CONSTRAINT LINEAGE AS AN AUDITABLE CONTROL OBJECT FOR CONVERSATIONAL TEXT-TO-SQL

Anonymous authors
Paper under double-blind review

ABSTRACT

Conversational parsers can carry a predicate from one SQL turn into a later one even when the predicate's origin no longer justifies its persistence. Local executability and edit traces reveal that a clause survived, but they do not encode the evidence under which it became state. We introduce LINEAGEGATE, an auditable state-commit interface that decomposes SQL into atomic hypotheses and records each hypothesis's origin, observed support, admission, endorsement, and reuse. The interface separates runtime selection features from an independent post-hoc entitlement judgment, making it possible to ask whether a retained constraint was licensed when it entered dialogue state rather than merely whether the current query is executable. We formalize the lifecycle of an inherited constraint and define three complementary measurement objects: Constraint Propagation Rate (CPR) for unsupported hypotheses that cross the commit boundary and reach first reuse, Downstream Error Amplification (DEA) for counterfactual attribution of a later failure, and Provenance Consistency (PC) for supported active state. The design composes a learned semantic selector with deterministic authorization and SQL-safety checks, while keeping their responsibilities explicit. This formulation turns conversational SQL continuity into a reviewable control object: an evaluator can inspect the admission history of each inherited clause, distinguish prediction from entitlement measurement, and trace a later reuse back to the evidence available at commitment time.

1 INTRODUCTION

A conversational parser can introduce a predicate while answering one request and preserve it through later SQL revisions even when the user never supplied evidence for that predicate. For example, a request for orders above `$500` can acquire `currency='EUR'` during generation; an explanation can then describe euros as settled context, and the filter can remain in a later export. Every query in this trajectory may execute, and a conventional edit trace will faithfully report that the filter persisted. The missing object is the basis on which the filter was entitled to become persistent dialogue state.

We call this basis *constraint lineage*. Each SQL candidate is decomposed into atomic hypotheses—bindings, predicates, joins, ordering choices, and aggregation scopes—and each hypothesis carries an explicit record of its origin, observed support, admission status, endorsement, and reuse path. The resulting state representation distinguishes two identical predicates whose histories differ: a clause grounded in an explicit request and a clause inherited from an unsupported guess have different state-commit records even when their current SQL text is identical.

This paper develops LINEAGEGATE, an evidence-conditioned control interface for that state-commit boundary. The controller serializes the lineage of a candidate's active hypotheses for utility, policy, and provenance scoring; it commits state only after admission and a separate deterministic authorization and SQL-safety check. An independent entitlement predicate evaluates whether a committed hypothesis was licensed by the utterance, schema documentation, grounded prior state, or bounded execution facts. Keeping runtime features and post-hoc entitlement labels separate makes the record usable for both selection and audit.

The contribution is a concrete language for identifying where conversational continuity becomes consequential. It formalizes injection, admission, endorsement, and first reuse as distinct transitions;

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

defines Constraint Propagation Rate (CPR), Downstream Error Amplification (DEA), and Provenance Consistency (PC) over those transitions; and specifies an evaluator that can trace a later reused clause to the support available when it entered state. This object gives conversational text-to-SQL systems an inspectable control surface for preserving justified context across turns.

2 Related Work

Following evaluation on WikiSQL and Spider, single-turn text-to-SQL progressed to intermediate-representation, graph-based, schema-aware, and constrained-decoding parsers, before advancing to prompt-based systems (Zhong et al., 2017; Yu et al., 2018; Finegan-Dollak et al., 2018; Guo et al., 2019; Bogin et al., 2019; Lin et al., 2020; Wang et al., 2020; Rubin & Berant, 2021; Scholak et al., 2021; Li et al., 2023; Sun et al., 2023; Pourreza & Rafiei, 2023). Conversational benchmarks such as SParC and CoSQL require cross-turn revision (Yu et al., 2019b;a). EditSQL and IGSQL encode prior SQL to predict the next query (Zhang et al., 2019; Cai & Wan, 2020); CoE-SQL makes an edit chain explicit (Zhang et al., 2024). These systems represent what prior information persists. Constraint lineage adds the evidence-bearing admission history of each retained clause, so an inherited predicate remains associated with the support path under which it entered state.

