aggregation, temporal reasoning, and compositional SQL logic. Experiments with broad LLM agents reveal persistent limitations in robust EHR reasoning, especially for population-level reasoning, medical-code grounding, and clinically faithful SQL construction. Trajectory supervision also transfers to unseen templates and clinical intents, although substantial gaps remain for dependable large-scale EHR analysis.

Limitations

EHR-Complex is built on MIMIC-IV v3.1, a large-scale publicly available EHR database released for research under credentialed access (Johnson et al., 2024). This inevitably bounds the benchmark to a single hospital system, including its EHR schema, ICU information system, coding conventions, patient population, and clinical practice patterns. Therefore, results may not directly transfer to other hospitals, EHR vendors, or international clinical settings. At the same time, MIMIC-IV provides a reproducible and privacy-preserving substrate for clinical-agent research: it has undergone professional de-identification, identifier replacement, PHI removal, and patient-level date shifting (Johnson et al., 2023). This enables us to release the full EHR-Complex train/test task definitions, SQL programs, evaluation environment, and construction code, while requiring users to access the underlying MIMIC-IV records through the standard credentialed-access procedure. EHR-Complex focuses on structured database reasoning and does not cover free-text notes, imaging data, real-time

clinician interaction, or prospective clinical deployment.

Ethical Considerations

EHR-Complex is constructed from MIMIC-IV v3.1, a large-scale de-identified EHR database released through PhysioNet for research under credentialed access (Johnson et al., 2024; Pollard et al., 2026; Goldberger et al., 2000). The underlying data have undergone privacy-preserving processing, including HIPAA identifier removal, random identifier replacement, structured-data filtering, free-text PHI removal, and date shifting (Johnson et al., 2023). We use MIMIC-IV only for research purposes and do not redistribute raw patient records, identifiers, or restricted database files. Any released EHR-Complex artifacts, such as task definitions, SQL programs, evaluation code, and construction scripts, will be shared only in forms compatible with the original MIMIC-IV access conditions. EHR-Complex is intended solely as a research benchmark for structured clinical database reasoning, not as a clinical decision-support system.

Acknowledgments

This work was supported in part by National Natural Science Foundation of China (62502435), the Zhejiang Provincial Natural Science Foundation (LQN26F020002), and Ant Group through CCF-Ant Research Fund.

References

- Anthropic. 2026. [Introducing Claude Sonnet 4.6](#).
- DeepSeek-AI. 2025a. [DeepSeek-V3.1 release](#).
- DeepSeek-AI. 2025b. [DeepSeek-V3.2-Exp: Boosting long-context efficiency with deepseek sparse attention](#).
- Alexandre Drouin, Maxime Gasse, Massimo Caccia, Issam H. Laradji, Manuel Del Verme, Tom Marty, David Vazquez, Nicolas Chapados, and Alexandre Lacoste. 2024. [WorkArena: How capable are web agents at solving common knowledge work tasks?](#) In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 11642–11662. PMLR.
- Ary L. Goldberger, Luis A. N. Amaral, Leon Glass, Jeffrey M. Hausdorff, Plamen Ch. Ivanov, Roger G. Mark, Joseph E. Mietus, George B. Moody, Chung-Kang Peng, and H. Eugene Stanley. 2000. [PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals](#). *Circulation*, 101(23):e215–e220.
- Google Gemini Team. 2026. [Gemini 3.1 Pro: A smarter model for your most complex tasks](#).
- Qian Huang, Jian Vora, Percy Liang, and Jure Leskovec. 2024. [MLAgentBench: Evaluating language agents on machine learning experimentation](#). In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 20271–20309. PMLR.
- Yixing Jiang, Kameron C. Black, Gloria Geng, Danny Park, James Zou, Andrew Y. Ng, and Jonathan H. Chen. 2025. [MedAgentBench: A virtual EHR environment to benchmark medical LLM agents](#). *NEJM AI*, 2(9):AIdbp2500144.
- Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. 2024. [SWE-bench: Can language models resolve real-world GitHub issues?](#) In *International Conference on Learning Representations*, volume 2024, pages 54107–54157.
- Alistair Johnson, Lucas Bulgarelli, Tom Pollard, Brian Gow, Benjamin Moody, Steven Horng, Leo Anthony Celi, and Roger Mark. 2024. [MIMIC-IV](#). *PhysioNet*. Version 3.1.
- Alistair E. W. Johnson, Lucas Bulgarelli, Lu Shen, Alvin Gayles, Ayad Shammout, Steven Horng, Tom J. Pollard, Sicheng Hao, Benjamin Moody, Brian Gow, Liwei H. Lehman, Leo A. Celi, and Roger G. Mark. 2023. [MIMIC-IV, a freely accessible electronic health record dataset](#). *Scientific Data*, 10(1):1.
- Koray Kavukcuoglu. 2025. [Gemini 2.5: Our most intelligent AI model](#).
- Nikhil Khandekar, Qiao Jin, Guangzhi Xiong, Soren Dunn, Serina S Applebaum, Zain Anwar, Maame Sarfo-Gyamfi, Conrad W Safranek, Abid A Anwar, Andrew Zhang, Aidan Gilson, Maxwell B Singer, Amisha Dave, Andrew Taylor, Aidong Zhang, Qingyu Chen, and Zhiyong Lu. 2024. [MedCalc-Bench: Evaluating large language models for medical calculations](#). In *Advances in Neural Information Processing Systems*, volume 37, pages 84730–84745. Curran Associates, Inc.
- Kimi Team. 2026. [Kimi K2.5: Visual agentic intelligence](#). *Preprint*, arXiv:2602.02276.
- Gyubok Lee, Elea Bach, Eric Yang, Tom Pollard, Alistair Johnson, Edward Choi, Yugang Jia, and Jong Ha Lee. 2026. [FHIR-AgentBench: Benchmarking LLM agents for realistic interoperable EHR question answering](#). In *Proceedings of the Fifth Machine Learning for Health Symposium*, volume 297 of *Proceedings of Machine Learning Research*, pages 1297–1315. PMLR.

