AuDQ: Auditable Data Quality through Agentic Rule Discovery

Jacopo Biggiogera[1] Nicolas Colombo[1] Krati Saxena[2]
Bhaskarjit Sarmah[2] Stefano Pasquali[3]
[1]Domyn, Milan, Italy [2]Domyn, Gurugram, India [3]Domyn, New York City, USA
{jacopo.biggiogera, nicolas.colombo, krati.saxena}@domyn.com
{bhaskarjit.sarmah, stefano.pasquali}@domyn.com

Abstract

High-stakes tabular pipelines rely on scattered, static quality checks that are hard to evolve and audit against policy intent. Regulated domains particularly suffer as the LLM-era automation raises expectations for traceability and accountability. We propose a vision for an *ontology-grounded, repository-native* data quality layer in which rules are first-class artifacts: versioned, provenance-linked to governance texts, and re-playable through immutable execution traces. As a first step toward validating this vision, we present a concrete implementation of its agentic rule-discovery core with two rule acquisition modes (Full-Agentic and Guided) and a human-in-the-loop (HITL) review interface. We define an evaluation framework and run experiments addressing whether our fully autonomous, divide-and-conquer agentic rule discovery produces rules of the same quality and diversity as those specified by a human expert, reporting preliminary results on two real world industry datasets in the finance and retail domains.

1 Introduction

Tabular data drives high-stakes decisions, yet most organizations still rely on basic data quality (DQ) checks covering missingness, uniqueness, and simple range constraints. Recent enforcement actions underscore the cost of undetected failures: a USD $100M settlement tied to AML compliance gaps (BakerHostetler) and a USD $136M fine linked to data-management weaknesses (Michelle Price and Bautzer). The hardest production failures arise from context-dependent constraints tied to domain semantics (limits that depend on market hours, settlement cycles, or cross-entity reconciliation) which are documented in governance policies but never fully operationalized, remaining handcrafted, scattered, and prone to drift as schemas evolve.

We propose a vision for an *ontology-grounded, repository-native agentic DQ layer*: a system that automatically discovers rules from relational data, grounds them in semantic ontologies and governance artifacts, governs them through a human-in-the-loop lifecycle, and compiles approved rules into deterministic executable checks. Central to this vision is the idea that DQ rules should be treated as *living rulebooks*: first-class, versioned artifacts with clear provenance, not scattered code buried in pipelines.

As a concrete step towards this vision, we have implemented the agentic rule-discovery core of the proposed system and designed an evaluation to answer the following research questions:

1. Are rules discovered fully autonomously by the agentic system capable of approaching the quality ceiling set by those specified by a human?
2. Are rules discovered fully autonomously by the agentic system as diverse and non-redundant as those specified by a human?

In doing so, this paper makes three contributions:

- **An agentic architecture for automatic rule discovery in relational database systems** .
- **An empirical evaluation of rule-discovery quality and diversity**.
- **A comprehensive vision for a full data-quality layer governing the complete rule lifecycle**.

We use two real industry derived datasets from the financial banking and retail fields to test the domain-agnostic design of the system.

2 Related Work

Our work draws on three threads: data agents, DQ tooling, and ontology/audit infrastructure, but focuses on what they do not yet standardize: a shared, auditable representation of *DQ rules themselves* and the automated discovery of such rules from data.

Agentic data analytics. Recent data agents plan

EMNLP 2026 Industry Track, pages 1925–1943
October 24-29, 2026 ©2026 Association for Computational Linguistics

multi-step analyses and produce end-to-end outputs from heterogeneous data (Sun et al., 2025; Lapin et al., 2025; Qiao et al., 2025). Evaluation suites measure task success across analytics settings (Nie et al., 2026; Wang et al., 2025; Padala et al., 2025). None treat DQ as a lifecycle problem in which rules are created, versioned, executed repeatedly, and updated under audit, nor do they evaluate the quality and diversity of automatically discovered constraints.

Data-quality tooling. DQ stacks author and run checks and report violations (Pancini et al., 2025; Machireddy, 2023; Patel et al., 2022). Learned approaches detect or repair errors and explain anomalies (Dong et al., 2025; Sattarov et al., 2022). Formal constraint languages (denial constraints, SHACL/SHEx (Thornton et al., 2019)) offer expressive validators, but checks remain scattered in code with weak links to the policy text that motivated them and no replayable execution evidence for audit. Crucially, none of these approaches address the *automated discovery* of domain-grounded rules from relational data at scale.