ReAct, Reflexion, Self-Refine, verification prompting, and LLM-as-a-judge systems use tool feedback, critique, or preference scoring to improve candidate selection (Yao et al., 2023; Shinn et al., 2023; Madaan et al., 2023; Dhuliawala et al., 2024; Zheng et al., 2023). Their decision process can incorporate dialogue context and a current candidate. LineageGate exposes a finer-grained state object: each candidate is paired with the origins, admission events, endorsements, and reuse paths of its atomic SQL hypotheses. This supplies a decision surface for persistent state and a corresponding post-hoc audit target.

Grounding and attribution methods study whether generated content is supported by evidence (Ji et al., 2023; Lewis et al., 2020; Gao et al., 2023; Kadavath et al., 2022; Min et al., 2023). VET is a close execution-trace counterpart, constraining reasoning steps to retained schema items and requiring provenance-consistent values, joins, and aggregations (Wang et al., 2026). Constraint lineage focuses on the lifecycle through which an executable hypothesis is promoted into conversational state. Database provenance instead explains how query outputs depend on source records and transformations (Cui et al., 2000; Buneman et al., 2001; Green et al., 2007; Cheney et al., 2009; Herschel et al., 2017). The proposed representation concerns the upstream support history of a semantic constraint before it influences a later query.

3 Method

3.1 Constraint Lineage at the State-Commit Boundary

We model a dialogue with schema $\mathcal{S}$, utterances $u_{1:t}$, prior queries $q_{1:t-1}$, and explanations $e_{1:t-1}$. For every scored candidate q, SQL AST differencing extracts a population $C_t(q)$ of atomic hypotheses: table and column bindings, predicates, join paths, ordering choices, and aggregation scopes. The lineage state $\mathcal{G}_t = (V_t, E_t, z_t, \mathcal{A}_t)$ contains a node for every historical hypothesis, typed edges for origin, admission, endorsement, and reuse, an admission status $z_t(c)$, and the active set $\mathcal{A}_t$ present in the selected query. Node identity is a canonical AST signature containing the hypothesis type, fully qualified schema operands, normalized operator, and normalized literal. Aliases and commutative join order therefore identify the same node, whereas a changed operand, value, or scope produces a new proposal.

Four transitions define the lifecycle. *Injection* records a newly proposed signature in a candidate. *Admission* commits a proposal whose candidate passes the score checks. *Endorsement* records that an admitted explanation or later interpretation presents an admitted hypothesis as settled. *Reuse* records the appearance of an admitted hypothesis in a later selected query. Injection is an audit transition, while admission is the event that promotes a hypothesis into active dialogue state. The runtime commit operator is

$$\rho_t(c) : (\mathcal{G}_{t-1}, c, \widehat{\mathcal{E}}_t, \mathbf{A}_t, \boldsymbol{\tau}) \longrightarrow z_t(c), \tag{1}$$

where $\widehat{\mathcal{E}}_t$ is the runtime evidence record, $\mathbf{A}_t$ contains candidate scores, and $\boldsymbol{\tau}$ contains fixed admission thresholds.

3.2 Entitlement and Trace Metrics

The audit evidence is $\mathcal{E}_t = (u_t, D, \mathcal{G}_{t-1}, \ell_{1:t-1})$, where D is schema documentation and $\ell_{1:t-1}$ denotes bounded execution facts. A hypothesis has entitlement when it is explicit in the current utterance, stated by schema documentation, required by a previously entitled constraint, or recorded as a fact by a prior entitled execution. Execution success alone never supplies entitlement, and output derived from an unsupported ancestor cannot supply entitlement to that ancestor. Current-turn contradiction removes inherited entitlement; unresolved conflicts receive no support. For every active $c \in \mathcal{A}_t$, the post-hoc label

$$s_t^\star(c) = \mathbf{1}\{\mathcal{E}_t \vdash c\} \tag{2}$$

records whether the evidence licenses c under these rules. This audit target is distinct from the runtime evidence features $\widehat{\mathcal{E}}_t$ used by selection.