- Gyubok Lee, Hyeonji Hwang, Seongsu Bae, Yeonsu Kwon, Woncheol Shin, Seongjun Yang, Minjoon Seo, Jong-Yeup Kim, and Edward Choi. 2022. [EHRSQL: A practical Text-to-SQL benchmark for electronic health records](#). In *Advances in Neural Information Processing Systems*, volume 35, pages 15589–15601. Curran Associates, Inc.
- MIT Laboratory for Computational Physiology. n.d. [MIMIC-IV documentation: diagnoses_icd](#). PhysioNet. Accessed August 25, 2026.
- OpenAI. 2024. [GPT-4o system card](#). *Preprint*, arXiv:2410.21276.
- OpenAI. 2025. [Introducing GPT-4.1 in the API](#).
- OpenAI. 2026. [Introducing GPT-5.4](#).
- Tom Pollard, Benjamin E. Moody, Li-wei H. Lehman, Brian J. Gow, Chrystinne Fernandes, Chen Xie, Alistair Johnson, Roger G. Mark, and Thomas Heldt. 2026. [PhysioNet as a global platform for biomedical research](#). *Nature Health*, 1(8):792–795.
- Qwen Team. 2025a. [Qwen3-235B-A22B-Instruct-2507](#). Hugging Face model card.
- Qwen Team. 2025b. [Qwen3 technical report](#). *Preprint*, arXiv:2505.09388.
- Qwen Team. 2026. [Qwen3.5: Towards native multi-modal agents](#).
- Preethi Raghavan, Jennifer J Liang, Diwakar Mahajan, Rachita Chandra, and Peter Szolovits. 2021. [emrK-BQA: A clinical knowledge-base question answering dataset](#). In *Proceedings of the 20th Workshop on Biomedical Language Processing*, pages 64–73. Association for Computational Linguistics.
- Jaehee Ryu, Seonhee Cho, Gyubok Lee, and Edward Choi. 2024. [EHR-SeqSQL: A sequential Text-to-SQL dataset for interactively exploring electronic health records](#). In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 16388–16407. Association for Computational Linguistics.
- Wenqi Shi, Ran Xu, Yuchen Zhuang, Yue Yu, Jieyu Zhang, Hang Wu, Yuanda Zhu, Joyce C. Ho, Carl Yang, and May Dongmei Wang. 2024. [EHRAgent: Code empowers large language models for few-shot complex tabular reasoning on electronic health records](#). In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 22315–22339. Association for Computational Linguistics.
- Ali Soroush, Benjamin S. Glicksberg, Eyal Zimlichman, Yiftach Barash, Robert Freeman, Alexander W. Charney, Girish N Nadkarni, and Eyal Klang. 2024. [Large language models are poor medical coders—benchmarking of medical code querying](#). *NEJM AI*, 1(5):AIdbp2300040.
- Ping Wang, Tian Shi, and Chandan K. Reddy. 2020. [Text-to-SQL generation for question answering on electronic medical records](#). In *Proceedings of The Web Conference 2020*, pages 350–361. ACM.
- Ran Xu, Yuchen Zhuang, Yishan Zhong, Yue Yu, Zifeng Wang, Xiangru Tang, Hang Wu, May Dongmei Wang, Peifeng Ruan, Donghan Yang, Tao Wang, Guanghua Xiao, Xin Liu, Carl Yang, Yang Xie, and Wenqi Shi. 2026. [MedAgentGym: A scalable agentic training environment for code-centric reasoning in biomedical data science](#). In *International Conference on Learning Representations*, volume 2026, pages 89688–89730.
- Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. 2025. [\$\tau\$ -bench: A benchmark for Tool-Agent-User interaction in real-world domains](#). In *International Conference on Learning Representations*, volume 2025, pages 9965–10017.
- Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. 2018. [Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and Text-to-SQL task](#). In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3911–3921. Association for Computational Linguistics.
- Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. 2024. [WebArena: A realistic web environment for building autonomous agents](#). In *International Conference on Learning Representations*, volume 2024, pages 15585–15606.

A Implementation Details

All evaluated systems use the same EHR-Complex agent interface. The agent receives the database schema description, the clinical question, and access to the executable tool, `execute_code`. Each tool call executes isolated SQL query or Python code. No variables are shared across tool calls. The agent must submit intermediate tool calls as valid JSON and return the final prediction inside a `<Final_answer>` tag. The SQL and verified answer are never exposed to the model during inference and are used only after submission for normalized final-answer evaluation. The released evaluator applies the same normalization and deterministic tie-breaking rules.

The implementation supports OpenAI, Azure, and local vLLM-style clients. All reported EHR-Complex runs use an OpenAI-compatible chat-completion API format. This format normalizes proprietary API calls and local or internally hosted open-weight endpoints under the same request schema. Malformed or unparsable actions are returned as environment feedback, and the environment enforces the 50-turn interaction budget used in the main experiments. Except for the Pass@k ablations, evaluation uses temperature 0.0. For locally served Qwen3 models, the client passes `enable_thinking` through both standard and SGLang-compatible request fields where supported. Table 5 lists the model identifiers and invocation settings used in our experiments. All open-weight models are served with SGLang.

B Template Materialization Examples

Table 6 summarizes one representative SQL template before parameter materialization. The template anchors an initial admission, retrieves later admissions, and aggregates the elapsed time after a subsequent laboratory event. Table 7 shows parameter-expanded instances from this template. Each row changes the clinical concepts and database constants while preserving the same executable reasoning pattern. Table 8 illustrates natural-language rewritings for one executable instance. All rewritings preserve the same temporal anchor, target measurement, and verified answer.

C SQL Structural Component Definitions

SQL task complexity is quantified with an AST-based extraction procedure implemented using `sqlglot`. For each query, we parse the SQL program and extract the structural components listed in Table 9, including nesting, aggregation, joins, conditions, grouping or ordering, and window functions. These components are summed within each query to obtain a per-query structural total, and the reported complexity is the average of these totals over tasks. We apply the same extraction procedure to all datasets in Table 1 and Figure 3.

Table 10 reports the fine-grained SQL structural component counts for EHR-Complex test tasks. TOTAL is computed before rounding.

Model	Run identifier	Backend	Max output
<i>Proprietary API models</i>			
GPT-4o	<code>gpt-4o-2024-11-20</code>	API	16,384
GPT-4.1 mini	<code>gpt-4.1-mini-2025-04-14</code>	API	32,768
GPT-4.1	<code>gpt-4.1-2025-04-14</code>	API	32,768
Gemini 2.5 Pro	<code>gemini-2.5-pro</code>	API	65,536
<i>Open-weight models $\geq 100B$</i>			
Qwen3-235B	<code>Qwen3-235B-A22B-Instruct-2507</code>	SGLang	65,536
Qwen3.5-397B	<code>Qwen3.5-397B-A17B</code>	SGLang	65,536
DeepSeek-V3.1	<code>DeepSeek-V3.1</code>	SGLang	65,536
DeepSeek-V3.2-Exp	<code>DeepSeek-V3.2-Exp</code>	SGLang	65,536
Kimi-K2.5	<code>Kimi-K2.5</code>	SGLang	65,536
<i>Open-weight models $< 100B$</i>			
Qwen3-4B	<code>Qwen3-4B</code>	SGLang	32,768 [†]
Qwen3-14B	<code>Qwen3-14B</code>	SGLang	32,768 [†]
Qwen3-32B	<code>Qwen3-32B</code>	SGLang	32,768 [†]

[†] Thinking is enabled for locally served Qwen3 models.

Table 5: Model invocation settings.

Template stage	Executable logic
Initial admission candidates	Join admissions, diagnoses_icd, and labevents; retain admissions matching the initial diagnosis placeholder [ICD_CODE_INITIAL] and initial laboratory placeholder [LAB_ITEM_ID_INITIAL].
First qualifying admission	Rank admissions by admittime within each patient and keep the earliest qualifying admission as the temporal anchor.
Subsequent admissions	Retrieve later admissions for the same patient by requiring admittime to be later than the temporal anchor.
Subsequent laboratory filter	Join later admissions with labevents and retain admissions containing the subsequent laboratory placeholder [LAB_ITEM_ID_SUBSEQUENT].
Population aggregation	Compute the day difference between the later admission and the initial admission, then return the average interval as the verified answer.

Table 6: Representative SQL template structure.