Ontologies, audit, and regulated-industry architectures. Auditable LLM pipelines argue for traceable, replayable decision-making (Xia et al., 2025; Ojewale et al., 2026). Ontology-guided methods improve grounding and validation (Guo et al., 2025; Chepurova et al., 2025; Feng et al., 2025). Regulated-industry proposals motivate agentic DQ (Tomar et al., 2024; Durgam, 2025), but do not define a concrete, testable contract linking ontologies, rules, and execution traces, nor an evaluation of discovery quality.

We synthesize these threads by making rules and their evidence the central auditable artifacts, introducing an agentic discovery architecture, and providing an empirical evaluation of discovery quality and diversity.

3 Problem Setting

Organizations rely on tabular data (single tables, streaming logs, and multi-table warehouses) to power downstream reporting and high-stakes decisions. In finance, for example, small data errors can become material incidents: misreported exposures, missed AML signals, and model drift. End-to-end traceability is essential, requiring clear links from policy statements to implemented DQ checks.

Current practice and limitations. Most production DQ remains a patchwork of simple checks spread across code, dashboards, and tools (Saini et al., 2025), with weak links to the semantic, temporal, and cross-table requirements in Table 1 and limited governance (versioning, provenance, auditability). Beyond basic checks, policy-grounded semantic rules are costly and hard to identify and operationalize, lacking a principled automated mechanism to discover and formalize them. These gaps motivate both our architecture and our research questions.

Requirement	What it enforces / example
Syntactic validity	Types, ranges, formats across flat files and joins; referential integrity via keys/views.
Semantic correctness	Ontology-grounded invariants; e.g., notional > 0 consistent with product type in finance via trade $\rightarrow$ instrument $\rightarrow$ client, or a medication dose within an age-adjusted safe range in healthcare.
Temporal consistency	No future-dated events relative to ingestion; consistent time-series snapshots across grains.
Cross-table reconciliation	Aggregates align with primaries; e.g., ledger totals vs. front-office trades, or risk vs. accounting views.

Table 1: Core data-quality requirements for analytics-aware rules.

4 Implemented System

4.1 Overview and design principles

Figure 1 shows the four-stage pipeline we have implemented. Data flows from a **Statistical Profiler** through an **Ontology Generator** and a **Rule Discovery/Definition** module to a **Rule Implementer**. Every stage reads from and writes to a central **Artifacts Repository**, which stores versioned semantic profiles, ontology nodes, natural language and implemented rules alongside orchestrator plans, subagent rule proposals, thoughts/tool calls made by every agent and recorded HITL decisions. A `LEARNINGS.MD` file is updated by a dedicated critic agent after each run, appending timestamped entries under sections for patterns to avoid, under-explored scopes, failure analyses, coverage notes, and heuristics that worked. This persistent memory informs subsequent rule-generation episodes. All coordination is repository-mediated: components write typed artifacts to the central Repository and read only from it, with no direct agent-to-agent message passing. This decoupling keeps lineage and audit straightforward and makes agents independently replaceable. Concrete artifact schema are given in Appendix due to page limits.

Three design principles govern the implementation. *Determinism before reasoning*: the profiler is LLM-free and produces the same output for the same input, providing a stable epistemic foundation. *Text-first rules*: rules are stored in natural language and reviewed before compilation, making them auditable also by domain experts without programming skills. *Repository-mediated coordination*: all component interaction is mediated through versioned artifacts, keeping lineage and audit straightforward.

4.2 Statistical Profiler

The profiler connects directly to the database and produces a *Statistical Evidence Pack*: a structured artifact containing distributional summaries, null rates, cardinality estimates, temporal patterns, candidate keys, and inter-column dependencies for every table and column in scope. Being LLM-free, it is fully deterministic; downstream agents reason over its outputs rather than raw rows, avoiding sampling bias and token-budget issues on large tables.

