AIDEN: An Agentic Intention-Driven Assistant for Exploration and Navigation of Data Lakes

Ibraheem Taha
Aalborg University, National and Kapodistrian University of Athens, and Athena Research Center
Aalborg, Denmark
ibta@cs.aau.dk

Abstract

Open data lakes are increasingly used in scientific and societal analysis, but finding useful tables remains difficult when analysts do not know the data in advance. Existing table-discovery systems rank tables by query relevance, but often ignore the task context and the next operation the analyst needs. We present AIDEN, an agentic LLM framework for intention-aware table discovery in data lakes. AIDEN extracts query, intention, and operation signals, maintains session state, retrieves candidate tables, and uses an LLM-based recommender to rerank candidates for the current task. We evaluate AIDEN on a LakeBench-based benchmark with 5,593 tables, 100 query tables, manually validated requests, and operation and intention labels. Reranking improves nDCG@10 from 0.383 to 0.394 on join queries and from 0.539 to 0.575 on union queries. Signal extraction achieves 0.967 macro-F1 for intention labels and 0.771 macro-F1 for operation labels using request text alone. AIDEN frames data-lake exploration as a retrieval- and tool-augmented LLM agent for structured data, connecting memory, task understanding, and recommendation for scientific and societal analysis.

Keywords

Agentic AI, LLMs, Data Lakes, Intention-aware Recommendation

ACM Reference Format:
Ibraheem Taha. 2026. AIDEN: An Agentic Intention-Driven Assistant for Exploration and Navigation of Data Lakes. In *KDD'26 SciSoc Agents & LLMs Workshop*. ACM, New York, NY, USA, 5 pages.

1 Introduction

Open data repositories are becoming an important substrate for scientific and societal analysis [3]. Public-health agencies, urban planners, environmental scientists, and policy researchers increasingly draw on multi-million-table repositories of open data to ground decisions that affect real people [3, 10]. Yet the ability of *non-expert* domain analysts to navigate these repositories has not kept pace with their growth. This gap is especially important as LLM agents are increasingly used to support scientific and societal data work, where they must retrieve data, interpret user goals, and recommend useful next steps. Substantial progress on table search has been made under the assumption that the user's query is a complete specification of what they want [5, 6, 8]. In practice, it is not.

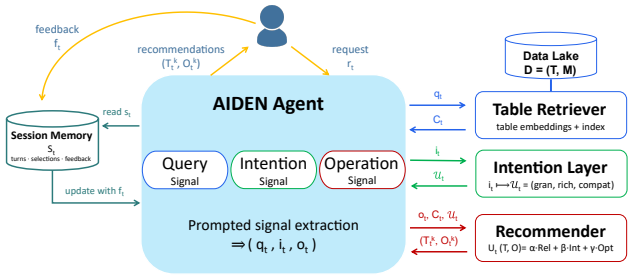

Figure 1: AIDEN Architecture. A central LLM agent extracts (q_t, i_t, o_t), invokes retrieval, intention modeling, and recommendation, and updates memory s_t from user feedback f_t.

Two fundamental limitations of current systems become visible whenever the user lacks prior knowledge of the underlying data, a routine condition in data lakes, which by design lack a global schema and explicit inter-table relationships.

Relevance is not intention. The same natural-language query can correspond to fundamentally different analytical intentions. A city-planning analyst asking for "public transit usage and neighborhood socioeconomic indicators" may intend to analyze trends across neighborhoods, to fit a predictive model from individual-level records, or to prepare an integrated dataset for downstream reporting. Each intention implies a different notion of table usefulness: aggregated summaries suit the first, microdata the second, and join-compatible reference tables the third. A retriever may return tables that are relevant to the user's query, but are still useless for the analyst's actual task. Without a way to capture this task context, the system cannot know which tables are truly useful.

