An OMOP-Based Open-Source Text-to-SQL Benchmark Dataset

Paul Legrand[1], **Kawsar Noor**[1], **Satyam Bhagwanani**[1], **Richard J Dobson**[1,2]
[1] University College London, London, United Kingdom
[2] King's College London, London, United Kingdom
paul.legrand@etu.univ-amu.fr, kawsar.noor.15@ucl.ac.uk,
s.bhagwanani@ucl.ac.uk, richard.j.dobson@kcl.ac.uk

Abstract

Access to electronic health record (EHR) warehouses is limited by SQL expertise and complex clinical schemas. We present an open-source OMOP Common Data Model text-to-SQL benchmark (CDM v5.4) with a safety contract: output one executable SQL statement or the abstention token (<NO_SQL>) for unanswerable requests. Inputs are concept-normalized (entities as OMOP concept IDs) to decouple SQL generation from entity linking. We evaluate by executing predicted and reference queries on a synthetic OMOP PostgreSQL database, reporting Execution Accuracy (result equivalence) and a reliability score that rewards correct abstention and penalizes unsafe attempts. The dataset includes 6,690 paraphrases from 75 OMOP-adapted templates with leakage-resistant template/SQL-variation splits. LoRA-tuned Llama-3-8B-Instruct achieves 93.55% execution accuracy with improved abstention reliability, while schema-injected baselines fail the contract. We release the dataset, splits, database dump, and a reproducible evaluation pipeline to support reliable clinical analytics assistants.

Keywords: Text-to-SQL, Electronic Health Records (EHR), OMOP CDM, Large Language Models (LLMs)

1. Introduction

EHRs hold rich structured clinical data, yet extracting cohorts or summary statistics still requires SQL and schema expertise. Text-to-SQL could enable natural-language querying, but clinical deployment is difficult: performance is highly schema-dependent, and seemingly plausible SQL can silently yield incorrect cohorts. Evaluation should therefore prioritize executable, schema-faithful SQL and quantify risk-sensitive reliability, including abstention (Geifman and El-Yaniv, 2019) when a request is not answerable from the database. General benchmarks such as Spider (Yu et al., 2018) have driven progress, but their schemas and evaluation settings differ from real hospital warehouses, limiting transfer; execution-based scoring is essential because syntactically valid SQL can still be clinically wrong due to faulty joins or predicates. Earlier systems often relied on explicit schema linking and relation-aware schema encoders (e.g., RAT-SQL (Wang et al., 2020)).

Recent clinical datasets add templated questions and unanswerables to test abstention, yet benchmark-style resources explicitly grounded in OMOP-CDM remain scarce despite the model's widespread adoption. OMOP-oriented efforts are often narrower in scope (e.g., epidemiological question answering) and are seldom framed as engine-specific, fully executable benchmarks (e.g., PostgreSQL) with an explicit abstention/refusal contract. For instance, Ziletti and DAmbrosi (2024) explore OMOP-CDM text-to-SQL for epidemiological QA with a RAG-based pipeline; however, our setting differs in that we explicitly include non-answerable requests and specify a dedicated abstention token to support a refusal-aware executable benchmark.

The OMOP Common Data Model (CDM) is a standardized relational schema promoted by the Observational Health Data Sciences and Informatics (OHDSI) community (Hripcsak et al., 2015), validated for scalable observational safety surveillance across heterogeneous databases (Overhage et al., 2012), and evaluated as a practical CDM for longitudinal registries (Garza et al., 2016).

We cast clinician-style free-text requests as executable queries over OMOP CDM v5.4 under a strict interface: output either SQL only (one statement ending with exactly one semicolon) or the abstention token <NO_SQL>. Because OMOP concept names are unreliable keys, robust cohort definitions use concept_id-based predicates (optionally expanded via vocabulary relationships); we therefore encode medical constraints as concept identifiers to evaluate OMOP-grounded SQL generation rather than brittle string matching. Accordingly, this benchmark evaluates the downstream SQL generation component given concept-normalized inputs; integrating upstream clinical entity linking from raw text is left to future end-to-end work.

We release[1] an open-source benchmark and reproducible pipeline with:

- a text-to-SQL dataset grounded in OMOP CDM v5.4 with an explicit safe-abstention label (<NO_SQL>) for unanswerable requests;
- leakage-resistant train/validation/test splits that keep paraphrases grouped and control

[1] https://github.com/red1dwarf/TEXT-TO-OMOP

Proceedings of the 8th Workshop on Clinical Natural Language Processing (Clinical NLP) @ LREC 2026, pages 381–393
16 May 2026.

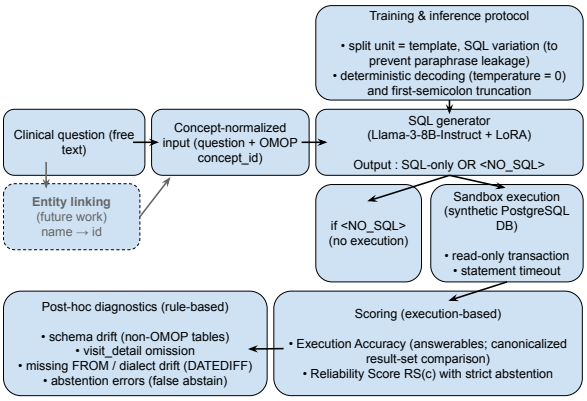

Figure 1: Overview of the benchmark and evaluation protocol.

for template/SQL variation overlap;

- a standardized evaluation suite combining text-based metrics, answerability detection, and execution-based scoring in read-only PostgreSQL, plus post-hoc diagnostic analyses;

- end-to-end reproducibility artifacts, including training/inference scripts (LoRA fine-tuning and schema-injected baselines) and an optional synthetic OMOP PostgreSQL dump.

2. Method

We describe the benchmark task, dataset construction, split protocol, training/evaluation setup, and metrics for OMOP CDM v5.4 text-to-SQL.

2.1. Task definition

We study schema-grounded SQL generation for EHR analytics under a strict executable contract. Given an input question x, the model must output either a single executable, OMOP-grounded SQL statement $\hat{q}$ (SQL only, no prose), ending with exactly one semicolon, or the exact abstention token `<NO_SQL>` when the question is unanswerable from the database (defined in Section 2.4). At inference, we enforce the single-statement constraint by truncating generation at the first semicolon. Any output that is not exactly `<NO_SQL>` is treated as a SQL attempt and executed for scoring; invalid or empty SQL thus results in an execution failure. All models are trained and evaluated with one instruction: use only OMOP CDM tables/columns and valid relationships; never output `SELECT *`; output only SQL ending with exactly one semicolon; otherwise abstain with `<NO_SQL>`. OMOP-grounding violations (e.g., schema drift) are detected via post-hoc diagnostics and reflected in execution-based metrics.