Parameter expansion	Generated clinical question	Verified answer
PHPT (25201); Total Chol (50907); Alk Phos (50863)	Among patients whose first qualifying admission carried diagnosis code PHPT and included lab Total Chol, what is the population-level mean days until a subsequent hospitalization where Alk Phos was drawn?	439.40 days
HLD (E785); Baso (51146); TC/HDL ratio (50903)	For the whole patient cohort, calculate the average time elapsed in days from an initial admission with diagnosis HLD and lab Baso to a subsequent hospitalization featuring lab TC/HDL ratio.	609.53 days
Liver disease (K7689); G6PD (52123); Retic count (51282)	What is the average days separating an initial admission defined by Liver disease and G6PD from a later hospitalization where Retic count was tested, across all patients?	84.00 days
h/o tobacco use (V1582); Total/HDL Ratio (50903); Sed Rate (51288)	Looking at the entire patient panel, what is the average duration in days separating an index h/o tobacco use admission with a Total/HDL Ratio baseline and a later readmission involving a Sed Rate test?	457.27 days

Table 7: Examples of parameter-expanded instances.

Rewrite	Question
1	What was the nadir hematocrit during the second admission for patient 10000032?
2	Can we identify the lowest Hct recorded during patient 10000032’s second hospitalization?
3	I need the lowest hematocrit value from the second admission for patient 10000032.
4	What was the trough Hct during the second hospitalization for patient 10000032?
5	Please provide the minimum hematocrit reading from patient 10000032’s second hospital encounter.
6	What is the trough hematocrit value for patient 10000032 during admission number two?

Table 8: Natural-language rewritings for one executable instance.

D Temporal Reasoning Taxonomy

We characterize the test set with a multi-label taxonomy derived from each question and its reference SQL: chronological selection of admissions or clinical events, comparison of admission times within a patient trajectory, and computation of elapsed intervals or lengths of stay. Among the 3,915 test tasks, 3,777 (96.5%) require chronological selection, 2,964 (75.7%) compare admission times, and 491 (12.5%) compute elapsed intervals or lengths of stay. These dimensions may co-occur. Because these operations use event-relative anchors within each patient, their order and elapsed intervals are

preserved under MIMIC-IV’s patient-specific date shifting (Johnson et al., 2023).

E Human Validation of Task Quality

Execution-based filtering verifies that a synthesized SQL program can be executed successfully and yields a deterministic answer. However, successful execution alone does not guarantee that the natural-language question, SQL logic, and final answer are semantically aligned. We therefore perform a two-stage task-quality validation procedure: a strong-model stress test over the full test set followed by an expert human audit on challenging cases.

Component	Operationalization	Interpretation
NEST	Number of SELECT clauses minus one.	Captures nested queries and common table expressions, reflecting multi-step retrieval or intermediate logic.
AGG	Number of aggregate function calls, including COUNT, SUM, AVG, MIN, and MAX.	Measures statistical summarization over raw records, such as counting patients or averaging measurements.
JOIN	Sum of explicit joins, implicit cross-table equality links, and subquery-based associations such as IN (SELECT ...).	Reflects how many database tables or event sources must be connected to answer the question.
COND	Number of filtering and logical clauses, including WHERE, HAVING, AND, and OR.	Captures the granularity of clinical constraints, such as time windows, diagnosis filters, item identifiers, or cohort restrictions.
GROUP/ORDER	Number of GROUP BY and ORDER BY clauses.	Reflects result-shaping operations such as ranking, top-k retrieval, or aggregation by patient, admission, unit, or category.
WINDOW	Number of OVER clauses, covering window functions such as ROW_NUMBER, RANK, LEAD, and LAG.	Captures temporal or within-group reasoning, such as selecting the first admission, latest measurement, or ordered event.
TOTAL	Sum of the six reported component counts for each query.	Used as the per-query structural complexity value before averaging over tasks.

Table 9: Definitions of SQL structural components used in the complexity analysis.

Scope	Clinical intent	Tasks	NEST	AGG	JOIN	COND	GROUP/ORDER	WINDOW	TOTAL
All	Overall	3,915	10.27	1.36	4.23	10.65	3.95	1.46	31.93
Pop.	Overall	2,155	9.87	1.42	4.12	9.75	4.07	1.42	30.65
Pop.	Labs	835	10.28	1.58	3.97	9.58	4.36	1.47	31.24
Pop.	Demographics	491	9.56	1.30	4.12	9.77	4.01	1.43	30.18
Pop.	Diagnoses	410	9.45	1.24	4.33	9.80	3.67	1.33	29.82
Pop.	Medications	324	10.01	1.37	4.23	10.26	3.95	1.52	31.33
Pop.	Cost	40	8.50	1.63	4.25	8.25	3.38	0.38	26.38
Pop.	Vitals	55	9.64	1.55	4.27	10.00	4.18	1.36	31.00
Pat.	Overall	1,760	10.76	1.29	4.36	11.76	3.82	1.51	33.49
Pat.	Labs	670	10.82	1.57	4.02	11.57	3.48	1.40	32.86
Pat.	Demographics	460	10.79	1.20	4.40	11.27	3.86	1.54	33.07
Pat.	Diagnoses	240	10.94	0.83	5.06	12.13	4.04	1.65	34.65
Pat.	Medications	260	10.98	1.15	4.60	12.79	4.15	1.63	35.31
Pat.	Cost	90	9.50	1.11	4.11	11.39	4.28	1.39	31.78
Pat.	Vitals	40	9.50	1.88	4.50	12.50	4.38	1.50	34.25

Table 10: Fine-grained SQL structural component counts for EHR-Complex test tasks.

Strong-model stress test. Before human review, we run four strong commercial model configurations from three model families: GPT-5.4 (OpenAI, 2026) with high and low reasoning effort, Gemini 3.1 Pro (Google Gemini Team, 2026), and Claude Sonnet 4.6 (Anthropic, 2026). For each test task, the model receives the same agent interface as evaluated models, including the natural-language question, schema information, and executable SQL/Python environment. The reference SQL program and verified answer are not exposed to the model. Each model independently interprets the question, interacts with the environment, and

submits a final answer for normalized final-answer evaluation.

Table 11 reports the strong-model results on the EHR-Complex test set. This stage is used to identify challenging cases rather than to certify task validity. Among the 3,915 test tasks, 954 are not solved by any of the four configurations. We use these 954 unsolved tasks as the candidate pool for human review, because they concentrate the cases most likely to expose either genuine task difficulty or potential semantic issues.

Sampling and annotation. From the 954 unsolved candidate tasks, we sample 500 task in-

stances for human validation. The sample is stratified across the six clinical intents and two query scopes, with allocation proportional to the number of retained instances in each intent–scope cell. To avoid auditing only repetitive cases, we also preserve variation in SQL structure and task templates during sampling. The audited tasks cover 390 structurally unique templates.

Each task is independently reviewed by two annotators with medical AI backgrounds, as well as experience in EHR data analysis and SQL-based database querying. Annotators are shown the natural-language question, reference SQL query, execution result, extracted final answer, relevant schema information, and supporting database evidence used during task construction. They focus on clinical plausibility and semantic alignment rather than SQL syntax alone. Specifically, they assess whether the natural-language question faithfully reflects the clinical intent encoded by the SQL and whether the extracted ground-truth answer is supported by the retrieved evidence. The human annotation criteria are summarized in Figure 7.

Annotators assign a binary validity label: 1 indicates that no substantive issue is found, and 0 indicates a concern requiring further inspection. They are blinded to strong-model predictions and downstream failure analyses. Disagreements are resolved through joint adjudication after inspecting the question, SQL, execution result, and supporting evidence.