Retrieval is not the end of the task. Analysts rarely stop after retrieval. Exploration continues as they decide what to *do* next: filter, join, union, aggregate, or clarify. Recent work on data-preparation prediction [2] and visualization recommendation [7] shows that predicting the user's next operation is a useful task. But these systems operate *after* discovery, treat retrieval as a fixed upstream input, and do not feed user intention back into the retrieval step. A central limitation of existing table discovery systems is that table retrieval and analytical use remain architecturally decoupled: systems can identify relevant tables, but they do not explicitly model how those tables should support the analyst's next step.

Our position. We argue that table discovery should be framed not as a one-shot search problem, but as an interactive agentic process: a tool-augmented LLM agent that perceives the user's evolving goal through dialogue, maintains session memory, invokes tools

KDD'26 SciSoc Agents & LLMs Workshop, Jeju, Republic of Korea

for retrieval, intention extraction, and operation extraction, and recommends both tables and next-step operations under an explicit utility function. This framing aligns with recent agentic-AI patterns emerging in scientific and societal applications, including agentic chemistry workflows [1], multi-agent clinical decision support [16], discourse-augmented agentic legal reasoning [11], and strategy-guided conversational agents for cognitive support in older adults [21]. What unites these systems is the decomposition of a complex analytical task into perception, memory, planning, and tool use, a decomposition that data-lake interfaces have so far lacked. We summarize our contribution in this paper as follows:

- We formulate table discovery as an agentic recommendation problem over tables, operations, and session state (Section 3).
- We present AIDEN, a framework that extracts query, intention, and operation signals, retrieves candidate tables, and reranks them through a task-aware recommender (Section 4).
- We build a benchmark with manually validated requests and operation and intention labels (Section 5).
- We evaluate retrieval, recommender reranking, and signal extraction, showing how each component contributes to intention-aware table discovery under explicit retrieval, operation, and intention labels (Section 5).

These contributions make AIDEN a concrete step toward agentic LLM systems for structured data discovery, combining retrieval, tool use, memory, and evaluation in practical data-lake tasks.

2 Related Work

We review three lines of work relevant to AIDEN.

Table search in data lakes. Aurum introduced a graph-based view of data lakes that supports navigation across heterogeneous sources [6]. SANTOS introduced relationship-based semantics between column pairs for union search [8]. Starmie learns contextualized column embeddings via contrastive pre-training and remains a strong baseline on standard benchmarks [5]. Join discovery has also been studied through column-overlap and learned matching approaches such as DeepJoin [4]. TabSketchFM uses data sketches to preserve semantics at lake scale [9]. Benchmarks have co-evolved: TUS [14], SANTOS [8], and LakeBench [3]. All of these systems assume the query is the complete specification of what the user wants [20]. AIDEN relaxes this assumption by treating the query as one of three signals extracted from a richer interaction context.

Next-operation prediction. A complementary literature predicts what the user should do *after* discovery. Auto-Prep jointly predicts transformations and joins using a graph-based formulation inspired by Steiner trees and outperforms GPT-4 on real Power BI projects [10]. CoWrangler recommends data-wrangling scripts [2]. LakeVisage proposes visualization recommendations over data-lake tables [7]. These systems do not condition recommendations on intention extracted from dialogue, and they treat retrieval as a separate step. AIDEN closes this loop by using extracted task signals to connect retrieval, operation recommendation, and session context.

Agentic AI and conversational search. A recent survey documents the iterative nature of information seeking and the latent, partially expressed nature of user intent in conversational search, but its development has been document-centric rather than tabular [13]. In parallel, the agentic-AI community has shown that decomposing complex analytical tasks into LLM-driven perception, memory, planning, and tool use yields gains in scientific and societal applications [1, 11, 16, 21]. AIDEN brings this agentic pattern to the data-lake setting, treating retrieval, intention extraction, operation extraction, and recommendation as composable tools coordinated over a persistent session state.

Prior work leaves a gap for agentic data discovery over tables, a core data structure across scientific and societal domains. Table-search methods retrieve candidates without modeling the user's evolving task, while operation recommenders often assume that the right data has already been found. AIDEN connects retrieval, task-signal extraction, recommendation, and session memory in one loop, making table discovery an interactive agentic process rather than a one-shot search task.