2.2. Target schema

The target environment is the OMOP Common Data Model (CDM) v5.4 using the PostgreSQL dialect (Observational Health Data Sciences and Informatics (OHDSI), 2021; PostgreSQL Global Development Group, 2025). Valid queries may reference only OMOP CDM tables/columns and must follow OMOP relationships and conventions (e.g., concept-driven filtering through standardized vocabularies and event-table semantics). Queries that reference non-OMOP schema elements are treated as schema drift and are penalized in diagnostic and execution-based evaluation.

2.3. Dataset construction

The benchmark includes 75 question templates aligned with OMOP join patterns and conventions. Starting from the EHRSQL template set (a template-based clinical text-to-SQL resource over EHR data; (Lee et al., 2023)), we built a local pool and retained a subset to limit structural redundancy (i.e., templates sharing the same query skeleton with only slot/value substitutions or minor surface differences). We then adapted these templates to OMOP CDM v5.4 join paths and conventions and added a small number of OMOP-specific templates to better cover under-represented domains and joins (e.g., `note`/`note_nlp` free-text tables, `provider` and `death`, and `episode`/`episode_event` patterns).

To capture free-text-linked querying, we included note-centric templates using `note` and `note_nlp`. In a local OMOP-mapped MIMIC-IV instance (Johnson et al., 2023; OHDSI, 2025), the `note` table was populated from the MIMIC-IV NOTE module and the `note_nlp` table from MedCAT-derived annotations over radiology reports and discharge summaries (Kraljevic et al., 2021). We used an OMOP-mapped MIMIC-IV environment to reflect common clinical warehouse settings where structured OMOP domains and unstructured text coexist; it also enabled us to check schema connectivity and template plausibility during development. This OMOP-mapped MIMIC-IV environment was used only during dataset/template development; execution-based evaluation is performed exclusively on the released synthetic OMOP PostgreSQL database (Section 2.8).

Each template is typically instantiated into three (template, SQL) variations: the SQL structure is fixed while inserted values (e.g., concept IDs, patient IDs, thresholds, time windows) vary, yielding 223 (template, SQL) variations.

Gold queries often use common table expressions (CTEs; `WITH`) to match clinical analytics practice and to make cohort definitions, temporal constraints, and visit-hierarchy logic explicit, encourag-

ing structured OMOP decompositions over monolithic nested subqueries.

For each variation, we produce 30 surface forms: one canonical question plus 29 GPT-4.1 paraphrases (OpenAI, 2025a). Paraphrases are constrained to (i) preserve meaning and (ii) keep placeholder values within typed tags unchanged, resulting in 6,690 answerable examples. Unanswerable questions are not included in this total; we take them from EHRSQL-2024 solely for abstention evaluation. The released *.jsonl files include these unanswerables alongside the answerable splits after applying the typed-tag and formatting normalization described in Section 2.4.

Typed identifier tags. Identifiers are enclosed in lightweight typed tags (e.g., <DRUG>...</DRUG>, <PERSON_ID>...</PERSON_ID>). These markers have no XML semantics; they enable concept-normalized inputs by representing clinical mentions as `concept_id` values (rather than free-text mentions or `concept_name` strings) while making the identifier type explicit. For instance, "Did patient 289 take Panadol yesterday?" becomes "Did patient <PERSON_ID>289</PERSON_ID> take <DRUG>839681</DRUG> yesterday?". Typing both concept identifiers (drugs, conditions, etc.) and non-concept identifiers (patient IDs) reduces ambiguity, discourages brittle associations between raw integers and OMOP fields, and encourages schema-driven reasoning (mapping typed roles to the correct *_concept_id predicates and join paths rather than relying on lexical cues).

Human verification protocol. Authors reviewed each paraphrase to ensure semantic equivalence, preservation of all slot values and temporal constraints, and absence of clinically implausible additions; failures were edited or removed.

2.4. Normalization of unanswerables

We use unanswerable questions from the EHRSQL-2024 shared task (Lee et al., 2024b). As in EHRSQL (Lee et al., 2023), a question is labeled unanswerable when the structured OMOP instance alone is insufficient: it either (i) requires external/normative clinical knowledge not stored in the database, or (ii) asks for information outside the database/schema.

In the initial setup, there was a shortcut: unanswerables often lacked typed tags, making tag presence predictive of answerability. In the released normalized splits (*.jsonl), we mitigate this by (i) inserting typed tags when entity-like identifiers occur, (ii) sampling synthetic identifiers to mirror answerable distributions (e.g., person-like ID lengths), and (iii) standardizing punctuation, thereby weakening the "tags $\Rightarrow$ answerable" cue.

2.5. Concept-normalized inputs and anti-memorization design

OMOP queries typically require standardized vocabulary identifiers (e.g., *_concept_id). Because mapping surface mentions (e.g., "troponin") to OMOP IDs is an upstream entity-linking step, each example includes two aligned inputs: `question_name` (human-readable concept names) and `question_concept` (key entities replaced by concept IDs). Importantly, there is no explicit name-to-ID linkage between the two views: the surface concept names in `question_name` do not correspond to the concept IDs appearing in `question_concept`. We release both for readability and future end-to-end studies, but report results only for SQL generation from the concept-normalized input (Figure 1), so metrics reflect generation with gold-like OMOP identifiers rather than raw-text entity linking. To limit spurious "ID binding", identifiers are synthetic (concept-/person-like integers) and not tied to real trajectories, though a minority may correspond to public OMOP concept_ids from the OHDSI Standardized vocabularies (Reich et al., 2024).

2.6. Unanswerables & abstention

Unanswerables are included to stress-test reliability. They include both normative/external-knowledge requests and "near-miss" out-of-schema queries (e.g., cross-ontology code lookups such as ICD-11$\rightarrow$ICD-10), mirroring realistic user intents that a database-backed assistant should refuse safely rather than answer with a plausible proxy SQL. For these examples, the only correct output is the single abstention token <NO_SQL> (no explanatory text), making abstention machine-actionable for downstream routing/clarification.

2.7. Split protocol

Instance-level splits can leak in paraphrase-heavy datasets, so we split at the (template, SQL) variation level: paraphrases of the same underlying query never cross splits. Per template, up to two variations are designated *seen*; each has 30 paraphrases split 20/5/5 into train/validation/test. All remaining variations are *unseen*; we sample 37 unseen variations for validation and 37 for test, adding 5 paraphrases per unseen variation to evaluate controlled generalization. Unanswerables are injected at fixed rates (10% train; 20% validation/test), yielding 3,312 train examples (2,980 answerable + 332 unanswerable) and 1,162 examples for both validation and test (930 + 232). The split is deterministic with a fixed seed (default 42), and we release the JSONL split files for exact reproducibility.