4.3 Ontology Generator

For each table, entity, and column, a focused LLM call receives only the relevant portion of the Statistical Evidence Pack. A *batched, asynchronous* strategy issues N parallel calls, reducing wall-clock time proportionally to the number of objects. Three harness properties are enforced: *Context isolation*, each call sees only its target object's statistics; *Lifecycle hooks* run a duplicate-check function after each node proposal; if the node already exists in the repository, the LLM call is suppressed. *Evaluation* exposes proposed nodes to a human reviewer who can modify or delete them before they are committed. Two usage modes are supported: *automated generation*, which produces one node per table, entity, and column plus the three most salient vocabulary and within-table relationship nodes; and *manual node* creation, where the user describes the desired node, selects a type and column association, and the agent either creates a new node or interprets the request as a modification of an existing one.

4.4 Rule Discovery and Definition

Rule discovery is the architectural core of the system and the focus of our evaluation.

Full agentic mode. An *orchestrator* receives the full profiled database and segments the column space into N disjoint *subspaces*, each assigned to a *researcher subagent*. Each subagent explores its scope autonomously and proposes rules until a *coverage criterion* is met: every column in the subspace must be associated with at least one rule. The orchestrator has access to lookup tools for table/column listings, profile summaries and ontology nodes. Finally, planning tools help it draft/modify a plan file to divide the search space in meaningful sub spaces and spawn workers; researcher subagents additionally have column-level profile queries, ontology lookups, a vector-DB search over business documents (governance grounding), and a rule-proposal tool that saves decisions to the artifact repository. During proposal each rule has the hard constraint of having to be linked to at least one ontology node in order for it to be considered valid.

The harness manages long-running execution through *context compaction* (token-budget-triggered summarization with smart reinitialisation injecting uncovered columns and existing rules), *coverage enforcement* (a lifecycle hook terminates the loop when the criterion is met), and *persistent state* (all outputs written to storage for run recovery from partial states).

Guided mode. A domain expert describes the rules they already know should exist, optionally specifying the relevant columns and tables, and the system serializes the rule structure and generates code. If no subset is specified, the orchestrator finds one subspace containing the relevant tables/columns and spawns a subagent to generate the described rule. In the following experiments this mode serves as our human-expert baseline. The agent played strictly the role of *serialization*: converting a human-identified constraint into the same schema and logic-block structure used in Full-Agentic mode, so that both conditions yielded structurally comparable artefacts. The rule mining act itself (exploring the dataset and identifying a candidate constraint together with the tables and columns it covers) was performed entirely by the human actor.

4.5 Rule Implementer and HITL Governance

For each NLP rule selected for implementation, a single LLM call receives the rule text, its associated ontology nodes, and the relevant portions of the statistical profile. As with the ontology generator, a batched, asynchronous strategy is used and the system then translates each into executable SQL or Python code. A HITL step allows the user to view the NLP rule and the generated code, mod-

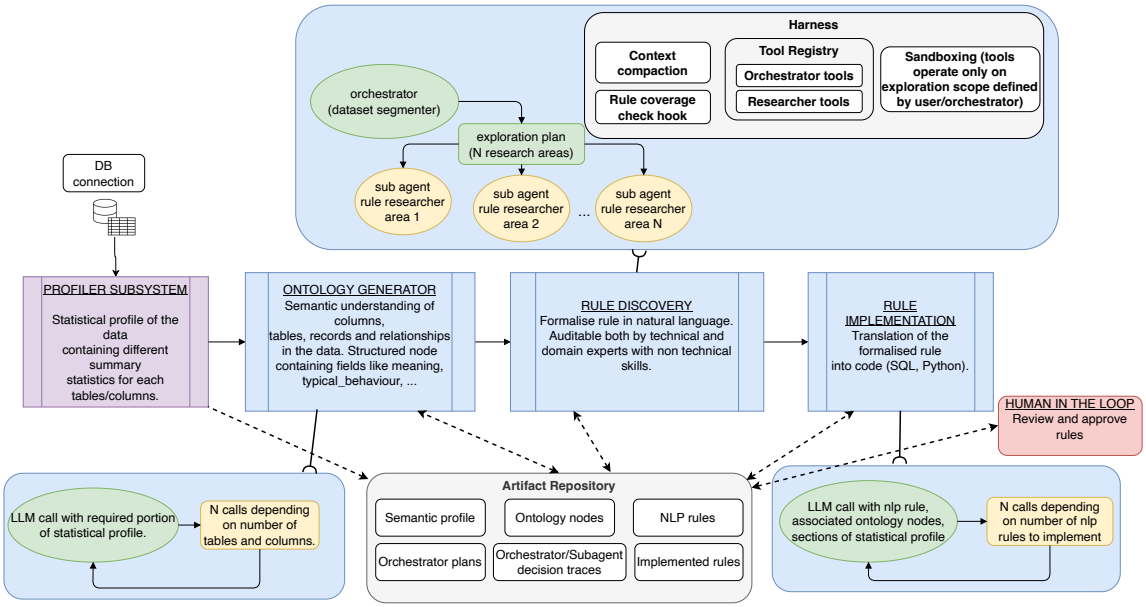

Figure 1: Implemented system overview.