3 Problem Statement

We formalize table discovery as an iterative, intention-aware recommendation process performed by an LLM-driven agent.

Let $\mathcal{D} = (\mathcal{T}, \mathcal{M})$ denote the data lake, where $\mathcal{T}$ is the set of tables and $\mathcal{M}$ their metadata. Let Ω be a finite operation space whose elements are defined in Section 4.4.

Let $r_t \in \mathcal{R}$ denote the user's request at step t. We let the request space admit natural-language utterances, query tables, or both: $\mathcal{R} = \mathcal{L} \cup \mathcal{Q} \cup (\mathcal{L} \times \mathcal{Q})$, where $\mathcal{L}$ is the space of natural-language strings, and $\mathcal{Q}$ is the space of query-table inputs. Let s_t denote the session state, which records previous selections, accepted or rejected recommendations, and executed operations.

At each step, an LLM extractor maps the request and state into three complementary signals:

$$(r_t, s_t) \xrightarrow{\text{Extract}} (q_t, i_t, o_t), \tag{1}$$

where q_t is the query signal (the relevance signal expressed by the request), $i_t \in \mathcal{I}$ is the inferred analytical intention drawn from a closed taxonomy $\mathcal{I}$ (Section 4.3), and $o_t \in \Omega$ is the predicted next operation implied by the request and history. Retrieval over $\mathcal{D}$ produces a candidate pool conditioned on the query signal, $C_t = \text{Ret}(q_t, \mathcal{D}) = \{(T_j, \sigma_j)\}_{j=1}^{m}$, where $T_j \in \mathcal{T}$ and σ_j is the retriever score. The intention layer translates i_t into a structured task specification $\mathcal{U}_t = \phi(i_t)$. The recommender returns a ranked top-k table list and a ranked top-k operation list:

$$\text{Rec}(o_t, s_t, C_t, \mathcal{U}_t, \Omega) \to (\mathbf{T}_t^k, \mathbf{O}_t^k). \tag{2}$$

After observing $(\mathbf{T}_t^k, \mathbf{O}_t^k)$, the user produces feedback f_t and the session updates as $s_{t+1} = \text{Upd}(s_t, r_t, \mathbf{T}_t^k, \mathbf{O}_t^k, f_t)$. The loop repeats until the user's task is complete.

Problem 1. *Given a data lake $\mathcal{D}$, an intention space $\mathcal{I}$, an operation space Ω, a user request r_t, and a session state s_t, an agent must return two ranked lists: candidate tables $\mathbf{T}_t^k$ and next-step operations $\mathbf{O}_t^k$. The tables should be relevant to the request and aligned with the inferred intention, while the operations should be appropriate for the current session state and extracted operation signal.*

4 AIDEN

Figure 1 sketches the AIDEN architecture. The agent integrates a signal extractor, intention layer, retriever, and recommender, around a persistent session-state memory, following the memory,

planning, and tool-use pattern established in the agentic-AI literature [1, 16, 21]. Table retrieval is well established in the literature. In contrast, signal extraction, intention modeling, and operation recommendation remain less established in data-lake search. We instantiate them with LLM prompts (Section A) as a first step toward studying their role in an end-to-end agentic architecture.

4.1 Signal Extractor

At each turn, AIDEN derives three signals from the request and session state. The query signal q_t captures the table-retrieval content of the request. In our experiments, requests are constructed as table retrieval tasks, so q_t is taken directly from the user request and, when available, the query table is passed to the retriever. The remaining two signals are extracted with task-specific LLM prompts: an intention prompt returns a label $i_t \in \mathcal{I}$ (Section 4.3), and an operation prompt returns a label $o_t \in \Omega$ (Section 4.4). Each prompt receives the last w turns of session state so that the extracted signals reflect the dialogue history, not only the latest request.

4.2 Retriever

Retrieval over $\mathcal{D}$ can use learned table embeddings [5, 9] stored in an HNSW similarity index [18, 19] or lexical retrieval [15, 17]. Given the query signal q_t, the retriever returns the top-k tables with their retrieval scores, forming the scored candidate pool C_t defined in Section 3. This pool is then re-ranked by the recommender.