Let $\mathcal{I}_t$ contain audit-unsupported signatures whose first appearance in the scored candidate population occurs at turn t, and let $\mathcal{R}_t$ contain lineages that have crossed admission and are first reused without fresh support at turn t. Each lineage enters $\mathcal{R}_t$ at most once. Constraint Propagation Rate and Provenance Consistency are

$$\mathrm{CPR} = 100\frac{\sum_t |\mathcal{R}_t|}{\sum_t |\mathcal{I}_t|}, \qquad \mathrm{PC} = 100\frac{\sum_t \sum_{c\in\mathcal{A}_t} s_t^\star(c)}{\sum_t |\mathcal{A}_t|}. \tag{3}$$

Both metrics are defined as zero when their denominators are empty.

Let $\mathcal{F}$ be the incorrect evaluated turns and let $\mathcal{P}_t \subseteq \mathcal{R}_{<t}$ be propagated unsupported hypotheses active at $t \in \mathcal{F}$. For $c \in \mathcal{P}_t$, let $a_t(c)$ indicate whether removing the minimal AST fragment induced by c changes the turn from incorrect to correct under the same result-equivalence criterion. Downstream Error Amplification is

$$\mathrm{DEA} = 100\frac{\sum_{t\in\mathcal{F}} \mathbf{1}\left[\sum_{c\in\mathcal{P}_t} a_t(c) > 0\right]}{|\mathcal{F}|}. \tag{4}$$

This turn-level aggregation prevents several correlated fragments from counting the same repaired failure more than once.

3.3 LineageGate Realization

LineageGate instantiates evidence-conditioned selection under a fixed generator-call budget. At each turn, a planner proposes candidate SQL–explanation pairs. Each candidate is canonicalized, matched against $\mathcal{G}_{t-1}$, and serialized as an ordered list of hypotheses with type, source tag, origin turn, support record, prior status, and endorsement/reuse path. New signatures are proposed; exact carried signatures retain their lineage and receive current runtime evidence features.

Utility, policy-compliance, and provenance critics score the candidate serialization. Writing $h_t(q)$ for that serialization, the candidate-level outputs are A_t^{util}, A_t^{pol}, and A_t^{prov}. Their weighted combination ranks candidates, and admission requires the corresponding score checks to pass. When a check fails, the controller can refine the candidate using the failed dimensions and recompute the lineage representation. After learned selection, a deterministic gate rejects forbidden objects and unsafe SQL constructs independently of the learned scores. An admitted candidate commits its proposed nodes, records eligible endorsement edges, and records reuse edges for inherited admitted nodes. The post-hoc entitlement label remains an independent audit object, so an admitted hypothesis can be inspected against the evidence that was available at commitment time.

4 Evaluation Protocol

The evaluation protocol follows the carry-over trajectory from candidate injection to downstream reuse. Its primary comparison holds the generator interface, schema serialization, dialogue context, executor, decoding policy, and generator-call budget fixed while comparing an edit-continuity controller with an evidence-conditioned lineage controller. The unit for interaction success is a dialogue

whose evaluated turns all satisfy execution-result equivalence; the unit for turn execution is an evaluated turn. CPR uses first-appearing unsupported candidate hypotheses as its denominator, PC uses active hypotheses, and DEA uses incorrect turns and counts a turn once when a minimal-fragment intervention repairs it.

The protocol separates controller development from one-time evaluation by grouping databases and dialogues across partitions and freezing prompts, checkpoints, weights, and thresholds before the evaluation campaign. It defines matched conversational settings on CoSQL and SParC (Yu et al., 2019a;b), a single-turn SQL setting on Spider (Yu et al., 2018), and database-disjoint analyst-session settings. Controlled settings vary documentation availability, schema breadth, SQL dialect, and enforcement configuration while preserving the measurement definitions.