Results. Table 12 summarizes the human validation results. After adjudication, 475 of the 500 audited unsolved tasks are judged valid, corresponding to a validity rate of 95.0%. The most common concerns involve borderline temporal wording, cohort-unit interpretation, medical-code granularity, or answer-format mismatches. Because the audited set is sampled from tasks unsolved by all four strong configurations, this result provides a conservative check of benchmark validity on difficult cases. The high validity rate suggests that most strong-model failures in this subset reflect task difficulty rather than invalid task construction. All tasks marked invalid after adjudication are corrected rather than removed, and their answers are regenerated and verified before the final test-set release. The release will include the corresponding audit annotations and correction notes.

Inter-annotator agreement. Before adjudication, the two annotators agree on 465 of the 500

binary labels, yielding 93.0% raw agreement. Because both annotators mark approximately 95% of tasks as valid, Cohen’s κ is reduced by the strong label-prevalence imbalance ($\kappa = 0.25$). The prevalence-adjusted bias-adjusted kappa (PABAK) is 0.86. Disagreements are primarily caused by borderline temporal wording, alternative interpretations of cohort units, or medical-code granularity, and are resolved through adjudication.

F Representative Population-Level Failure Cases

This appendix provides compact snapshots of representative population-level failures. These examples support the observation in Section 4.2 that cohort queries are harder than patient-level retrieval: the SQL often executes successfully, but a local assumption can silently change the cohort before aggregation.

Orange highlights mark the local assumption that changes the cohort definition. The three representative cases are shown in Tables 13–15.

G Error Analysis Details

G.1 Taxonomy Construction

We analyze failed trajectories from two representative models, Kimi-K2.5 and GPT-4.1, to characterize recurring failure modes on EHR-Complex. Annotators first manually inspect more than 50 failures per model, reviewing the clinical question, intermediate SQL/Python actions, execution feedback, and final answer to identify the primary local error that causes divergence. Based on this inspection, we define a compact taxonomy and use Gemini 2.5 Pro to annotate the remaining failures at scale under the same rubric, with ambiguous or low-confidence cases manually reviewed by annotators before aggregate statistics are reported.

G.2 Comparative Failure Patterns

The two models exhibit different failure profiles. Kimi-K2.5 is dominated by Semantic Misunderstanding, accounting for 37.9% of its failures. These errors often arise before SQL construction and then propagate through otherwise executable queries. GPT-4.1 is dominated by SQL Logic Error, accounting for 41.1% of its failures. These cases often reflect incorrect joins, aggregation keys, temporal filters, or patient-level deduplication rules after the broad task intent has been identified.

Table 11: Strong commercial model results on the EHR-Complex test set.

Model	Demographics		Vitals		Medications		Cost		Labs		Diagnoses		Avg.
	Pat.	Pop.	Pat.	Pop.	Pat.	Pop.	Pat.	Pop.	Pat.	Pop.	Pat.	Pop.	
GPT-5.4 (high)	0.85	0.36	0.78	0.34	0.8	0.37	0.93	0.76	0.91	0.49	0.84	0.36	0.65
GPT-5.4 (low)	0.81	0.32	0.65	0.26	0.73	0.31	0.88	0.66	0.86	0.43	0.8	0.29	0.58
Gemini 3.1 Pro	0.87	0.32	0.67	0.38	0.84	0.37	0.92	0.63	0.92	0.44	0.83	0.34	0.63
Claude Sonnet 4.6	0.59	0.21	0.45	0.12	0.45	0.12	0.83	0.2	0.49	0.33	0.4	0.11	0.36

Human annotation criteria

A task is marked as *valid* only if it satisfies all four checks below. Minor wording issues that do not affect the intended clinical meaning, SQL logic, or final answer are recorded separately but are not counted as invalid.

Check	Requirement
Question clarity	The natural-language question is understandable, clinically plausible, and answerable from the provided EHR database.
Semantic alignment	The SQL faithfully implements the intent expressed in the natural-language question, including joins, filters, temporal constraints, aggregation units, and cohort definitions.
Medical grounding	Clinical concepts, medical codes, laboratory items, medications, procedures, and diagnoses are grounded to appropriate database identifiers.
Answer support	The SQL execution result and supporting evidence justify a unique final answer under the benchmark’s answer-extraction rule.

Major issues include ambiguous clinical questions, incorrect medical-code grounding, wrong temporal constraints, incorrect aggregation units, mismatches between the SQL result and the final answer, or answers that are not medically supported by the retrieved evidence.

Figure 7: Human annotation criteria.

Table 12: Human validation summary for sampled EHR-Complex tasks.

Metric	Value	Count
Unsolved candidate tasks	24.4%	954 / 3,915
Audited sample	52.4%	500 / 954
Structurally unique templates	–	390
Adjudicated valid tasks	95.0%	475 / 500
Annotator 1 valid labels	95.0%	475 / 500
Annotator 2 valid labels	95.2%	476 / 500
Raw annotator agreement	93.0%	465 / 500
Cohen’s κ	0.25	–
PABAK	0.86	–

Medical-Code Lookup Error is the second-largest category for both models, accounting for 28.6% of Kimi-K2.5 failures and 32.1% of GPT-4.1 failures. This indicates that mapping clinical language to database-native identifiers remains a common bottleneck, consistent with prior evaluations of LLM-based medical-code querying (Soroush et al., 2024). One source of difficulty is the mismatch between surface forms and database-native code formats. For example, MIMIC-IV stores ICD

codes without dot notation, such as 25000 rather than 250.00 (MIT Laboratory for Computational Physiology, n.d.).

Turn counts further distinguish the failure styles. Kimi-K2.5 has long Stuck-in-Loop failures, averaging 44.7 turns, whereas GPT-4.1’s Variable Scope Error cases are the most turn-intensive within its profile, averaging 17.1 turns. These patterns suggest that execution feedback is only partially informative: the database may return clean outputs even when a local concept, code, join key, or temporal condition has already drifted from the intended clinical semantics.

G.3 Representative Failure Trajectories

The following examples illustrate the six error categories. Each case shows the task snapshot, the key erroneous excerpt, and a brief analysis of how the local mistake propagates to the final answer.

G.3.1 Trajectory 1: SQL Logic Error

Corresponding case: Table 17.

Case 1: Per-patient temporal alignment	
Question	What is the total number of individual patients whose first hospitalization had a diagnosis of Colonic diverticulosis (no bleed) and included lab test eGFR (MDRD), and who then, on their next (second) inpatient encounter, had any microbiology testing performed?
Agent assumption	The agent first finds the earliest <i>qualifying</i> admission for each patient and then uses <code>admission_order = f.admission_order + 1</code> to define the next admission.
Agent output	Final answer: 771
Ground truth	250
Analysis	The task requires the patient’s <i>first hospitalization overall</i> to satisfy the diagnosis and lab conditions. The agent instead anchors on the first qualifying admission anywhere in the timeline, which shifts the next-admission alignment and inflates the cohort count.

Table 13: Representative population-level failure case: per-patient temporal alignment.

Case 2: Silent aggregate failure	
Question	What are the top 5 most common insurance types among patients who had an index admission for ICD 43400 and a documented lab 51301 during a subsequent hospital admission?
Agent assumption	The agent treats the subsequent admission as a 30-day bounce-back and adds <code>DATE_DIFF('day', ia.index_disctime, a.admittime) <= 30</code> .
Agent output	Medicaid 2; Private 1. Final answer: Medicaid, Private
Ground truth	Medicare 6, Medicaid 6, Private 2
Analysis	The benchmark asks for a subsequent admission, not a 30-day readmission. The added time window silently shrinks the cohort, while the query still returns a plausible insurance summary.