4.3 Intention Layer

The intention layer converts a label i_t drawn from a defined taxonomy, $\mathcal{I} = \{\text{Exploration, Prediction, Integration, Summarization}\}$, into a structured task specification $\mathcal{U}_t$ that the recommender can use. We adopt a three-dimensional specification with a *granularity* dimension (aggregate vs. instance-level), a *richness* dimension (few columns vs. many), and a *compatibility* (required vs. optional). The intention-to-specification mapping is given in Table 1. We score how well a candidate table T matches the task specification $\mathcal{U}_t$:

$$\mathsf{Int}(T, \mathcal{U}_t) = \frac{1}{|\mathcal{A}|} \sum_{\rho \in \mathcal{A}} \mathbb{1}\left[\mathsf{match}_\rho(T, \mathcal{U}_t)\right]. \tag{3}$$

where $\mathcal{A} = \{\text{gran, rich, compat}\}$. Here, match_ρ denotes a deterministic check: row count and aggregate-related column names for granularity, column count for richness, and key overlap or schema/type similarity when compatibility is required. When compatibility is optional, the compatibility check is treated as satisfied. Realizing match_ρ as robust, scalable detectors at data-lake scale, and validating which intention dimensions matter most for downstream retrieval and recommendation quality, are open research questions [12]. In this initial work, we implement the intention layer with an LLM prompt that maps i_t and the available session context s_t to $\mathcal{U}_t$. Learning or tuning the scoring functions in Equation 3, e.g., model-based implementations, is left to future work.

4.4 Recommender

The recommender is the decision layer of AIDEN. It operates over a defined operation space $\Omega = \{\mathsf{Filter}, \mathsf{Join}, \mathsf{Union}, \mathsf{Aggregate}, \mathsf{Clarify}\}$ (Table 2). Given the candidate pool C_t, the query signal q_t, the task specification $\mathcal{U}_t$, the operation signal o_t, and the session state s_t, the recommender produces a ranked top-k list of tables $\mathbf{T}_t^k$ and a ranked top-k list of next-step operations $\mathbf{O}_t^k$. A target utility for scoring a table-operation pair is:

$$U_t(T, O) = \alpha \cdot \mathsf{Rel}(T, q_t) + \beta \cdot \mathsf{Int}(T, \mathcal{U}_t) + \gamma \cdot \mathsf{Opt}(O, o_t, s_t), \tag{4}$$

where Rel is the retriever similarity score, Int is the intention-alignment score of Equation 3, and Opt measures whether operation O is appropriate under the current operation signal and session state. For this work, the recommender is realized as an LLM prompt that consumes $(q_t, C_t, \mathcal{U}_t, o_t, s_t)$ and returns rankings directly. In the current experiments, we evaluate the recommender only as a table reranker; operation prediction is evaluated separately as part of signal extraction. This keeps the initial evaluation focused while preserving the full architecture in which the recommender can produce both table and operation rankings.

Table 1: Intention-to-specification mapping.

Intention	Granularity	Richness	Compatibility
Exploration	either	few	optional
Prediction	instance	many	optional
Integration	either	many	required
Summarization	aggregate	few	optional

Table 2: Operation space used in AIDEN.

Operation	Definition
Filter	Select rows or columns that satisfy user constraints.
Join	Enrich a working table using related entities or shared keys.
Union	Append compatible tables to increase coverage.
Aggregate	Summarize data using aggregation operations.
Clarify	Ask for more information when the next operation is unclear.

4.5 Session State as Agent Memory

The session state s_t is a structured JSON object that records the sequence of prior turns, retrieved candidate pools, accepted and rejected recommendations, and executed operations. The signal extractor, intention layer, and recommender all receive the last w turns of session state in their prompts, which allows AIDEN to track intention drift across turns. State is serialized as a single string appended to each LLM prompt, keeping AIDEN stateless at the service level and making sessions resumable.