Interaction and turn outcomes are summarized across independently seeded controller executions. CPR, DEA, and PC are pooled ratios over their explicitly stated units. Paired bootstrap intervals resample the dialogue or session while preserving method pairs inside each resampled unit. The annotation protocol assigns support, endorsement, reuse, and counterfactual-attribution labels independently of model identity; disagreement adjudication creates the reference labels used to evaluate the automatic lineage extractor.

For governance, the protected objects are inherited constraints, authorized database objects, and unsafe SQL operations. The threat model includes conversational manipulation and adaptive requests against a fixed policy surface. Learned selection supplies semantic ranking, while deterministic parsing and database-native controls enforce the enumerated authorization and SQL-safety rules. This division makes the control boundary, assumptions, failure modes, utility measurement, and evidence trace separately inspectable.

5 Evaluation Objects

Constraint lineage makes a conversational SQL controller evaluable at the state-commit boundary. The primary outcome pairs complete-dialogue correctness with CPR: the former asks whether the selected dialogue is correct under result equivalence, while the latter asks whether an unsupported proposal was admitted and later reused. Reporting both quantities prevents an audit from treating reduced carry-over as sufficient without accounting for task behavior.

DEA localizes the downstream consequence of a propagated hypothesis through minimal-fragment intervention. PC summarizes the support status of active state. Together, these quantities distinguish an unsupported proposal that was considered and blocked from one that was ratified, endorsed, and reused. The lineage graph also makes each measurement traceable to a bounded set of utterance, schema, state, and execution facts.

The evaluation protocol further separates semantic selection from deterministic enforcement. An enforcement analysis reports the protected object, attacker capability, trust assumptions, policy property, utility outcome, observed failure mode, and internal evidence trace. This organization establishes an auditable evidence hierarchy for conversational text-to-SQL systems whose output must remain both useful and authorized across turns.

6 Discussion

The central distinction is between exact continuity and justified continuity. An edit history preserves which clauses persisted; constraint lineage preserves the support history under which each clause entered and remained in active state. That difference provides a common interface for runtime selection, independent entitlement measurement, and post-hoc investigation of a later reuse.

The lifecycle also clarifies the meaning of the measurements. CPR is a first-reuse rate over unsupported proposals, PC concerns active state, and DEA is a counterfactual attribution over incorrect turns. The deterministic gate has a distinct role: it enforces enumerated authorization and SQL-safety rules after semantic selection. Keeping these mechanisms distinct lets an evaluator attribute a state transition to the relevant control boundary.

7 CONCLUSION

Constraint lineage makes the state-commit history of an inherited SQL clause explicit. LINEAGEGATE combines that representation with an entitlement predicate, lifecycle semantics, trace metrics, learned selection, and deterministic enforcement. The resulting control object gives conversational text-to-SQL a precise basis for auditing whether persistent state remained justified across turns.

AI USE STATEMENT

Generative AI was used only for manuscript writing, editing, and presentation. The authors reviewed the edits and remain responsible for the final content.

ETHICS STATEMENT

Executable natural-language data interfaces require safeguards against unauthorized access, unsafe SQL, and misplaced trust in fluent queries. The proposed design records the evidence and admission history of inherited constraints, applies deterministic rejection of forbidden objects and unsafe statements, and treats database-native controls as the final authorization boundary. This separation supports auditing of the learned selector, the policy surface, and the enforcement decision under changing workloads.

REPRODUCIBILITY STATEMENT

Sections 3 and 4 specify the state representation, scoring and enforcement procedure, evaluation units, metrics, denominators, and uncertainty conventions; the appendix gives the entitlement rules and state-commit protocol.

REFERENCES

Ben Bogin, Jonathan Berant, and Matt Gardner. Representing schema structure with graph neural networks for text-to-SQL parsing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4560–4565. Association for Computational Linguistics, 2019. doi: 10.18653/v1/P19-1448. URL https://aclanthology.org/P19-1448/.

Peter Buneman, Sanjeev Khanna, and Wang-Chiew Tan. Why and where: A characterization of data provenance. In *Database Theory—ICDT 2001*, volume 1973 of *Lecture Notes in Computer Science*, pp. 316–330. Springer, 2001. doi: 10.1007/3-540-44503-X_20. URL https://doi.org/10.1007/3-540-44503-X_20.