Category	Statistic (answerables)
Templates / variations	75 templates; 223 (template, SQL) variations
Size (answerable)	6,690 examples
JOINs (gold SQL)	median 1; p90 3; max 4
CTE usage (WITH)	40.36%
GROUP BY / DISTINCT	28.25% / 20.63%
ORDER BY / LIMIT	60.54% / 60.54%
Tagging (questions)	median 1 tag; 12.57% no-tag
Most-used OMOP tables	concept (45.74%), visit_occurrence (30.04%), condition_occurrence (26.91%), drug_exposure (21.52%), measurement (17.49%), visit_detail (16.14%), person (13.45%), procedure_occurrence (10.76%), device_exposure (10.76%)

Table 1: Dataset summary statistics for answerable examples.

2.8. Synthetic OMOP database for execution-based evaluation

Execution-based evaluation is the most faithful measure of text-to-SQL correctness (Wang et al., 2018), but with synthetic identifiers even correct SQL can return empty outputs on a real OMOP instance. We therefore release a controlled synthetic OMOP PostgreSQL database for evaluation only (never training). It is generated *post hoc* from the fixed gold SQL set (model-agnostic) and is not iteratively altered using model outputs: for each gold query, we insert the minimal synthetic rows needed in the relevant OMOP tables so that the gold query returns a non-empty result, while preserving basic heterogeneity (multiple patients, dates, events). Population was performed through GPT-5.1-assisted PostgreSQL insert scripts (OpenAI, 2025b), followed by author review, verification, and execution; OMOP integrity constraints were not enforced (duplicates may exist). All gold queries are verified to execute successfully, and we release the full database dump and profiling artifacts used for all reported execution metrics. Gold SQL queries were authored/validated against a real OMOP-mapped MIMIC-IV environment during dataset development (Section 2.3) and finalized before creating this synthetic instance, ensuring no circularity between query design and the evaluation database.

Profiling shows the instance is non-trivial: 23 OMOP tables with 117,010 rows (∼10 MB), including 1,858 persons and 6,453 visits. Event density is strongly long-tailed, with 125.48 events/person on average, a median of 9 events/person, a p99 of 992.92 events/person, and a maximum of 2,374 events/person. The visit hierarchy is substantial (63,826 `visit_detail` rows, ∼9.9× `visit_occurrence`), making join paths and aggregation/temporal logic challenging rather than "toy-like". The database is designed for discriminative scoring rather than epidemiological realism; for time-relative templates (e.g., "today"), we use a single fixed reference date (see Section 6).

2.9. Models and training setup

We fine-tune Llama-3-8B-Instruct (Grattafiori et al., 2024) with LoRA (Hu et al., 2021). Inputs follow the Llama-3 chat template with a fixed system prompt enforcing SQL-only outputs and `<NO_SQL>` abstention; loss is computed only on generated tokens (prompt masked). LoRA: $r = 32$, $\alpha = 64$, dropout 0.05, applied to attention and MLP projection modules. Training: 3 epochs, batch 8, grad accumulation 4, LR 1.5×10^{-4} with cosine schedule, warmup ratio 0.05, optionally using bf16 and/or gradient checkpointing.

2.10. Inference protocol

All models use deterministic inference (seed 42; temperature 0.0; top_p 1.0; max_new_tokens 1024), producing exactly one output per example (SQL or `<NO_SQL>`). Outputs are saved as JSONL and scored with the same execution-based pipeline on the released synthetic OMOP PostgreSQL instance. We score raw generations truncated at the first semicolon, with no model-specific post-processing.

Schema-injected baseline prompting. For baselines, we prepend an OMOP PostgreSQL DDL schema block (OMOP tables/columns only) to the user message for every example, aligning with Natural-SQL-style prompting. We run Natural-SQL-7B (ChatDB, 2024) (a public 7B ChatDB text-to-SQL instruction model), plus Llama-3-8B-Instruct (base) (Grattafiori et al., 2024) and Mistral-7B-Instruct-v0.3 (Jiang et al., 2023), under the same schema-injection setup. Although prompts follow each model's chat/template conventions, we keep an identical system instruction and the same SQL-only / `<NO_SQL>` output contract across all models (baselines and LoRA); the

injected schema block is the only prompt content that differs between baselines and LoRAs.

2.11. Evaluation metrics

We prioritize semantic correctness and reliability, and treat Execution Accuracy (ExecAcc) and the risk-sensitive Reliability Score (RS(c)) as the primary benchmark metrics for EHR text-to-SQL. All other metrics are reported as secondary analyses and diagnostics.

Primary text-to-SQL metrics.

Execution Accuracy (ExecAcc). Predicted and gold SQL are executed on PostgreSQL over the synthetic OMOP database (read-only transactions; per-query statement timeout). ExecAcc is computed only on answerable examples: a prediction is correct iff it is not <NO_SQL>, executes successfully, and returns the same normalized (order-invariant) result set as the gold query. Unanswerable examples are excluded from ExecAcc and evaluated via RS(c) and answerability-oriented metrics.

Formal definition. Let A be the set of answerable examples (gold $\neq$ <NO_SQL>) whose gold SQL executes without error on the evaluation database. Let $\hat{q}_i$ be the model output and $\mathrm{Res}(q)$ the normalized (order-invariant) result set returned by PostgreSQL. We define $\mathrm{Res}(q) = \perp$ if execution fails (e.g., syntax error, timeout), so the equality $\mathrm{Res}(\hat{q}_i) = \mathrm{Res}(q_i)$ can only hold for successful executions. ExecAcc is computed as:

$$\mathrm{ExecAcc} = \frac{1}{|A|} \sum_{i \in A} \mathbf{1}[\hat{q}_i \neq \texttt{<NO\_SQL>} \wedge \mathrm{Res}(\hat{q}_i) = \mathrm{Res}(q_i)] \quad (1)$$

We compare gold and predicted results after normalization: (i) map NULL to the empty string, (ii) keep values in native Python types (no string casting), (iii) canonicalize each row as the sorted tuple of its values, sorted deterministically by grouping values by Python type, then by a stable textual sorting key, so the comparison is invariant to output-column permutations, and (iv) sort the resulting list of rows to be order-invariant.
Column names and column order are deliberately ignored. Mapping NULL to the empty string improves robustness to driver-specific NULL handling; in our evaluation set, gold queries for answerable questions never return NULL or empty strings, so this choice does not change the reported execution metrics.

Reliability Score (RS(c)). Following the risk-sensitive reliability evaluation proposed in Trust-SQL (Lee et al., 2024a), we assign a sample-level score based on five cases, with penalty parameter $c \in \{0, 1, 10\}$:

1. answerable + correct SQL $\rightarrow +1$
2. answerable + abstain (<NO_SQL>) $\rightarrow 0$
3. answerable + incorrect/failed SQL $\rightarrow -c$
4. unanswerable + abstain (<NO_SQL>) $\rightarrow +1$
5. unanswerable + SQL attempt $\rightarrow -c$

Dataset-level RS(c) is the average sample score; higher c penalizes unsafe attempts more strongly than abstention, matching the clinical preference that wrong is worse than refuse. Reporting $c \in \{0, 1, 10\}$ shows how conclusions change from a neutral setting ($c = 0$), where unsafe SQL attempts are not penalized more than abstention, to increasingly risk-averse settings.