5 Evaluation

Benchmark Construction. We evaluate AIDEN on a data lake constructed from the LakeBench WebTable corpus [3]. We selected 100 query tables, with 50 join queries and 50 union queries, prioritizing queries with sufficient ground-truth coverage. We build the lake from two groups of tables: 593 gold tables from LakeBench and 5,000 unrelated distractor tables. This gives 5,593 searchable tables in total. The query tables are not included in the searchable lake. Table 3 summarizes the benchmark size. For evaluation, we label pairs between each query table and lake tables. This gives 1,697 positive pairs. We also identify 2,000 hard-negative pairs from

Table 3: Benchmarks used in the experiments

Tables	# Tables	# Cols	# Rows
Data Lake	5593	38696	256899
Join Query	50	539	4793
Union Query	50	575	3691

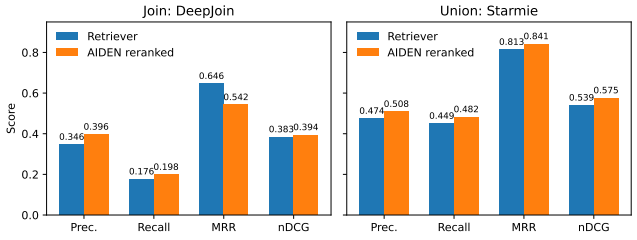

Figure 2: Effect of AIDEN reranking on retriever outputs at $k = 10$. The figure compares Precision, Recall, MRR, and nDCG for the original retriever and AIDEN reranking on DeepJoin for join queries and Starmie for union queries.

the same lake tables: tables with value overlap for join queries, and tables with similar schema and types for union queries.

Interaction Data and Annotation. For each query table, we created a first-turn analyst request from the table schema and row preview. The requests were then manually validated and edited to be human-like and to cover the operation space Ω and the intention space $\mathcal{I}$. Each example includes a query table, a user request, validated operation and intention labels, and gold retrieval tables when external tables are needed. The labels are used only for evaluation and for controlled recommender experiments.

Experimental Setup. We evaluate three parts of AIDEN. First, we evaluate the retriever using BM25 and TF-IDF over the same table serialization, DeepJoin for join retrieval, and Starmie-style embeddings for union retrieval [4, 5, 15, 17]. Second, we evaluate the recommender by asking a local LLM (`Gemma-4-31B-IT-NVFP4`) to rerank the top 30 retrieved candidates without seeing gold labels. The prompt receives the user request, the query table preview, candidate table previews, the operation signal, and the intention-derived task specification. Third, we evaluate signal extraction by asking the same LLM to predict the operation and intention from either the request alone or the request plus query-table preview. We scope our evaluation to the first-turn setting, leaving feedback-based updates to future work. Retrieval and reranking are evaluated with precision, recall, MRR, MAP, and nDCG at $k = 10$. Signal extraction is evaluated with macro-F1.

Reranking Results. The recommender improves retriever outputs (Fig. 2). For join search, reranking DeepJoin improves precision@10 from 0.346 to 0.396, recall@10 from 0.176 to 0.198, and nDCG@10 from 0.383 to 0.394. For union search, reranking Starmie improves precision@10 from 0.474 to 0.508, recall@10 from 0.449 to 0.482, and nDCG@10 from 0.539 to 0.575. This supports the role of AIDEN as a task-aware refinement layer over an initial candidate pool. MRR improves for union from 0.813 to 0.841, but drops for join from 0.646 to 0.542. This is likely due to a granularity mismatch: join relevance is derived from column-level matches, while the recommender reasons over table-level previews. As a result, reranking

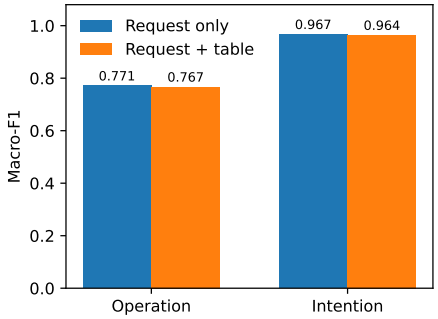

Figure 3: Signal extraction performance for operation and intention labels. The figure compares request-only input with request plus query-table input. Intention extraction remains strong in both settings, while query-table context gives little change in this first-turn setting.

can improve aggregate top-k quality while moving a rank-1 join-bearing table lower, which strongly affects MRR. Finally, reranking does not uniformly improve strong lexical baselines, especially TF-IDF for join search, suggesting that the prompt-based recommender complements rather than replaces a strong retriever.