Yitao Cai and Xiaojun Wan. IGSQL: Database schema interaction graph based neural model for context-dependent text-to-SQL generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6903–6912. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.emnlp-main.560. URL https://aclanthology.org/2020.emnlp-main.560/.

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

Yingwei Cui, Jennifer Widom, and Janet L. Wiener. Tracing the lineage of view data in a warehousing environment. *ACM Transactions on Database Systems*, 25(2):179–227, 2000. doi: 10.1145/357775.357777. URL https://doi.org/10.1145/357775.357777.

Shehzaad Dhuliawala, Mojtaba Komeili, Jing Xu, Roberta Raileanu, Xian Li, Asli Celikyilmaz, and Jason Weston. Chain-of-verification reduces hallucination in large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 3563–3578. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-acl.212. URL https://aclanthology.org/2024.findings-acl.212/.

Catherine Finegan-Dollak, Jonathan K. Kummerfeld, Li Zhang, Karthik Ramanathan, Sesh Sadasivam, Rui Zhang, and Dragomir Radev. Improving text-to-SQL evaluation methodology. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 351–360. Association for Computational Linguistics, 2018. doi: 10.18653/v1/P18-1033. URL https://aclanthology.org/P18-1033/.

Tianyu Gao, Howard Yen, Jiatong Yu, and Danqi Chen. Enabling large language models to generate text with citations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 6465–6488. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.398. URL https://aclanthology.org/2023.emnlp-main.398/.

Todd J. Green, Grigoris Karvounarakis, and Val Tannen. Provenance semirings. In *Proceedings of the 26th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS)*, pp. 31–40. ACM, 2007. doi: 10.1145/1265530.1265535. URL https://doi.org/10.1145/1265530.1265535.

Jiaqi Guo, Zecheng Zhan, Yan Gao, Yan Xiao, Jian-Guang Lou, Ting Liu, and Dongmei Zhang. Towards complex text-to-SQL in cross-domain database with intermediate representation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4524–4535. Association for Computational Linguistics, 2019. doi: 10.18653/v1/P19-1444. URL https://aclanthology.org/P19-1444/.

Melanie Herschel, Ralf Diestelkämper, and Houssem Ben Lahmar. A survey on provenance: What for? what form? what from? *The VLDB Journal*, 26(6):881–906, 2017. doi: 10.1007/s00778-017-0486-1. URL https://doi.org/10.1007/s00778-017-0486-1.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Yejin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38, 2023. doi: 10.1145/3571730. URL https://doi.org/10.1145/3571730.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, et al. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*, 2022. doi: 10.48550/arXiv.2207.05221. URL https://arxiv.org/abs/2207.05221.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 9459–9474, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html.

Haoyang Li, Jing Zhang, Cuiping Li, and Hong Chen. RESDSQL: Decoupling schema linking and skeleton parsing for text-to-SQL. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(11):13067–13075, 2023. doi: 10.1609/AAAI.V37I11.26535. URL https://doi.org/10.1609/AAAI.V37I11.26535.

Xi Victoria Lin, Richard Socher, and Caiming Xiong. Bridging textual and tabular data for cross-domain text-to-SQL semantic parsing. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 4870–4888. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.findings-emnlp.438. URL https://aclanthology.org/2020.findings-emnlp.438/.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pp. 46534–46594, 2023. URL https://openreview.net/forum?id=S37hOerQLB.

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. FActScore: Fine-grained atomic evaluation of factual precision in long form text generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 12076–12100. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.741. URL https://aclanthology.org/2023.emnlp-main.741/.

Mohammadreza Pourreza and Davood Rafiei. DIN-SQL: Decomposed in-context learning of text-to-SQL with self-correction. In *Advances in Neural Information Processing Systems*, volume 36, pp. 36339–36348, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/72223cc66f63ca1aa59edaec1b3670e6-Abstract.html.