Table 14: Representative population-level failure case: silent aggregate failure.

Case 3: Multi-stage query composition	
Question	How many unique patients with an index admission diagnosis code J45909 went on to have at least one later hospital admission that included both a FSH laboratory test and a prescription for Acetaminophen?
Agent assumption	The agent broadens the target asthma diagnosis code to <code>493*/J45*</code> codes and matches prescriptions with <code>p.drug ILIKE '%acetaminophen%'</code> .
Agent output	<code>unique_patient_count = 33</code> . Final answer: 33
Ground truth	15
Analysis	The overall query structure is close, but two local relaxations change the cohort: the diagnosis condition is broader than the target asthma diagnosis code J45909, and the drug condition accepts substring matches rather than the exact drug string. Together, these choices increase the final count from 15 to 33.

Table 15: Representative population-level failure case: multi-stage query composition.

G.3.2 Trajectory 2: Medical-Code Lookup Error

Corresponding case: Table 18.

G.3.3 Trajectory 3: Semantic Misunderstanding

Corresponding case: Table 19.

G.3.4 Trajectory 4: Variable Scope Error

Corresponding case: Table 20.

G.3.5 Trajectory 5: Stuck-in-Loop

Corresponding case: Table 21.

G.3.6 Trajectory 6: Answer Format Error

Corresponding case: Table 22.

H Pass^k and Pass@k Ablation Results

This appendix presents detailed Pass^k and Pass@k results for the representative models used in the robustness analysis. Pass^k (temperature=0.0) mea-

Category	Definition
Semantic Misunderstanding	Misinterprets the clinical request, such as the wrong admission, cohort criterion, or clinical ontology.
Medical-Code Lookup Error	Fails to ground clinical terms to database-native identifiers, including ICD, laboratory, medication, or procedure codes.
SQL Logic Error	Implements incorrect query logic, such as wrong joins, filters, aggregation keys, deduplication units, or temporal constraints.
Variable Scope Error	Assumes variables or intermediate tables persist across isolated tool executions.
Answer Format Error	Retrieves near-correct information but violates the required final-answer format.
Stuck-in-Loop	Repeats unproductive variants of the same plan without meaningful revision.

Table 16: Failure taxonomy used for trajectory annotation.

Trajectory 1: SQL Logic Error — Task snapshot	
Question	Among patients who underwent laboratory test 50820 during a hospital admission, which 5 ICD diagnosis codes were recorded for the largest number of distinct patients in the same admission?
Ground truth	[E785, 4019, I10, Z87891, N179]
Agent answer	[D62, E785, J9601, 4019, E872]
Local error	Counts joined lab–diagnosis rows rather than distinct patients.

Trajectory 1: SQL Logic Error — Key excerpt
<pre> SELECT d.icd_code, d.icd_version, COUNT(*) AS frequency FROM labevents l JOIN diagnoses_icd d ON l.subject_id = d.subject_id AND l.hadm_id = d.hadm_id WHERE l.itemid = 50820 GROUP BY d.icd_code, d.icd_version ORDER BY frequency DESC LIMIT 5 </pre> COUNT(*) counts all joined lab–diagnosis combinations. The intended unit is distinct patients, e.g., COUNT(DISTINCT d.subject_id). Agent output: D62, E785, J9601, 4019, E872

Table 17: **Analysis.** The query executes and returns plausible ICD codes, but the counting unit is wrong. Because each admission can contain multiple lab events and diagnoses, counting joined rows instead of distinct patients inflates frequencies and changes the top-5 ranking.

sure the proportion of tasks where *all* k runs are correct, whereas Pass@ k (temperature=0.6) measures the proportion of tasks where *at least one* of k runs is correct. Both metrics are estimated using the unbiased combinatorial estimator over $n=4$ independent runs.

H.1 Overall Results

Table 23 reports the overall robustness results across all evaluated models. The results show the expected contrast: Pass ^{k} decreases as k increases,

while Pass@ k increases with additional samples.

H.2 Results by Query Scope

Table 24 separates the robustness results by query scope. Population-level tasks show substantially lower Pass ^{k} consistency than patient-level tasks, matching the main-paper finding that cohort reasoning is more fragile.

Trajectory 2: Medical-Code Lookup Error — Task snapshot	
Question	Which five entry points contributed the most admissions for patients with ICD code Ileus who underwent laboratory test Vanc during their stay?
Ground truth	EMERGENCY ROOM; TRANSFER FROM HOSPITAL; PHYSICIAN REFERRAL; PACU; CLINIC REFERRAL
Agent answer	TRANSFER FROM HOSPITAL; PHYSICIAN REFERRAL; EMERGENCY ROOM; WALK-IN/SELF REFERRAL; TRANSFER FROM SKILLED NURSING FACILITY
Local error	After finding relevant Ileus codes, the agent hardcodes only a narrow ICD-10 subset and ignores dominant ICD-9 codes.

Trajectory 2: Medical-Code Lookup Error — Key excerpt	
<pre> ICD codes for Ileus: icd_code icd_version long_title 27700 9 Cystic fibrosis w/o meconium ileus 27701 9 Cystic fibrosis with meconium ileus 5372 9 Chronic duodenal ileus 5601 9 Paralytic ileus </pre>	
The later SQL keeps only ('K560', 'K563', 'K567') with icd_version = 10, discarding ICD-9 codes such as 5601. <pre> WITH ileus_patients AS (SELECT DISTINCT subject_id, hadm_id FROM diagnoses_icd WHERE icd_code IN ('K560', 'K563', 'K567') AND icd_version = 10) </pre>	

Table 18: **Analysis.** The agent performs an initially useful code lookup but fails to use the returned identifiers. The resulting cohort is much smaller and has a different admission-location distribution.

H.3 Results by Clinical Intent

Tables 25–27 report the intent-level k-sweeps. Across intents, Cost is closest to saturation under sampling, whereas Diagnoses, Vitals, and Medications remain more brittle under repeated rollouts.

I Interaction Budget Results

This appendix reports full interaction-budget curves for the evaluated models. $SR(B)$ denotes the fraction of tasks solved within budget B , where $B=1, 3, 5, 10, 15$, and 20 . The final two columns report the full success rate under the original 50-turn cap and the mean number of interaction turns over all tasks.

Table 28 shows that the representative models used in the main interaction analysis are not outliers. For most systems, success rises once several rounds of execution feedback are available and then saturates well before the 50-turn cap.

J SFT Training Details

J.1 Trajectory Data

We extract 6,000 successful task completions generated by Kimi-K2.5 from the EHR-Complex training partition. All selected trajectories are disjoint from the test set and are correct under the normalized final-answer evaluator. Each trajectory contains the original clinical question, multi-turn SQL/Python execution attempts, database feedback, iterative query refinements, and the final verified answer. The average trajectory contains 20.9 messages, preserving the interactive reasoning process rather than reducing supervision to static input–output pairs.

J.2 Training Configuration

Table 29 summarizes the supervised fine-tuning setup. We use the same EHR-Complex test set and normalized final-answer evaluation protocol as in the main experiments.