ify the rule logic if needed, and re-trigger code generation. Rules move through three lifecycle states: *Proposed*, *Approved*, *Rejected*, with every transition recorded in the artifact repository.

5 Evaluation

A central capability for an agentic rule-discovery system is whether autonomous discovery can approach the quality and diversity ceiling set by human specified rules, thus making this the key first step for evaluation when developing said systems. This is non-trivial: the full-agentic mode must be able to cover the entire column space and produce semantically varied rules. The divide-and-conquer strategy previously described in section §4.4 introduces a specific risk: *cross-agent leakage*: researcher subagents operating on adjacent subspaces may independently converge on semantically equivalent rules encoding the same constraint with different syntax. Our evaluation is designed to quantify both failure modes.

We compare *Full Agentic* mode and *Guided* mode, with the latter being a structural upper bound on both dimensions: quality (the human specifies intent directly) and diversity (user specifies rules manually one at a time to prevent redundancy and contradictions).

5.1 Metrics

Good-rule rate. For dataset D and mode $M \in [A(automated), G(guided)]$, let N_M be the number of rules reviewed, A_M accepted as-is, U_M accepted with modifications, and R_M rejected. We define:

$$\hat{p}^{M}_{\text{good}} = \frac{A_M + \alpha\, U_M}{N_M}, \quad \alpha = 0.5,$$

where modified-but-approved rules receive partial credit through the $\alpha = 0.5$ weight.

Information Divergence Score. Following from (Lee et al., 2025) we define an IDS measure adapted to our own domain to evaluate pairwise diversity across N proposed rules:

$$\text{IDS} = 1 - \frac{1}{\binom{N}{2}} \sum_{i<j} \text{sim}_{ij}$$

where:

$$\text{sim}_{ij} = w_{\text{syn}}\, \text{Jac}(t_i, t_j) + w_{\text{sem}}\, \cos(e_i, e_j) + w_{\text{scope}}\, \text{Jac}(s_i, s_j)$$

Pairwise similarity thus combines three components,

1. **Syntactic**: token-level Jaccard similarity over rule logic and query text, $\text{Jac}(t_i, t_j)$.
2. **Semantic**: dense-embedding cosine similarity over rule logic and query text, $\cos(e_i, e_j)$.
3. **Rule scope**: Jaccard similarity over the union of ontology nodes and table/column names referenced by each rule, $\text{Jac}(s_i, s_j)$.

IDS $\in [0, 1]$ with a higher score signifying more diverse rules.

5.2 Datasets

We evaluate the system on two publicly available industry relational databases that capture real diversities in domain, schema and complexity.

Berka (retail banking). The Berka dataset contains real, anonymized data from a Czech bank covering the period 1993–1998. It comprises 8 tables (`account`, `client`, `disp`, `card`, `loan`, `order`, `trans`, and `district`) linked by foreign keys that model the full lifecycle of retail banking. The database contains 5,369 clients and over 1,000,000 transaction records, making it a rich, multi-table relational benchmark with realistic referential integrity constraints, coded categorical fields, and cross-table financial consistency invariants. It is widely used in the relational data mining literature (Li et al., 2023; Singh et al., 2025).

rel-HM (fashion retail). The rel-HM dataset contains real, anonymized data from retail purchases of HM products. It contains 3 tables (`customers`, `articles`, and `transactions`) with approximately 16.7 million purchases from 2018 to 2020. Despite its simpler schema, it presents high-cardinality categorical vocabularies, implicit referential constraints between articles and transactions, and cross-field consistency rules between product metadata fields (Robinson et al., 2024). As both dataset do not ship with any grounding documentation the rule discovery system ran without the governance-grounding search tool, using only the dataset statistics and the inferred ontology.