Secondary metrics (supporting analysis).

We also report: (i) Answerability F1 ($F1_{ans}$) for the binary choice SQL vs. <NO_SQL> (answer vs. abstain), with "answerable" as the positive class (SQL = positive, <NO_SQL> = negative); (ii) Unanswerable Recall (UnansRec), the fraction of unanswerables predicted as <NO_SQL>; (iii) False abstain (%), the fraction of answerables predicted as <NO_SQL>; and (iv) $F1_{exe}$, which summarizes executable correctness over attempted SQL outputs (with R_{exe} = ExecAcc by definition), computed as the harmonic mean of P_{exe} and R_{exe}. $F1_{exe}$ and $F1_{ans}$ are inspired by EHRSQL (Lee et al., 2023).

Formalization of $F1_{exe}$. Let A be answerable examples whose gold SQL runs without error on the evaluation database; let $T \subseteq A$ be those where the model outputs SQL ($\hat{q}_i \neq$ <NO_SQL>), and $C \subseteq T$ those executed correctly ($\mathrm{Res}(\hat{q}_i) = \mathrm{Res}(q_i)$). Unlike EHRSQL, we compute $\mathrm{Precision}_{exe}$ over T (conditioned on gold-answerables), since attempts on unanswerables are handled by our abstention/reliability metrics. Thus $\mathrm{Precision}_{exe} = |C|/|T|$, $\mathrm{Recall}_{exe} = |C|/|A|$ (= ExecAcc), and $F1_{exe} = 2PR/(P+R)$ with $P = \mathrm{Precision}_{exe}$ and $R = \mathrm{Recall}_{exe}$.

Auxiliary string-overlap and rule-based diagnostics. We report Exact Match (EM), Exact Match Normalized (EMN), and ROUGE-L as string-overlap diagnostics, but treat them as secondary because semantically equivalent SQL can differ syntactically (e.g., join order, predicate reordering) (Finegan-Dollak et al., 2018). EM is exact string match (Rajpurkar et al., 2016); EMN applies a lightweight normalization (lowercasing, trailing semicolon removal, whitespace collapsing);

ROUGE-L is based on the longest common subsequence (LCS) overlap (Lin, 2004). We also report rule-based diagnostics (e.g., schema drift, table-set mismatch, dialect drift, visit hierarchy omissions) to characterize common failure modes.

3. Results

We report results on the held-out test split under a unified pipeline: identical instruction (SQL-only; single statement; abstain with <NO_SQL>), deterministic decoding, truncation at the first semicolon, and scoring using execution-based and reliability-oriented metrics on the synthetic database.

3.1. Main results

Table 2 shows a clear gap between OMOP-adapted and general-purpose models. For transparency about training dynamics, we also report LoRA-150, which is not a separate model family but an earlier checkpoint from the same LoRA training run. The best adapter, LoRA-312, reaches 93.55% ExecAcc and $F1_{exe}$=93.65%, while preserving near-perfect answerability ($F1_{ans}$=99.46%) and positive risk-sensitive reliability (RS(10)=0.373). LoRA-150 and LoRA-312 should be read as an early-vs.-final checkpoint comparison within one training run, rather than as two independently trained methods.

Sensitivity analysis. With a stricter equivalence (matching column headers and preserving within-row value order, still row-order invariant), LoRA-312 performance remains essentially unchanged (93.44% ExecAcc), suggesting ExecAcc is not an artifact of result-set canonicalization (i.e., our default comparison ignores column names/order and canonicalizes rows; Section 2.11). Moreover, the released synthetic database enforces unique identifiers (distinct entities never share an ID) and gold answerable queries never return NULL/empty strings, so ambiguous identical-value multisets across columns are not expected in practice.

Off-the-shelf baselines remain far lower in executable correctness despite schema-injected prompting: Natural-SQL-7B (18.39%), Llama-3-8B-Instruct (base) (10.54%), and Mistral-7B-Instruct-v0.3 (4.09%). Their RS(10) is strongly negative (-8.381 to -9.517), showing that under our strict contract they frequently attempt SQL instead of abstaining, which is heavily penalized by reliability-oriented scoring. All baselines have EM/EMN = 0.00, largely due to incorrect SQL, while also suggesting that the rare correct outputs seldom match the gold SQL form, reinforcing the need for execution-based evaluation.

Overall, strict OMOP querying requires schema- and domain-aligned supervision beyond generic instruction or text-to-SQL tuning.

3.2. Answerability: abstention behavior

With 20% unanswerables in the test set (232/1,162), abstention must be evaluated explicitly. Under class imbalance (930 answerable vs 232 unanswerable), $F1_{ans}$ (SQL vs <NO_SQL>) can be inflated by models that almost always attempt SQL: baselines still reach $F1_{ans}$ ≈89% yet have RS(10)≈ -9, indicating unsafe over-attempting rather than calibrated abstention. We therefore prioritize Unanswerable Recall and RS(c) for safety-critical behavior. Table 3 shows that LoRA-312 abstains correctly on 96.55% (224/232) of unanswerables while rarely over-refusing answerables (0.22%, 2/930); LoRA-150 is slightly lower on unanswerables (92.67%) with the same false-abstain rate (0.22%). Baselines almost never abstain (UnansRec 0–5.60%), yielding large high-penalty reliability losses.

3.3. Diagnostics: failures & challenges

To complement aggregate scores, Table 3 reports interpretable failure proxies. False Abstain and Unanswerable Recall follow Section 3.2; Schema drift / Table-set mismatch / DATEDIFF usage / CTE usage are computed on gold-answerable items where the model attempts SQL (excluding <NO_SQL>), while Unanswerable Recall is computed on unanswerables. Schema drift flags any FROM/JOIN relation token that is neither an OMOP table, a CTE name, nor a whitelisted set-returning function. Table-set mismatch compares the set of base tables in gold vs. prediction (CTEs excluded). DATEDIFF usage detects PostgreSQL-incompatible DATEDIFF. CTE usage checks whether the prediction starts with WITH. visit_detail miss is computed on examples whose gold SQL uses visit_detail, counting a miss whenever the prediction omits visit_detail (or abstains).

Baselines show large table-set mismatch (52-79%) and frequent visit_detail omission (70-89%) when visit hierarchy reasoning is required, plus schema drift (non-OMOP tables in FROM/JOIN; up to 14.62% for Llama-3-8B-Instruct (base)) and dialect drift (e.g., DATEDIFF, ≈5-6% for some baselines). LoRA largely eliminates schema/dialect drift (≈0%) and reduces table-set mismatch to 0.65-1.19%, leaving visit hierarchy grounding as the key residual error: omitting visit_detail (1.33% for LoRA-312) can still produce executable SQL with clinically different semantics (e.g., ICU segment time vs. overall hospitalization).