Trajectory 3: Semantic Misunderstanding — Task snapshot

Question	What are the top 5 most common last care units during the second hospital admission for patients with an initial diagnosis of ICD code 486, given they had laboratory test 51221 during their first admission and laboratory test 50983 during their second?
Ground truth	Medicine; Hematology/Oncology; Medicine/Cardiology; Med/Surg; Neurology
Agent answer	Medical Intensive Care Unit (MICU); Medical/Surgical Intensive Care Unit (MICU/SICU); Surgical Intensive Care Unit (SICU); Coronary Care Unit (CCU); Trauma SICU (TSICU)
Local error	Interprets “care unit” as ICU-specific units and queries icustays instead of transfers.

Trajectory 3: Semantic Misunderstanding — Key excerpt

```
SELECT
  i.last_careunit,
  COUNT(*) AS count
FROM icustays i
JOIN admissions_ordered ao ON i.hadm_id = ao.hadm_id
WHERE ao.admission_num = 2
  AND i.hadm_id IN (
    SELECT hadm_id FROM labevents WHERE itemid = 50983
  )
GROUP BY i.last_careunit
ORDER BY count DESC
LIMIT 5
```

icustays contains ICU-specific units. The question asks for last care units during admission, which should be derived from transfers.

Agent output:
MICU, MICU/SICU, SICU, CCU, TSICU

Table 19: **Analysis.** The SQL is internally coherent but operates on the wrong clinical entity. As a result, the answer is restricted to ICU units and misses the general ward units in the ground truth.

J.3 Cross-Template and Cross-Intent Generalization

The original EHR-Complex split is instance-disjoint: the training and test sets share structural template families but use non-overlapping parameter instantiations, and the training questions are additionally rewritten. To measure transfer beyond this split, we conduct two controlled Qwen3-8B experiments. For cross-template evaluation, we fine-tune for one epoch on 1,000 trajectories sampled from a 300-template partition and evaluate 204 questions from entirely unseen templates, balanced across six clinical intents and two query scopes. For cross-intent evaluation, we sample 1,000 trajectories approximately evenly from Demographics, Cost, and Diagnoses, and evaluate 204 questions across all six intents on non-overlapping template splits. Vitals, Medications, and Labs are held out from training.

Table 4 reports the full intent- and scope-level results. On unseen templates, SFT improves macro-average accuracy from 10.3 to 37.3. In cross-intent

evaluation, the macro-average over all six intents improves from 13.2 to 43.1, while the average over the three held-out intents improves from 11.8 to 33.3. These results show transfer beyond both the query structures and clinical intents observed during training.

J.4 Training Dynamics

Figure 8 shows the training loss curves for the two fine-tuned models. Both models fit the trajectory data within the one-epoch SFT setting.

The Qwen3-14B loss decreases from 0.311 to 0.089 over 110 steps, while the Qwen3-32B loss decreases from 0.240 to 0.093 over 55 steps. These curves indicate that the models optimize quickly on the successful interaction trajectories.

J.5 Additional SFT Result Analysis

Both fine-tuned models improve across most clinical intents and query scopes. Qwen3-14B-SFT reaches 0.45 macro-averaged accuracy, an absolute gain of 0.15 over the base model. Qwen3-32B-SFT reaches 0.55, an absolute gain of 0.19 over the base

Trajectory 4: Variable Scope Error — Task snapshot

Question	Across the cohort, what's the mean number of days from a patient's first admission coded Depression to their first subsequent readmission where Tbili was obtained?
Ground truth	726.78 days
Agent answer	341.08 days
Local error	Assumes variables persist across isolated code executions, repeatedly triggering NameError and later reconstructing a truncated cohort.

Trajectory 4: Variable Scope Error — Key excerpt

Turn 1:
depression_codes = conn.execute("""
SELECT icd_code, icd_version
FROM d_icd_diagnoses
WHERE LOWER(long_title) LIKE '%depression%'
""").df()

Turn 3:
NameError: name 'depression_codes' is not defined

Turn 5:
NameError: name 'depression_admissions' is not defined

Turn 7:
NameError: name 'pd' is not defined

Turn 9:
NameError: name 'readmissions' is not defined

Turn 11:
NameError: name 'tbili_lab_events' is not defined

Repeated state-loss errors force redundant reconstruction. The final consolidated query produces 341.08 days instead of 726.78 days.

Table 20: **Analysis.** The environment is stateless across tool calls, but the agent repeatedly assumes previous variables remain available. After several recovery attempts, the final query executes but uses an incomplete cohort.

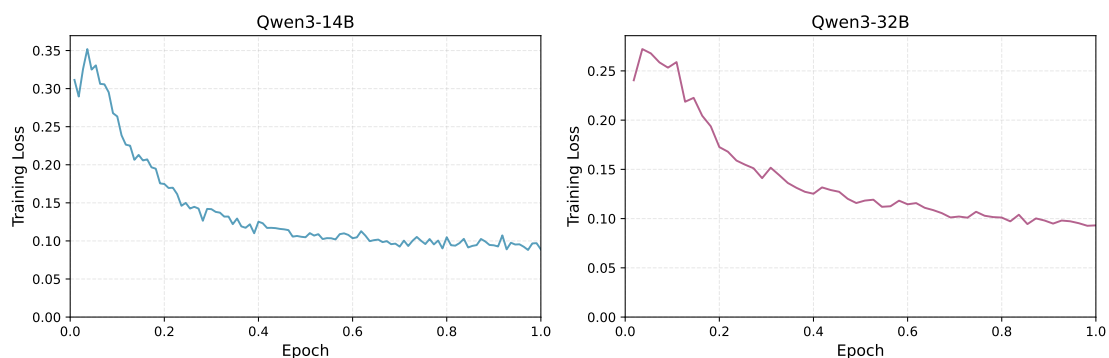


Figure 8: Training loss curves during supervised fine-tuning.

model. The largest gains appear on patient-level tasks requiring structured temporal or schema reasoning, including Labs Pat. for Qwen3-14B (0.40 $\rightarrow$ 0.74), Cost Pat. for Qwen3-14B (0.63 $\rightarrow$ 0.84), Demographics Pat. for Qwen3-32B (0.51 $\rightarrow$ 0.83), and Labs Pat. for Qwen3-32B (0.50 $\rightarrow$ 0.80).

These results should be interpreted as an in-domain training-signal validation. The experiment uses a single teacher model and evaluates on the same benchmark distribution, so it does not establish transfer to other EHR databases or clinical environments. Nevertheless, the gains suggest that suc-

Trajectory 5: Stuck-in-Loop — Task snapshot

Question	During the hospitalization immediately following patient 10000980's initial stay with ICD-9 diagnosis code 496, how many Chloride tests were ordered?
Ground truth	7
Agent answer	None
Local error	Interprets ICD-9 diagnosis code 496 as a literal <code>stay_id</code> , then repeatedly searches for nonexistent stay identifiers.

Trajectory 5: Stuck-in-Loop — Key excerpt

```
-- Initial mistaken hypothesis
SELECT * FROM icustays
WHERE subject_id = 10000980 AND stay_id = 496;

-- Empty result, but the agent keeps the same interpretation
SELECT * FROM icustays
WHERE CAST(stay_id AS VARCHAR) LIKE '%496%';

-- Later repeated through the remaining turns
SELECT * FROM icustays
WHERE subject_id = 10000980 AND stay_id = 30496496;
```

The agent never revises the assumption that 496 is a stay identifier, even though the question specifies it as an ICD-9 diagnosis code.