Ohad Rubin and Jonathan Berant. SmBoP: Semi-autoregressive bottom-up semantic parsing. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pp. 311–324. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.naacl-main.29. URL https://aclanthology.org/2021.naacl-main.29/.

Torsten Scholak, Nathan Schucher, and Dzmitry Bahdanau. PICARD: Parsing incrementally for constrained auto-regressive decoding from language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 9895–9901. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.emnlp-main.779. URL https://aclanthology.org/2021.emnlp-main.779/.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pp. 8634–8652, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/1b44b878bb782e6954cd888628510e90-Abstract-Conference.html.

Ruoxi Sun, Sercan Arik, Rajarishi Sinha, Hootan Nakhost, Hanjun Dai, Pengcheng Yin, and Tomas Pfister. SQLPrompt: In-context text-to-SQL with minimal labeled data. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 542–550. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.findings-emnlp.39. URL https://aclanthology.org/2023.findings-emnlp.39/.

Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. 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)*, pp. 7567–7578. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.acl-main.677. URL https://aclanthology.org/2020.acl-main.677/.

Dongyu Wang, Jingyu Li, Lan Zhang, Ganggang Yu, and Liang Huang. VET: Verifiable execution tracing for reliable text-to-SQL generation. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 30867–30883. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.findings-acl.1544. URL https://aclanthology.org/2026.findings-acl.1544/.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 3911–3921. Association for Computational Linguistics, 2018. doi: 10.18653/v1/D18-1425. URL https://aclanthology.org/D18-1425/.

Tao Yu, Rui Zhang, Heyang Er, Suyi Li, Eric Xue, Bo Pang, Xi Victoria Lin, Yi Chern Tan, Tianze Shi, Zihan Li, Youxuan Jiang, Michihiro Yasunaga, Sungrok Shim, Tao Chen, Alexander Fabbri, Zifan

Li, Luyao Chen, Yuwen Zhang, Shreya Dixit, Vincent Zhang, Caiming Xiong, Richard Socher, Walter Lasecki, and Dragomir Radev. CoSQL: A conversational text-to-SQL challenge towards cross-domain natural language interfaces to databases. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 1962–1979. Association for Computational Linguistics, 2019a. doi: 10.18653/v1/D19-1204. URL https://aclanthology.org/D19-1204/.

Tao Yu, Rui Zhang, Michihiro Yasunaga, Yi Chern Tan, Xi Victoria Lin, Suyi Li, Heyang Er, Irene Li, Bo Pang, Tao Chen, Emily Ji, Shreya Dixit, David Proctor, Sungrok Shim, Jonathan Kraft, Vincent Zhang, Caiming Xiong, Richard Socher, and Dragomir Radev. SParC: Cross-domain semantic parsing in context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4511–4523. Association for Computational Linguistics, 2019b. doi: 10.18653/v1/P19-1443. URL https://aclanthology.org/P19-1443/.

Hanchong Zhang, Ruisheng Cao, Hongshen Xu, Lu Chen, and Kai Yu. CoE-SQL: In-context learning for multi-turn text-to-SQL with chain-of-editions. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 6487–6508, Mexico City, Mexico, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.361. URL https://aclanthology.org/2024.naacl-long.361/.

Rui Zhang, Tao Yu, Heyang Er, Sungrok Shim, Eric Xue, Xi Victoria Lin, Tianze Shi, Caiming Xiong, Richard Socher, and Dragomir Radev. Editing-based SQL query generation for cross-domain context-dependent questions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 5338–5349. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-1537. URL https://aclanthology.org/D19-1537/.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pp. 46595–46623, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/91f18a1287b398d378ef22505bf41832-Abstract-Datasets_and_Benchmarks.html.

Victor Zhong, Caiming Xiong, and Richard Socher. Seq2SQL: Generating structured queries from natural language using reinforcement learning. *arXiv preprint arXiv:1709.00103*, 2017. doi: 10.48550/arXiv.1709.00103. URL https://arxiv.org/abs/1709.00103.

A Additional Method Details

A.1 State-Commit Protocol

For each dialogue turn, the controller performs the following state-commit procedure.