Signal Extraction Results. The signal extractor performs well on the manually validated interaction data (Fig. 3). For intention prediction, it achieves 0.967 macro-F1 with the request only and 0.964 when the query table is added. For operation prediction, it achieves 0.771 macro-F1 with the request only and 0.767 with the query table. These results show that the request text already carries most of the signal needed to infer the operation and intention. Adding the query table gives little change in this first-turn setting.

These experiments validate AIDEN's pipeline: retrieval builds the candidate pool, reranking refines it, and signal extraction provides the operation and intention labels needed by the recommender.

Scalability. Using our LakeBench-derived evaluation benchmark, AIDEN keeps its LLM stages bounded: reranking operates on a fixed top-k pool, and signal extraction depends only on the request and session context. Thus, scaling mainly depends on retrieval and indexing [20], while k controls the quality/latency trade-off.

Availability. The code and benchmark construction scripts are available at https://github.com/IbraheemTaha/AIDEN.

6 Conclusion

We presented AIDEN, an agentic framework for intention-aware table discovery in data lakes. AIDEN extracts query, intention, and operation signals, retrieves candidate tables, and uses an LLM-based recommender to rerank them for the current task. On a benchmark constructed from LakeBench, AIDEN improves specialized retriever outputs and extracts intention and operation labels reliably. Future work will extend the evaluation to multi-turn feedback and replace prompt-based components with learned or tuned models.

Acknowledgments

This work has received funding from the EU's Horizon 2020 MSCA-ITN programme under grant agreement No. 955895. I thank Enas Khwaileh for an initial discussion related to this work.

References

[1] Xu Chen, Fangning Ren, Dazhou Yu, Lechen Dong, and Fang Liu. 2025. AutoSolvate-Agent: An Autonomous Agent for GPU-accelerated Solution-phase Quantum Chemistry. In *Large Language Models for Scientific and Societal Advances*. Toronto, ON, Canada. https://openreview.net/forum?id=ATEtqdGn6W

[2] Bhavya Chopra, Anna Fariha, Sumit Gulwani, Austin Z. Henley, Daniel Perelman, Mohammad Raza, Sherry Shi, Danny Simmons, and Ashish Tiwari. 2023. CoWrangler: Recommender System for Data-Wrangling Scripts. In *Companion of the 2023 International Conference on Management of Data* (Seattle, WA, USA) *(SIGMOD '23)*. Association for Computing Machinery, New York, NY, USA, 147–150. doi:10.1145/3555041.3589722

[3] Yuhao Deng, Chengliang Chai, Lei Cao, Qin Yuan, Siyuan Chen, Yanrui Yu, Zhaoze Sun, Junyi Wang, Jiajun Li, Ziqi Cao, Kaisen Jin, Chi Zhang, Yuqing Jiang, Yuanfang Zhang, Yuping Wang, Ye Yuan, Guoren Wang, and Nan Tang. 2024. LakeBench: A Benchmark for Discovering Joinable and Unionable Tables in Data Lakes. *Proc. VLDB Endow.* 17, 8 (April 2024), 1925–1938. doi:10.14778/3659437.3659448

[4] Yuyang Dong, Chuan Xiao, Takuma Nozawa, Masafumi Enomoto, and Masafumi Oyamada. 2023. DeepJoin: Joinable Table Discovery with Pre-Trained Language Models. *PVLDB* 16, 10 (jun 2023), 2458–2470. doi:10.14778/3603581.3603587