Table 21: **Analysis.** The failure begins with a semantic misinterpretation of the diagnosis code as an ICU stay identifier, but the dominant behavior is lack of hypothesis revision. Even after repeated empty results, the agent continues searching for guessed `stay_id` values until the turn budget is exhausted.

cessful EHR-Complex trajectories provide useful supervision for schema navigation, temporal reasoning, medical grounding, and execution-guided query refinement.

Trajectory 6: Answer Format Error — Task snapshot

Question	Looking across all admissions with 414.01 documented and a 50899 lab obtained, which five medications are most commonly ordered, based on the count of separate admissions in which each drug was prescribed?
Ground truth	Acetaminophen; Sodium Chloride 0.9% Flush; Carbamazepine; Heparin; Docusate Sodium
Required format	A structured list with both drug and prescription_count, e.g., {'drug': 'Acetaminophen', 'prescription_count': 31}.
Agent answer	Acetaminophen; Sodium Chloride 0.9% Flush; Carbamazepine; Heparin; Atorvastatin
Local error	Returns medication names without the required counts and does not follow the benchmark's deterministic answer-extraction outcome for the tied fifth medication.

Trajectory 6: Answer Format Error — Key excerpt

```
SELECT p.drug, COUNT(DISTINCT p.hadm_id) AS admission_count
FROM prescriptions p
INNER JOIN common_admissions ca ON p.hadm_id = ca.hadm_id
GROUP BY p.drug
ORDER BY admission_count DESC
LIMIT 5
```

Retrieved result:

Acetaminophen	31
Sodium Chloride 0.9% Flush	29
Carbamazepine	25
Heparin	24
Atorvastatin	23

Final answer:

Acetaminophen, Sodium Chloride 0.9% Flush,
Carbamazepine, Heparin, Atorvastatin

The final answer omits the required count fields and does not match the benchmark's deterministic answer-extraction outcome for the tied fifth medication.

Table 22: **Analysis.** The retrieval is close to correct, but the final response violates the required structured format by omitting prescription_count. It also returns Atorvastatin rather than the ground-truth tied medication Docusate Sodium under the benchmark's deterministic answer-extraction rule. The error occurs at answer rendering rather than during SQL execution.

Model	Pass ^k (temp=0.0)				Pass@k (temp=0.6)			
	k=1	k=2	k=3	k=4	k=1	k=2	k=3	k=4
Qwen3.5-397B	62.31	57.18	54.42	52.69	62.60	67.24	69.18	70.20
Kimi-K2.5	60.74	54.96	51.87	49.81	59.90	66.50	69.35	71.14
DeepSeek-V3.2-Exp	58.23	50.89	46.35	42.95	58.83	66.43	69.45	71.17
Qwen3-235B	50.57	40.57	35.19	31.72	50.72	60.43	64.86	67.88
GPT-4.1-mini	47.12	37.55	32.49	29.33	45.85	55.26	59.55	62.31
GPT-4.1	44.72	35.18	30.06	26.76	45.16	55.49	60.24	63.07
Qwen3-32B	34.33	23.53	18.36	15.35	33.63	46.15	53.31	58.31
Gemini-2.5-Pro	31.73	20.46	16.01	13.84	31.00	45.56	54.40	60.05
GPT-4o	28.71	18.80	14.44	12.18	28.59	38.79	44.30	47.86

Table 23: Overall Pass^k and Pass@k results for k=1, 2, 3, and 4.

(a) Patient-level tasks									(b) Population-level tasks								
Model	Pass^k				Pass@k				Model	Pass^k				Pass@k			
	k=1	k=2	k=3	k=4	k=1	k=2	k=3	k=4		k=1	k=2	k=3	k=4	k=1	k=2	k=3	k=4
Qwen3.5-397B	88.11	84.00	81.34	79.49	88.93	92.29	93.29	93.71	Qwen3.5-397B	41.20	35.24	32.40	30.77	41.26	46.95	49.65	51.16
Kimi-K2.5	87.75	83.57	81.27	79.77	86.85	92.19	94.18	95.40	Kimi-K2.5	39.10	32.04	28.31	25.80	38.33	45.94	49.48	51.72
DeepSeek-V3.2-Exp	83.62	76.58	71.48	67.53	82.50	89.95	92.07	93.14	DeepSeek-V3.2-Exp	38.10	30.52	26.42	23.46	39.75	47.47	51.21	53.46
Qwen3-235B	76.21	66.14	59.21	54.00	77.91	87.64	90.84	92.61	Qwen3-235B	29.94	20.00	15.86	13.79	29.07	38.76	44.17	48.19
GPT-4.1-mini	69.87	58.83	52.56	48.43	69.11	80.30	84.73	87.50	GPT-4.1-mini	28.98	20.57	16.48	14.09	27.42	35.40	39.58	42.34
GPT-4.1	70.45	60.27	54.19	49.72	69.18	81.34	86.22	88.64	GPT-4.1	24.32	15.28	10.92	8.56	26.13	34.98	39.64	42.79
Qwen3-32B	50.58	36.15	28.48	23.84	48.26	65.46	75.00	81.70	Qwen3-32B	21.90	13.88	10.63	8.86	21.82	30.58	35.81	39.44
Gemini-2.5-Pro	43.11	27.70	21.73	19.03	42.68	62.31	73.79	80.40	Gemini-2.5-Pro	22.69	14.71	11.46	9.71	21.73	32.28	39.02	43.92
GPT-4o	46.95	33.00	26.42	23.01	47.30	62.12	69.89	74.72	GPT-4o	13.99	7.34	4.76	3.44	13.69	20.21	23.93	26.47

Table 24: Pass^k and Pass@k results by query scope.

(a) Cost & Billing									(b) Demographics & Tracking								
Model	Pass^k				Pass@k				Model	Pass^k				Pass@k			
	k=1	k=2	k=3	k=4	k=1	k=2	k=3	k=4		k=1	k=2	k=3	k=4	k=1	k=2	k=3	k=4
Qwen3.5-397B	84.62	82.05	80.77	80.77	85.58	87.82	88.46	88.46	Qwen3.5-397B	59.55	54.97	52.62	50.79	60.16	64.67	66.93	68.23
Kimi-K2.5	86.54	82.05	78.85	76.92	80.77	84.62	86.54	88.46	Kimi-K2.5	63.02	57.56	54.64	52.58	60.88	67.27	69.69	70.98
DeepSeek-V3.2-Exp	82.69	77.56	73.08	69.23	86.54	88.46	88.46	88.46	DeepSeek-V3.2-Exp	57.60	50.60	46.65	43.81	59.28	67.18	70.62	72.68
Qwen3-235B	83.65	79.49	75.96	73.08	82.69	88.46	88.46	88.46	Qwen3-235B	48.59	38.80	33.59	30.26	48.86	58.12	62.56	65.48
GPT-4.1-mini	84.62	81.41	78.85	76.92	81.73	88.46	91.35	92.31	GPT-4.1-mini	50.26	40.85	35.64	32.31	48.10	56.68	60.53	62.94
GPT-4.1	75.00	69.23	65.38	61.54	76.92	86.54	90.38	92.31	GPT-4.1	48.48	39.26	33.88	30.46	47.59	57.95	62.69	65.48
Qwen3-32B	64.42	53.21	47.12	42.31	73.96	83.33	89.58	95.83	Qwen3-32B	32.21	22.16	17.35	14.12	30.80	42.82	49.59	54.14
Gemini-2.5-Pro	50.00	36.54	29.81	26.92	50.00	69.23	79.81	84.62	Gemini-2.5-Pro	32.49	21.57	17.64	15.23	31.09	43.99	51.52	56.35
GPT-4o	58.65	47.44	40.38	34.62	63.00	78.67	82.00	84.00	GPT-4o	28.97	18.55	13.59	10.26	28.30	38.24	44.42	48.73

Table 25: Intent-level Pass^k and Pass@k results for Cost and Demographics tasks.