5.3 Experimental protocol

For each dataset, we ran both the Full Agentic mode and the Guided mode with $k = 4$ reviewers, each yielding between 25-55 proposed rules per run and a total of 16 runs overall. Reviewers were mid/senior data engineering professionals with finance and retail domain experience. Every reviewer authored one Guided run to which a Full agentic run was paired for rule reviewing load purposes. Rule assignment for review followed a blinded, mixed-mode, round-robin batch design: each rule was rated by exactly two reviewers other than its author, every batch mixed rules from both generation modes and reviewers were blind to which mode had produced each rule. We report the full block-construction protocol and validation checks in Appendix §C. All rules were reviewed via the standard HITL flow and labeled accept/modify/reject. All the ontology and rule generation processes utilised Qwen3.5-397B-A17B as the underlying LLM model, chosen for its competitive performances, open source nature and generous context/output limits. A temperature of 0.5 was selected to allow for variability across exploration episodes.

Experiment 1 (RQ1): Rule-generation quality by mode. We test for equivalence, H_0: $|\hat{p}^A_{\text{good}} - \hat{p}^G_{\text{good}}| \geq \Delta$ against H_1: $|\hat{p}^A_{\text{good}} - \hat{p}^G_{\text{good}}| < \Delta$, by estimating the shift in good-rule rate between modes, $\hat{p}^A_{\text{good}} - \hat{p}^G_{\text{good}}$, per dataset and pooled across datasets. Because per-run scores are few and non-normal, we use a non-parametric two one-sided test (TOST): a Hodges–Lehmann (HL) shift estimator, exact confidence intervals obtained by inverting the Mann-Whitney statistic, rank-biserial effect sizes, and the TOST power achieved. The equivalence margin was fixed before the analysis at the standard deviation of Guided-mode run scores ($\Delta = 0.0556$), so that modes count as equivalent only if they differ by less than reviewers differ among themselves. We report the HL shift and its interval as our primary estimate, and the TOST verdict alongside its achieved power. Because every rule carries two independent ratings, we additionally quantify inter-rater agreement on the accept/modify/reject scale. We report raw agreement alongside two chance-corrected coefficients: ordinal Krippendorff's α and Gwet's AC2, both with 95% bootstrap intervals. We report both because they behave differently under skewed marginals, and our verdict distribution is heavily skewed; relying on either alone would misstate the reliability of the labels.

Matched-scope volume control. The two modes propose different rule volumes as the Full Agentic coverage criterion forces at least one rule per column generating roughly 1.6x the rules of the Guided mode (Table 13). Good-rule rate is a precision-style measure, so comparing it across sets of very different size favors the mode that proposes fewer, safer rules. Since neither dataset ships with a reference rule set to compare against we repeat the equivalence test with a matched-scope volume control. We recorded the tables and columns referenced in each rule of all Guided mode runs and took the intersection across all four reviewers (a conservative proxy for the scope every reviewer agreed was rule-worthy). We then restricted the agentic rules set to that same scope. This removes the potentially low-value rules forced by the coverage

Dataset	Mode	$\hat{p}_{\text{good}}$
Berka	Full agentic	0.890
	Guided	0.880
H&M	Full agentic	0.922
	Guided	0.850
Overall	Full agentic	0.906
	Guided	0.865

Table 2: Mean good-rule rate across runs per dataset and mode.

criterion and makes volumes comparable (Agentic:Guided ratio = 1:1 on Berka, 1:1.3 on H&M). Scope matching is applied post hoc, at analysis time rather than at batch construction, so it does not affect what reviewers evaluated. We then recompute the RQ1 analysis on this matched set.

Experiment 2 (RQ2): Diversity and partition effectiveness. We test for equivalence, H_0: $|\text{IDS}_A - \text{IDS}_G| \geq \Delta$ against H_1: $|\text{IDS}_A - \text{IDS}_G| < \Delta$ by estimating the shift in Information Divergence Score between modes, $\text{IDS}_A - \text{IDS}_G$, using the same non-parametric TOST procedure, with the margin fixed a priori at the standard deviation of Guided mode IDS scores ($\Delta = 0.0202$). For each dataset and replicate we use pre-reviewed rules (to avoid human review masking the agentic run's inherent diversity through rule modification) and compute IDS per run. To ensure that any result is not an artefact of a single similarity component, we repeat the analysis across four weight configurations: equal weights $(w_{\text{syn}}, w_{\text{sem}}, w_{\text{scope}}) = (1/3, 1/3, 1/3)$, semantic-heavy $(0.25, 0.5, 0.25)$, syntax-heavy $(0.5, 0.25, 0.25)$, and scope-heavy $(0.25, 0.25, 0.5)$.