[5] Grace Fan, Jin Wang, Yuliang Li, Dan Zhang, and Renée J. Miller. 2023. Semantics-aware Dataset Discovery from Data Lakes with Contextualized Column-based Representation Learning. *Proceedings of the VLDB Endowment* 16, 6 (2023), 1726–1739. doi:10.14778/3587136.3587146

[6] Raul Castro Fernandez, Ziawasch Abedjan, Favour Koko, Guoliang Yuan, Samuel Madden, and Michael Stonebraker. 2018. Aurum: A Data Discovery System. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*. IEEE, Paris, France, 1001–1012. doi:10.1109/ICDE.2018.00094

[7] Yihao Hu, Jin Wang, and Sajjadur Rahman. 2025. LakeVisage: Towards Scalable, Flexible and Interactive Visualization Recommendation for Data Discovery over Data Lakes. *Proc. VLDB Endow.* 18, 10 (June 2025), 3545–3558. doi:10.14778/3748191.3748214

[8] Aamod Khatiwada, Grace Fan, Roee Shraga, Zixuan Chen, Wolfgang Gatterbauer, Renée J. Miller, and Mirek Riedewald. 2023. SANTOS: Relationship-Based Semantic Table Union Search. *Proc. ACM Manag. Data* 1, 1, Article 9 (may 2023), 25 pages. doi:10.1145/3588689

[9] Aamod Khatiwada, Harsha Kokel, Ibrahim Abdelaziz, Subhajit Chaudhury, Julian Dolby, Oktie Hassanzadeh, Zhenhan Huang, Tejaswini Pedapati, Horst Samulowitz, and Kavitha Srinivas. 2025. TabSketchFM: Sketch-Based Tabular Representation Learning for Data Discovery Over Data Lakes . In *2025 IEEE 41st International Conference on Data Engineering (ICDE)*. IEEE Computer Society, Los Alamitos, CA, USA, 1523–1536. doi:10.1109/ICDE65448.2025.00118

[10] Eugenie Y. Lai, Yeye He, and Surajit Chaudhuri. 2025. Auto-Prep: Holistic Prediction of Data Preparation Steps for Self-Service Business Intelligence. *Proc. VLDB Endow.* 18, 7 (March 2025), 2212–2225. doi:10.14778/3734839.3734856

[11] Pei-Chi Lo and Yi Lu. 2025. Dissecting Judicial Reasoning in U.S. Copyright Damage Awards: A Discourse-Based LLM Analysis. In *Large Language Models for Scientific and Societal Advances*. Toronto, ON, Canada. https://openreview.net/forum?id=nV9GAbXmUm

[12] Yuyu Luo, Guoliang Li, Ju Fan, and Nan Tang. 2026. Data Agents: Levels, State of the Art, and Open Problems. SIGMOD 2026 Tutorial. arXiv:2602.04261 [cs.DB] https://arxiv.org/abs/2602.04261

[13] Fengran Mo, Kelong Mao, Ziliang Zhao, Hongjin Qian, Haonan Chen, Yiruo Cheng, Xiaoxi Li, Yutao Zhu, Zhicheng Dou, and Jian-Yun Nie. 2025. A Survey of Conversational Search. *ACM Trans. Inf. Syst.* 43, 6, Article 167 (Sept. 2025), 50 pages. doi:10.1145/3759453

[14] Fatemeh Nargesian, Erkang Zhu, Ken Q. Pu, and Renée J. Miller. 2018. Table Union Search on Open Data. *PVLDB* 11, 7 (mar 2018), 813–825. doi:10.14778/3192965.3192973

[15] Stephen Robertson and Hugo Zaragoza. 2009. The Probabilistic Relevance Framework: BM25 and Beyond. *Found. Trends Inf. Retr.* 3, 4 (April 2009), 333–389. doi:10.1561/1500000019

[16] Divya Sharma, Ghazal Azarfar, Bima J. Hasjim, and Mamatha Bhat. 2025. When LLMs Decide Who Gets Care: A Vision for Multi-Agent Systems in High Stakes Clinical Decision-Making. In *Large Language Models for Scientific and Societal Advances*. https://openreview.net/forum?id=fwgjebAQMa

[17] Karen Sparck Jones. 1988. *A statistical interpretation of term specificity and its application in retrieval*. Taylor Graham Publishing, GBR, 132–142.