(a) Diagnoses & Procedures									(b) Labs & Microbiology								
Model	Pass^k				Pass@k				Model	Pass^k				Pass@k			
	k=1	k=2	k=3	k=4	k=1	k=2	k=3	k=4		k=1	k=2	k=3	k=4	k=1	k=2	k=3	k=4
Qwen3.5-397B	50.76	43.64	40.08	38.17	52.46	57.32	59.47	60.61	Qwen3.5-397B	67.37	63.31	61.41	60.40	67.31	71.35	73.16	74.25
Kimi-K2.5	51.88	45.11	41.35	39.10	50.95	58.96	62.31	64.39	Kimi-K2.5	64.07	59.27	56.62	54.64	64.35	70.47	73.07	74.50
DeepSeek-V3.2-Exp	48.87	39.60	34.02	30.08	52.71	61.63	64.92	66.67	DeepSeek-V3.2-Exp	63.29	56.70	52.33	48.84	63.25	70.03	72.43	73.84
Qwen3-235B	37.97	27.19	22.37	19.55	36.57	46.14	51.31	55.22	Qwen3-235B	56.50	46.85	41.05	37.16	56.62	66.50	70.61	73.51
GPT-4.1-mini	34.02	24.06	18.98	15.79	34.70	44.53	49.44	52.24	GPT-4.1-mini	49.92	39.90	34.69	31.46	48.18	58.03	62.05	64.69
GPT-4.1	34.89	27.11	23.32	20.90	34.89	42.91	46.46	48.51	GPT-4.1	48.84	38.45	32.59	28.71	48.51	58.86	63.53	66.34
Qwen3-32B	24.79	16.24	12.18	10.26	25.43	36.06	42.46	46.55	Qwen3-32B	36.36	24.12	18.18	14.77	35.04	48.82	56.88	62.83
Gemini-2.5-Pro	26.31	15.42	10.45	7.46	25.56	40.67	50.56	57.46	Gemini-2.5-Pro	33.99	23.10	18.65	16.50	32.34	47.36	56.44	62.05
GPT-4o	21.35	14.10	10.77	9.23	18.61	25.94	30.45	33.08	GPT-4o	31.60	21.40	16.91	14.85	32.34	43.78	49.42	52.81

Table 26: Intent-level Pass^k and Pass@k results for Diagnoses and Labs tasks.

(a) Medications & Orders									(b) Vital Signs & Observations								
Model	Pass^k				Pass@k				Model	Pass^k				Pass@k			
	k=1	k=2	k=3	k=4	k=1	k=2	k=3	k=4		k=1	k=2	k=3	k=4	k=1	k=2	k=3	k=4
Qwen3.5-397B	63.48	56.67	51.96	48.70	62.83	69.57	71.52	72.17	Qwen3.5-397B	52.63	45.61	40.79	36.84	50.00	54.63	55.56	55.56
Kimi-K2.5	54.57	47.10	43.70	41.74	54.96	62.93	67.24	70.69	Kimi-K2.5	48.68	39.47	34.21	31.58	43.06	44.44	44.44	44.44
DeepSeek-V3.2-Exp	53.73	44.88	39.69	35.96	49.12	58.19	62.50	64.91	DeepSeek-V3.2-Exp	43.42	40.35	38.16	36.84	46.05	53.51	56.58	57.89
Qwen3-235B	47.41	36.06	30.39	26.72	49.35	59.77	64.87	68.10	Qwen3-235B	40.79	28.95	23.68	21.05	40.79	54.39	60.53	63.16
GPT-4.1-mini	43.97	33.91	28.45	25.00	41.67	51.00	55.98	59.83	GPT-4.1-mini	30.26	22.81	21.05	21.05	40.79	52.63	59.21	63.16
GPT-4.1	33.12	21.94	17.09	14.53	38.89	52.14	58.55	62.39	GPT-4.1	39.47	32.46	28.95	26.32	34.21	42.98	48.68	52.63
Qwen3-32B	34.86	24.36	19.71	17.31	34.05	46.98	53.33	57.14	Qwen3-32B	40.62	28.12	21.88	18.75	38.33	51.11	60.00	66.67
Gemini-2.5-Pro	29.09	16.81	12.28	11.21	31.41	46.01	54.49	59.83	Gemini-2.5-Pro	17.11	2.63	0.00	0.00	18.42	32.46	43.42	52.63
GPT-4o	21.96	11.88	8.26	6.96	23.72	34.05	40.17	44.44	GPT-4o	30.26	14.91	10.53	10.53	26.32	31.58	34.21	36.84

Table 27: Intent-level Pass^k and Pass@k results for Medications and Vitals tasks.

Model	B=1	B=3	B=5	B=10	B=15	B=20	Full SR	Avg. turns
Qwen3.5-397B	0.00	0.00	0.05	0.39	0.54	0.58	0.60	11.82
Kimi-K2.5	0.00	0.00	0.05	0.40	0.54	0.57	0.59	11.10
DeepSeek-V3.2-Exp	0.00	0.15	0.25	0.43	0.51	0.54	0.57	9.73
DeepSeek-V3.1	0.00	0.02	0.20	0.49	0.52	0.53	0.53	7.41
Qwen3-235B	0.00	0.02	0.13	0.41	0.48	0.50	0.50	9.14
GPT-4.1-mini	0.00	0.10	0.17	0.34	0.42	0.44	0.45	8.85
GPT-4.1	0.00	0.03	0.10	0.34	0.42	0.43	0.44	9.29
Qwen3-32B	0.00	0.14	0.24	0.32	0.33	0.33	0.33	5.35
Gemini 2.5 Pro	0.06	0.20	0.26	0.29	0.30	0.30	0.30	2.35
Qwen3-14B	0.00	0.12	0.19	0.25	0.27	0.27	0.27	6.46
GPT-4o	0.00	0.00	0.05	0.24	0.27	0.27	0.27	8.24

Table 28: Interaction-budget results.

Component	Setting
Training data	6,000 successful Kimi-K2.5 trajectories from the training set. All trajectories are test-disjoint and correct under the normalized final-answer evaluator.
Trajectory format	Original clinical question, multi-turn SQL/Python attempts, execution feedback, iterative query refinements, and final verified answer. Average length: 20.9 messages.
Base models	Qwen3-14B and Qwen3-32B (Qwen Team, 2025b).
Optimization	Full-parameter supervised fine-tuning for one epoch.
Distributed setup	8 nodes with 8×H800 GPUs per node, for 64 GPUs total; DeepSpeed ZeRO-3.
Precision and context length	bfloat16 precision; maximum sequence length 32,768 tokens.
Learning rate	1×10^{-5} with 5% warmup.
Effective batch size	128.
Selected checkpoints	checkpoint-110 for Qwen3-14B; checkpoint-55 for Qwen3-32B.
Evaluation	Same EHR-Complex test set and normalized final-answer metric as Table 3.

Table 29: Supervised fine-tuning configuration.