3.4. Qualitative error cases (LoRA-312)

We highlight three representative residual failure modes for the best adapter (Table 4) and link each example to the corresponding diagnostic in Table 3.

Model	ExecAcc (%)	F1_exe (%)	F1_ans (%)	EM (%)	EMN (%)	ROUGE-L (%)	RS(0)	RS(1)	RS(10)
LoRA-312	**93.55**	**93.65**	**99.46**	**90.54**	**90.86**	**98.77**	**0.941**	**0.885**	**0.373**
LoRA-150	89.68	89.77	98.99	81.40	82.26	97.65	0.903	0.807	-0.052
Natural-SQL-7B	18.39	18.39	88.91	0.00	0.00	40.23	0.147	-0.706	-8.381
Llama-3-8B-Instruct (base)	10.54	10.54	89.21	0.00	0.00	36.74	0.090	-0.819	-9.006
Mistral-7B-Instruct-v0.3	4.09	4.09	89.47	0.00	0.00	31.08	0.043	-0.912	-9.517

Table 2: Model performance on the OMOP text-to-SQL benchmark (test set). LoRA-150 and LoRA-312 are two checkpoints from the same LoRA training run (early vs. final checkpoint). RS(0), RS(1), and RS(10) denote the Reliability Score with penalty parameter $c \in \{0, 1, 10\}$, where larger c penalizes unsafe SQL attempts more strongly relative to abstention.

Model	False abstain (%)	Unanswerable Recall (%)	Schema drift (%)	Table-set mismatch (%)	DATEDIFF usage (%)	CTE usage (%)	visit_detail miss (%)
LoRA-312	**0.22**	**96.55**	**0.00**	**0.65**	**0.00**	**40.30**	1.33
LoRA-150	0.22	92.67	0.00	1.19	0.00	40.09	**0.67**
Natural-SQL-7B	0.00	0.00	0.43	52.47	0.00	15.05	78.67
Llama-3-8B-Instruct (base)	0.00	3.02	14.62	76.88	5.81	0.32	70.00
Mistral-7B-Instruct-v0.3	0.00	5.60	8.17	79.25	5.27	9.68	88.67

Table 3: Diagnostic analysis on the test set (930 answerable / 232 unanswerable).

4. Discussion

Results show that OMOP-specific adaptation is necessary: generic SQL skill does not transfer, and remaining errors are concentrated in join-path, visit-hierarchy, and temporal/aggregation semantics.

4.1. Why generic SQL fails in OMOP

We observe a large gap between general instruction-tuned baselines and the OMOP-adapted LoRA model: baselines often produce plausible "SQL-like" outputs that fail in OMOP, where queries must follow domain conventions (concept-driven filtering via standardized vocabularies, event-table semantics, and canonical join paths across person, visits, and clinical-domain tables). Diagnostic results make this explicit: baseline errors are dominated by table-set mismatch, visit-hierarchy mistakes, and schema/dialect drift, showing that generic SQL competence does not yield OMOP-correct queries. Conversely, the best LoRA adapter suggests these conventions can be learned with modest parameter-efficient tuning in a lightweight open-source model, enabling on-prem hospital deployment and CDM/dialect adaptation without relying on large or proprietary systems.

4.2. Reliability and abstention should be first-class evaluation targets

Clinical analytics is unlike many text-to-SQL settings: a wrong query can silently produce plausible but incorrect cohorts or statistics. Accordingly, the benchmark includes unanswerable questions and enforces a single abstention token, `<NO_SQL>`. Reliability-aware scoring (RS(c)) complements ExecAcc by increasingly penalizing unsafe attempts relative to abstention as c grows, reflecting a common clinical preference: wrong is worse than refuse. The LoRA model's higher RS(c), together with high unanswerable recall and low false-abstain rates, shows abstention is both learnable and quantifiable under a simple, machine-actionable policy. Practically, free-form explanations inside the SQL-generation component may be counterproductive: "helpful" natural language can be ambiguous for downstream systems and mistaken for executable intent. A strict contract (SQL-only or `<NO_SQL>`) better supports safe agentic integration, where `<NO_SQL>` can trigger clarification, fallback tools, or human review.

4.3. Remaining challenges: joins, visits, time/aggregation

After fine-tuning, Table 3 shows the remaining gap is semantic, concentrated in three OMOP-critical choices: (i) join-path selection across event tables when multiple schema-consistent routes exist (especially for multi-domain queries); (ii) visit-hierarchy selection (`visit_occurrence` vs. `visit_detail`), where choosing the wrong level or omitting required `visit_detail` implicitly changes the cohort; and (iii) temporal/aggregation semantics (window bounds,

Failure mode	Representative example (gold vs. LoRA-312)
Visit hierarchy grounding (`visit_detail` omission; aligns with *visit_detail miss* in Table 3)	**Question:** How long has patient <PERSON_ID>6102133621015</PERSON_ID> been in the ICU during the current hospital stay? **Gold (uses ICU segment time):** `SELECT ... FROM visit_occurrence vo JOIN visit_detail vd ON vd.visit_occurrence_id = vo.visit_occurrence_id JOIN concept c ON vd.visit_detail_concept_id = c.concept_id WHERE ... c.concept_name ILIKE '%intensive care%' ...;` **LoRA-312 (drops `visit_detail`):** `SELECT ... FROM visit_occurrence vo JOIN concept c ON vo.visit_concept_id = c.concept_id WHERE ... c.concept_name ILIKE '%intensive care%' ...;` **Impact:** executable SQL but clinically different semantics (hospital-level stay vs. ICU segment duration).
Table grounding confusion (`measurement` vs. visit tables; aligns with *Table-set mismatch* in Table 3)	**Question:** Can you find the third instance of patient <PERSON_ID>3720242571871</PERSON_ID>'s weight recorded on May 6, last year? **Gold (OMOP measurement retrieval + offset):** `SELECT m.value_as_number FROM measurement m WHERE m.person_id = 3720242571871 AND m.measurement_concept_id = 3025315 AND m.measurement_date = MAKE_DATE(EXTRACT(YEAR FROM CURRENT_DATE)::int - 1, 5, 6) ORDER BY m.measurement_datetime OFFSET 2 LIMIT 1;` **LoRA-312 (incorrect table choice):** `SELECT ... FROM visit_occurrence vo JOIN concept c ON ... WHERE vo.person_id = 3720242571871 AND vo.visit_start_date = MAKE_DATE(...) AND c.concept_name ILIKE '%weight%' ORDER BY ... OFFSET 2 LIMIT 1;` **Impact:** wrong OMOP event-table semantics; misses concept-driven filtering and measurement ordering logic.
Unsafe attempt on an unanswerable (aligns with *Unanswerable Recall* and RS(c) penalties in Table 3)	**Question:** How long should patient <PERSON_ID>4773708619089828657</PERSON_ID> have taken <DRUG>942384</DRUG>, the last month? **Gold:** <NO_SQL> **LoRA-312 (unsafe SQL attempt):** `SELECT MAX(de.drug_exposure_start_datetime) ..., MIN(...) ... FROM drug_exposure de WHERE de.person_id = ... AND de.drug_concept_id = 942384 AND EXTRACT(MONTH FROM de.drug_exposure_start_date) = EXTRACT(MONTH FROM CURRENT_DATE) - 1;` **Impact:** answers a normative/external-knowledge request by (incorrectly) fabricating a computable proxy from timestamps.