1. Parse each candidate query into canonical atomic hypotheses and align each signature with the preceding lineage graph.
2. Attach the current utterance, schema documentation, previously admitted entitled state, and bounded execution facts to the candidate's runtime evidence record.
3. Score the serialized candidate with the utility, policy, and provenance critics, then admit an eligible candidate under the fixed controller policy.
4. Apply deterministic authorization and SQL-safety checks to the admitted candidate.
5. Commit the candidate's proposed nodes only after both admission stages pass; record eligible endorsement edges and reuse edges for inherited active nodes.
6. Independently evaluate the committed active state with the entitlement predicate when producing CPR, PC, or DEA labels.

The protocol intentionally keeps a runtime score and a post-hoc entitlement label separate. The score can use only the controller's available features, whereas entitlement is evaluated from the audit evidence object and its explicit rules. This separation allows a later audit to identify the exact transition at which an unsupported hypothesis became persistent state.

A.2 Entitlement Rules

Table 1: Type-specific entitlement rules for atomic SQL hypotheses.

Hypothesis type	Entitled evidence	Boundary condition
Table or column binding	Explicit user mention, documented schema semantics, or a binding required to realize an entitled constraint	A familiar name alone does not license a new binding.
Predicate	Explicit value or relation in the request, documented domain meaning, or a grounded antecedent that entails the predicate	An execution result cannot retroactively license an unsupported predicate.
Join path	A documented relationship required by entitled operands	The path is limited to the relationship needed for the grounded constraint.
Ordering or aggregation scope	Explicit request language or a documented semantic convention	A prior query's choice is evidence only when its own lineage is entitled.
Inherited hypothesis	Current support or a non-contradicted entitled state transition	A current contradiction withdraws inherited entitlement.

A.3 Qualitative Lineage Trace

Consider a dialogue in which an analyst asks for orders above `$500`. Suppose a candidate query adds `currency='EUR'` even though the utterance and schema documentation provide no support for a currency restriction. The AST extractor creates a predicate node whose origin is the candidate generation event and whose support record is empty. If the candidate is admitted, the node receives an admission edge; if an explanation presents the currency restriction as settled, the graph records an endorsement edge. When a later query retains the same predicate, the graph records reuse and permits the evaluator to test whether the predicate received fresh support.

An otherwise identical predicate can follow a different lineage. If the analyst explicitly requests euro-denominated orders, the current utterance licenses the predicate at injection, and the support record accompanies the admitted node into later state. The two SQL fragments are textually identical, but their state-commit histories provide different inputs to selection and different audit outcomes.

A.4 Metric Accounting

CPR counts a lineage when an unsupported proposal reaches its first downstream reuse; repeated later appearances do not create additional numerator events. PC assesses the support status of every active hypothesis at the relevant turn. DEA evaluates a failure turn once, even if several propagated hypotheses are implicated by minimal-fragment intervention. An evaluation report therefore records the unit of analysis, numerator, denominator, zero-denominator behavior, paired resampling unit, and support definition alongside each metric.

A.5 CONTROL-BOUNDARY MATRIX

The matrix turns lifecycle semantics into distinct review locations. A state-commit inquiry asks what licensed admission at the commitment turn, whereas a first-reuse inquiry asks whether the stored hypothesis still has support before it influences later SQL. Ranking and enforcement remain separate: an audit should not treat a semantic rationale for selecting a candidate as authorization to execute it.

Table 2: Control boundaries exposed by the lineage representation.

Boundary	Protected object	Audit question
State commit	Candidate hypothesis entering active dialogue state	What evidence licensed admission at this turn?
First reuse	Inherited hypothesis appearing in later SQL	Did the hypothesis retain or acquire support before influencing a later query?
Semantic selection	Candidate ranking under runtime evidence	Which provenance, utility, or policy signal determined the ranking?
Deterministic enforcement	Authorized schema objects and SQL constructs	Which enumerated rule admitted or rejected the selected candidate?

The resulting record attributes a later concern to the boundary that governed the relevant transition, rather than to an undifferentiated controller score. This also makes the protected object and the evidence required for each audit question explicit.