[18] Ibraheem Taha, Matteo Lissandrini, Alkis Simitsis, and Yannis Ioannidis. 2024. A Study on Efficient Indexing for Table Search in Data Lakes. In *2024 IEEE 18th International Conference on Semantic Computing (ICSC)*. IEEE, Laguna Hills, CA, USA, 245–252. doi:10.1109/ICSC59802.2024.00046

[19] Ibraheem Taha, Matteo Lissandrini, Alkis Simitsis, and Yannis Ioannidis. 2025. Comparative Analysis of Indexing Techniques for Table Search in Data Lakes. *International Journal of Semantic Computing* 19, 02 (2025), 173–196. doi:10.1142/S1793351X25420024

[20] Ibraheem Taha, Matteo Lissandrini, Alkis Simitsis, Torben Bach Pedersen, and Yannis Ioannidis. 2026. *Table Search in Data Lakes: Methods, Indexing Techniques, and Research Challenges*. Springer, Germany, Chapter 10.

[21] Zhengbang Yang, Junyuan Hong, Yijiang Pang, Jiayu Zhou, and Zhuangdi Zhu. 2025. ChatWise: A Strategy-Guided Chatbot for Enhancing Cognitive Support in Older Adults. In *Large Language Models for Scientific and Societal Advances*. https://openreview.net/forum?id=IiX9UFCZc8

A LLM Prompts Used in AIDEN

A.1 Reranker Prompt

```
- You are the AIDEN recommender.
- Your job is to rerank candidate tables for a  data-lake search task.
- Use only the provided request, query table, candidate table
previews, operation signal, and task specification.
- Never invent table names.
- Return valid JSON only.
USER = f""" AIDEN recommender reranking task
Query id: {query_id}, Task type: {task_type}, User request:
{user_request}, Operation signal: {operation}, Intention label:
{intention}, Task specification: {json.dumps(task_spec,
ensure_ascii=False)}
Query table: {query_table} Query table preview:
{query_preview} Candidate tables: {candidates_text}
Rules:
1. Rerank the candidate tables from most useful to least useful.
2. Only return table names that appear in the candidate list.
3. For Join, prefer tables that can enrich the query table through
shared keys, entities, or related identifiers.
4. For Union, prefer tables that can extend the query table with
useful records through compatible entities, observations,  or
attributes.
5. Return JSON only in exactly this format: {{ "reranked_tables": [
"table_a.csv", "table_b.csv"] }} """
```

A.2 Signal Extractor Prompt

```
You are the AIDEN signal extractor. For each user request, output two
labels:
- operation: the single primary data transformation needed, or
clarify if the request is too vague to choose among the others.
- intention: the user's high-level goal.

Choose exactly one operation and exactly one intention from the
allowed labels below.
Operations are mutually exclusive: pick the one that produces the
user's final result.

OPERATIONS:
- Filter: Select rows or columns from a single table using stated
conditions or constraints.
- Join: Enrich the working table by linking another table through
shared keys or related entities. Adds columns.
- Union: Append rows from one or more compatible tables to extend
coverage across sources, time periods, or partitions.
- Aggregate: Compute counts, sums, averages, group-bys,
distributions, or other summary statistics over rows, usually these
are applied on the same table.
- Clarify: Use ONLY when the request is genuinely too vague or
underspecified to choose one of the four operations above.

INTENTIONS:
- Exploration: User wants to inspect, browse, look up, or understand
the data. No downstream task is mentioned.
- Prediction: User wants instance-level rows or features for a
machine-learning task (training examples, features, labels, scoring
inputs, model inputs).
- Integration: User wants to combine, enrich, append, or connect data
from multiple tables or sources.
- Summarization: User wants aggregate insights such as counts,
totals, averages, group breakdowns, or distributions.

Output a single JSON object with exactly two keys: "operation" and
"intention". No prose, no code fences, no explanation.
```