Table 4: Representative residual failure modes for LoRA-312.

grouping granularity, `DISTINCT`), where small mismatches can look plausible yet invalidate the statistic. This implies (a) the benchmark remains discriminative for clinically consequential reasoning, including schema-grounded relational structure, event- vs. visit-level intent, and faithful temporal/aggregation cues, and (b) progress likely requires structured grounding and semantic verification beyond scaling alone. Three directions are: (1) constraint-aware decoding over an OMOP join graph (e.g., PICARD; Scholak et al. (2021)) to down-weight or block invalid table transitions; (2) selective schema retrieval that returns only the relevant tables, keys, and concept mappings, reducing distractors from full-schema injection (e.g., RASL; Eben et al. (2025)); and (3) targeted post-generation checks of high-impact commitments, including alignment of time predicates with the question's temporal framing and of aggregation level (including `DISTINCT`) with the requested measure (e.g., Ren et al. (2025)). Overall, the residual gap is schema-grounded semantic faithfulness, motivating hybrid generate-and-check pipelines for reliable clinical text-to-SQL.

5. Conclusion

In conclusion, this benchmark provides an OMOP CDM v5.4-grounded, reliability-oriented testbed for EHR text-to-SQL under a strict, deployment-friendly output contract. The results show that parameter-efficient adaptation can achieve high executable correctness and substantially improved abstention behavior, while general-purpose baselines remain brittle under OMOP constraints. We expect this resource to support research on safer clinical analytics assistants that combine upstream entity linking with reliable SQL generation, constrained execution, and robust abstention/routing strategies. By providing an OMOP-grounded, execution-scored benchmark with calibrated abstention, we aim to accelerate safer free-text interfaces for cohort discovery and clinical analytics.

6. Limitations

We report results on concept-normalized inputs (entities expressed as OMOP concept identifiers), so we do not evaluate end-to-end entity linking from raw clinical text (e.g., using a clinical entity linker such as MedCAT (Kraljevic et al., 2021)). While we release both human-readable and concept-normalized questions to support end-to-end work, real deployments will face concept ambiguity and linking errors that can propagate to SQL execution. In deployment, linking uncertainty may require confirmation or abstention policies (e.g., abstaining when concept ambiguity is high).

Our split protocol is designed to test controlled generalization to unseen (template, SQL-variation) structures and paraphrased surface forms. It does not directly measure cross-institutional distribution shift across independent OMOP deployments, where data distributions, local conventions, and integrity constraints may differ; absolute execution accuracy may therefore change on real institutional OMOP databases, even if the strict SQL-only/<NO_SQL> output contract and our reliability-oriented evaluation remain applicable.

The benchmark is also limited in coverage and lexical diversity by its template-based construction; future work should expand diversity with closer coordination to real hospital information needs.

In addition, our unanswerable examples are adapted from EHRSQL-2024 rather than constructed natively for OMOP-specific failure modes, so future work should curate OMOP-native unanswerables tailored to schema-specific abstention scenarios.

Finally, execution-based scoring relies on a controlled synthetic OMOP PostgreSQL instance populated so that gold queries return non-empty outputs, enabling discriminative execution scoring while preserving the anti-memorization design. This synthetic database is not intended to mirror real prevalence, coding distributions, or integrity constraints; absolute execution metrics may therefore differ in practice. For time-sensitive templates (e.g., "today"), we fix the reference date to 2026-02-02 in the released evaluation dump.

7. Ethics Statements

This work supports clinical data access through a strict text-to-SQL (or <NO_SQL>) interface and is intended for research use. The released benchmark includes (i) natural-language questions, (ii) gold SQL queries, and (iii) a synthetic OMOP CDM v5.4 PostgreSQL dump designed for discriminative execution-based evaluation.

No MIMIC-IV content is distributed: we do not release any MIMIC-derived clinical notes, patient-level records, or other protected/identifiable health information.

Despite the strict output contract and execution-based evaluation, model-generated SQL may still be incorrect or unsafe if used without safeguards; we recommend defense-in-depth measures (e.g., sandboxed execution, timeouts, query allow/deny-lists, logging, and human oversight for high-stakes use).

8. Supplementary Materials

8.1. Dataset construction

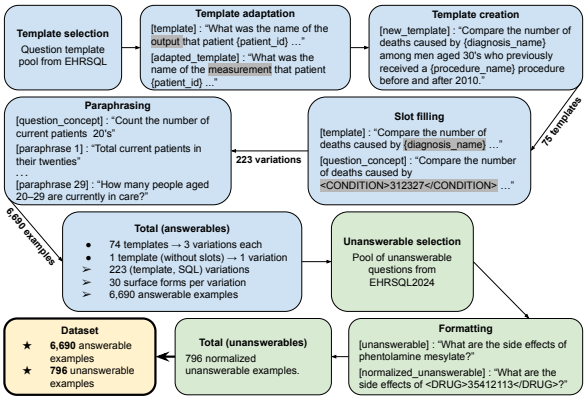

Figure 2: Dataset construction overview. OMOP-adapted EHRSQL templates yield 223 (template, SQL) variations and 6,690 answerable examples; separately selected and normalized EHRSQL-2024 unanswerables contribute 796 additional examples.

8.2. Split protocol

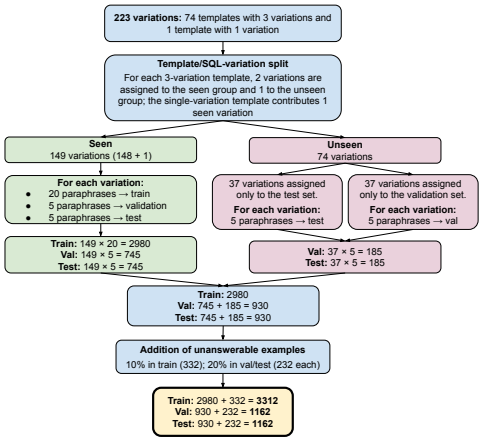

Figure 3: Leakage-resistant split construction at the (template, SQL)-variation level. Seen variations contribute to train/validation/test, unseen variations only to validation or test, and unanswerables are added at fixed rates.

8.3. Brief description of OMOP tables used in benchmark-specific examples

To improve accessibility for readers less familiar with OMOP CDM, Table 5 briefly summarizes several benchmark-relevant OMOP tables that may be less intuitive from their names alone.

Table	Role in OMOP / relevance here
visit_occurrence	Stores the coarse-grained care encounter (e.g., hospitalization or outpatient visit). Used for visit-level reasoning and last-visit queries.
visit_detail	Stores finer-grained sub-visit segments within a visit (e.g., ICU stay within a hospitalization). Important when the clinically correct answer depends on sub-visit granularity.
note	Stores unstructured clinical notes as text documents linked to patients and visits.
note_nlp	Stores NLP-derived annotations over clinical notes (e.g., extracted concepts/mentions), enabling note-linked structured querying.
episode	Represents a higher-level clinical episode grouping related events over time.
episode_event	Links an episode to the constituent OMOP events that belong to it.

Table 5: Short glossary of OMOP tables that may be less familiar to non-OMOP readers but appear in benchmark construction or error analysis.

8.4. Concrete examples of released instances

To make the benchmark format more explicit, we provide below one answerable example and one unanswerable example from the released JSON instances. As described in the main text, each answerable item may include a human-readable question (question_name), a concept-normalized question (question_concept), the gold SQL query, and the originating template. Unanswerable examples use the same input format but the only correct target output is the abstention token <NO_SQL>.

Answerable example. This example illustrates a concept-normalized input where the drug mention is replaced by an OMOP concept_id, while the corresponding human-readable version is also retained for readability.


```
{
  "question_name": "What are the top 4
      frequent prescribed drugs for
      patients who were also prescribed
      <DRUG>bisacodyl 10 MG Rectal
      Suppository [Bisac-Evac]</DRUG> at
       the same time on the last
      hospital visit?",
  "question_concept": "What are the top
      4 frequent prescribed drugs for
      patients who were also prescribed
      <DRUG>19041823</DRUG> at the same
      time on the last hospital visit?",
  "sql": "WITH last_visits AS (SELECT
      DISTINCT ON (vo.person_id) vo.
      visit_occurrence_id FROM
      visit_occurrence AS vo ORDER BY vo
      .person_id, vo.visit_start_date
      DESC), patients_with_19041823 AS (
      SELECT lv.visit_occurrence_id FROM
       last_visits AS lv JOIN
      drug_exposure AS de1 ON lv.
      visit_occurrence_id = de1.
      visit_occurrence_id WHERE de1.
      drug_concept_id = 19041823),
      co_prescribed_drugs AS (SELECT de2
      .drug_concept_id FROM
      drug_exposure de2 JOIN
      patients_with_19041823 AS p ON de2
      .visit_occurrence_id = p.
      visit_occurrence_id WHERE de2.
      drug_concept_id != 19041823)
      SELECT d.drug_concept_id AS
      concept_id, c.concept_name, COUNT
      (*) AS count FROM
      co_prescribed_drugs AS d JOIN
      concept AS c ON d.drug_concept_id
      = c.concept_id GROUP BY d.
      drug_concept_id, c.concept_name
      ORDER BY count DESC LIMIT 4;",
  "template": "what are_verb the top [
      n_rank] frequent prescribed drugs
      for patients who were also
```

```
      prescribed {drug_name} [
      time_filter_within] [
      time_filter_global1]?"
}
```

This example highlights the benchmark design choice of evaluating SQL generation from concept-normalized inputs: the model is expected to map the typed identifier <DRUG>19041823</DRUG> to the correct OMOP predicate structure (here, drug_exposure.drug_concept_id) and produce a single executable PostgreSQL query.

Unanswerable example. The following example illustrates the abstention setting. The question asks for a drug side effect, which cannot be answered from the OMOP database instance alone under our task definition. The only correct output is therefore the abstention token <NO_SQL>.

```
{
  "input": "What is the side effect of <
      DRUG>43833975</DRUG>?",
  "output": "<NO_SQL>"
}
```

This example belongs to the unanswerable portion of the benchmark and reflects the intended safety contract of the task: if the answer requires external or normative medical knowledge rather than information available in the structured OMOP database, the model must abstain rather than generate a plausible but unsupported SQL query.

8.5. Inference prompts used for LoRA models and schema-injected baselines

For reproducibility, we report below the exact prompt structure used at inference time. All models were run with deterministic decoding (temperature=0.0, top_p=1.0), and generations were truncated at the first semicolon so that outputs satisfied the single-statement contract. The same core instruction was used across LoRA models and baselines, with the main difference that baseline prompts additionally injected the PostgreSQL DDL schema into the user prompt.

Shared system instruction. The following system message was used in inference:

```
You are an assistant that generates
    valid PostgreSQL SQL queries for
    data stored in an electronic health
    record following the OMOP Common
    Data Model (CDM). Follow these rules
     strictly:
- Use only OMOP CDM tables, columns, and
     relationships.
- Never invent tables or columns.
- Never hallucinate schema details.
- Never use SELECT *.
- Terminate the SQL query with a
    semicolon.
- Output only the SQL query without
    prose, comments, or explanations.
- If the question cannot be answered
    from OMOP CDM data, output <NO_SQL>.
- Stop after the first semicolon.
```

LoRA inference prompt (Llama-3 chat format). For LoRA inference, the model was prompted in native Llama-3 chat format without schema injection. The user message only contained the question:

```
<|begin_of_text|><|start_header_id|>
    system<|end_header_id|>
[SYSTEM_MSG]<|eot_id|>
<|start_header_id|>user<|end_header_id|>
Generate the SQL query that answers the
    following question:
[QUESTION]
<|eot_id|>
<|start_header_id|>assistant<|
    end_header_id|>
```

Concretely, [SYSTEM_MSG] is replaced by the shared system instruction above, and [QUESTION] is replaced by the benchmark input string.

Schema-injected baseline prompts. For baseline models, the schema was injected into the prompt through the user message. The shared user-side content was:

```
You are given the database schema in
    PostgreSQL DDL.
Use it to write a correct query.

### Database schema (DDL)
[SCHEMA_DDL]

### Question
[QUESTION]
```

Here, [SCHEMA_DDL] denotes the OMOP PostgreSQL DDL block and [QUESTION] denotes the benchmark input.

Llama-3 baseline prompt. The Llama-3 baseline used the same native chat wrapper as the LoRA model, but with the schema-injected user block:

```
<|begin_of_text|><|start_header_id|>
    system<|end_header_id|>
[SYSTEM_MSG]<|eot_id|>
<|start_header_id|>user<|end_header_id|>
[USER_BLOCK_WITH_SCHEMA]
<|eot_id|>
<|start_header_id|>assistant<|
    end_header_id|>
```

where [USER_BLOCK_WITH_SCHEMA] is the schema-injected content shown above.

Mistral baseline prompt. The Mistral baseline used the tokenizer chat template when available, with the same two-message structure: (i) the shared system instruction, and (ii) the schema-injected user message. When the tokenizer chat template was unavailable, the fallback plain-text prompt was:

```
System: [SYSTEM_MSG]

User: [USER_BLOCK_WITH_SCHEMA]

Assistant:
```

This preserves exactly the same instruction content while adapting formatting to the tokenizer implementation.

Natural-SQL baseline prompt. The Natural-SQL baseline used the following task-style prompt:

```
# Task
[SYSTEM_MSG]

Generate a SQL query to answer the
    following question: `[QUESTION]`

### PostgreSQL Database Schema
The query will run on a database with
    the following schema:

[SCHEMA_DDL]

# SQL
Here is the SQL query that answers the
    question: `[QUESTION]`
```sql
```

Thus, the same instruction constraints were retained, but embedded in the Natural-SQL prompt format.

9. References

ChatDB. 2024. Natural-sql-7b. Hugging Face model card. Accessed: 2026-01-31.

Jeffrey Eben, Aitzaz Ahmad, and Stephen Lau. 2025. Rasl: Retrieval augmented schema linking for massive database text-to-sql. *arXiv preprint arXiv:2507.23104*. Accepted to the KDD Workshop on Structured Knowledge for Large Language Models (SKnowLLM '25).

Catherine Finegan-Dollak, Jonathan K. Kummerfeld, Li Zhang, Karthik Ramanathan, Sesh Sadasivam, Rui Zhang, and Dragomir Radev. 2018. Improving text-to-SQL evaluation methodology. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 351–360, Melbourne, Australia. Association for Computational Linguistics.

Maryam Garza, Guilherme Del Fiol, Jessica Tenenbaum, Anita Walden, and Meredith Nahm Zozus. 2016. Evaluating common data models for use with a longitudinal community registry. *Journal of Biomedical Informatics*, 64:333–341.

Yonatan Geifman and Ran El-Yaniv. 2019. Selectivenet: A deep neural network with an integrated reject option. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2151–2159. PMLR.

Aaron Grattafiori, Abhimanyu Dubey, Anish Jha, et al. 2024. The llama 3 herd of models.

George Hripcsak, Jon D. Duke, Nigam H. Shah, Christian G. Reich, Vojtech Huser, Martijn J. Schuemie, Marc A. Suchard, Rae Woong Park, Ian C. K. Wong, Peter R. Rijnbeek, Johan van der Lei, Nicole Pratt, G. Niklas Norén, Yu-Chuan (Jack) Li, Paul E. Stang, David Madigan, and Patrick B. Ryan. 2015. Observational health data sciences and informatics (ohdsi): Opportunities for observational researchers. In *Studies in Health Technology and Informatics*, volume 216, pages 574–578.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b.

Alistair E. W. Johnson, Lucas Bulgarelli, Lu Shen, Alvin Gayles, Ayad Shammout, Steven Horng, Tom J. Pollard, Sicheng Hao, Benjamin Moody, Brian Gow, Li-Wei H. Lehman, Leo A. Celi, and Roger G. Mark. 2023. Mimic-iv, a freely accessible electronic health record dataset. *Scientific Data*, 10(1):1.

Zeljko Kraljevic, Thomas Searle, Anthony Shek, Lukasz Roguski, Kawsar Noor, Daniel Bean, Aurelie Mascio, Leilei Zhu, Amos A. Folarin, Angus Roberts, Rebecca Bendayan, Mark P. Richardson, Robert Stewart, Anoop D. Shah, Wai Keong Wong, Zina Ibrahim, James T. Teo, and Richard
```

J. B. Dobson. 2021. Multi-domain clinical natural language processing with medcat: The medical concept annotation toolkit. *Artificial Intelligence in Medicine*, 117:102083.

Gyubok Lee, Woosog Chay, Seonhee Cho, and Edward Choi. 2024a. Trustsql: A reliability benchmark for text-to-sql models with diverse unanswerable questions. *arXiv preprint arXiv:2403.15879*.

Gyubok Lee, Hyeonji Hwang, Seongsu Bae, Yeonsu Kwon, Woncheol Shin, Seongjun Yang, Minjoon Seo, Jong-Yeup Kim, and Edward Choi. 2023. Ehrsql: A practical text-to-sql benchmark for electronic health records. *arXiv preprint arXiv:2301.07695*.

Gyubok Lee, Sunjun Kweon, Seongsu Bae, and Edward Choi. 2024b. Overview of the EHRSQL 2024 shared task on reliable text-to-SQL modeling on electronic health records. In *Proceedings of the 6th Clinical Natural Language Processing Workshop*, pages 644–654, Mexico City, Mexico. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Observational Health Data Sciences and Informatics (OHDSI). 2021. Omop common data model (cdm) v5.4 specifications. Specification webpage. Accessed: 2026-01-31.

OHDSI. 2025. Ohdsi/mimic: Mimic iv to omop cdm conversion (etl pipeline). GitHub repository (release v1.0.0). Accessed: 2026-02-11.

OpenAI. 2025a. Gpt-4.1 model (openai api documentation). Developer documentation. Accessed: 2026-02-11.

OpenAI. 2025b. Gpt-5.1 instant and gpt-5.1 thinking system card addendum. System card addendum (PDF). Accessed: 2026-02-11.

J. Marc Overhage, Patrick B. Ryan, Christian G. Reich, Abraham G. Hartzema, and Paul E. Stang. 2012. Validation of a common data model for active safety surveillance research. *Journal of the American Medical Informatics Association*, 19(1):54–60.

PostgreSQL Global Development Group. 2025. Postgresql documentation (current release). Online documentation. Accessed: 2026-01-31.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Christian Reich, Anna Ostropolets, Patrick Ryan, Peter Rijnbeek, Martijn Schuemie, Alexander Davydov, Dmitry Dymshyts, and George Hripcsak. 2024. Ohdsi standardized vocabularies—a large-scale centralized reference ontology for international data harmonization. *Journal of the American Medical Informatics Association*, 31(3):583–590.

Tonghui Ren, Chen Ke, Yuankai Fan, Yinan Jing, Zhenying He, Kai Zhang, and X. Sean Wang. 2025. The power of constraints in natural language to sql translation. *Proceedings of the VLDB Endowment*, 18(7):2097–2111.

Torsten Scholak, Nathan Schucher, and Dzmitry Bahdanau. 2021. Picard: Parsing incrementally for constrained auto-regressive decoding from language models. *arXiv preprint arXiv:2109.05093*.

Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. 2020. Rat-sql: Relation-aware schema encoding and linking for text-to-sql parsers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 7567–7578.

Chenglong Wang, Kedar Tatwawadi, Marc Brockschmidt, Po-Sen Huang, Yi Mao, Oleksandr Polozov, and Rishabh Singh. 2018. Robust text-to-sql generation with execution-guided decoding.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir R. 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 (EMNLP)*, pages 3911–3921.

Angelo Ziletti and Leonardo DAmbrosi. 2024. Retrieval augmented text-to-SQL generation for epidemiological question answering using electronic health records. In *Proceedings of the 6th Clinical Natural Language Processing Workshop*, pages 47–53, Mexico City, Mexico. Association for Computational Linguistics.