5.4 Results

RQ1: Rule-generation quality. Table 2 reports the mean good-rule rate per mode and dataset. The Full agentic mode achieves a higher mean good-rule rate on both datasets (Berka: 0.890 vs. 0.880; H&M: 0.922 vs. 0.850), and the gap is larger on the rel-HM, the dataset with a simpler structure.

Full Agentic mode scored slightly higher than the Guided mode overall (0.037 Hodges–Lehmann shift, 90% CI: -0.008 to $+0.089$), with a small shift for Berka ($+0.004$, 90% CI: -0.112 to $+0.135$) and a larger one for H&M ($+0.064$, 90% CI: $+0.029$ to $+0.129$), where the interval excludes zero: on this dataset the four agentic runs separate completely from the four guided runs ($r = -1$). Table 3 reports the full analysis. Equivalence is not established for any comparison since every

Comparison	HL	90% CI	r	Power
Berka	$+0.004$	$[-0.112, +0.135]$	0	$< 1\%$
H&M	$+0.064$	$[+0.029, +0.129]$	-1	36.5%
Overall	$+0.037$	$[-0.008, +0.089]$	-0.50	27.8%
Matched-scope control				
Overall	$+0.037$	$[-0.027, +0.094]$	-0.38	3.5

Table 3: Non-parametric TOST equivalence analysis for RQ1 good-rule rate (Full Agentic − Guided); equivalence margin $\Delta = 0.0556$. HL: Hodges–Lehmann shift with exact CI from the inverted Mann-Whitney statistic; r: rank-biserial effect size (negative favor Full Agentic mode); Power: achieved TOST power. Runs required for 80% power: 28 (Berka), 6 (H&M), 16 (Overall). The matched-scope: rules with column scope shared by all Guided runs.

CI extends beyond the set margin ($\Delta = 0.0556$). Achieved power is low throughout (27.8% overall), rendering the equivalence verdict uninformative rather than simply unestablished, and thus we draw no conclusion of quality parity between the modes. The matched-scope volume control leaves the estimate essentially unchanged ($+0.037$, 90% CI: -0.027 to $+0.094$), indicating that the volume asymmetry created by the coverage criterion is not what produces the agentic-favouring shift.

Inter-rater agreement. Table 4 reports agreement over the 541 doubly-rated rules. Agreement is high: reviewers selected the same verdict on 77.3% of rules, and Gwet's AC2 is 0.812 overall, with consistent values across both datasets and both modes. Ordinal Krippendorff's α is lower (0.174) because 84.8% of ratings are `accept`; when one category dominates, α's chance-correction term approaches the observed agreement and the coefficient is deflated. AC2 is a metric robust to exactly this situation, hence why both are reported to better understand the caveat around the ratings. Of note is that α is lower on agentic than on guided rules despite near-identical raw agreement (76.8% vs. 78.1%). The gap is largely arithmetic: accept is even more dominant on agentic rules (86.0% vs. 82.6%, Table 13), which raises the agreement expected by chance alone and leaves less credit for α to assign. Appendix §C.3 gives the full breakdown.

RQ2: Rule diversity and partition effectiveness. Table 5 reports mean IDS per mode across four weight configurations. IDS values are nearly identical between modes: under semantic-heavy weights both achieve a mean IDS of 0.648, and the largest absolute difference across configurations is 0.010 (scope-heavy, favouring the agentic mode). The

Subset	n	Exact	α [95% CI]	AC2 [95% CI]
Overall	541	77.3	0.174 [.074,.279]	0.812 [.770,.851]
Berka	317	76.0	0.165 [.035,.292]	0.810 [.754,.858]
H&M	224	79.0	0.189 [.021,.350]	0.815 [.747,.872]
Agentic	340	76.8	0.061 [−.056,.187]	0.816 [.764,.864]
Guided	201	78.1	0.326 [.150,.484]	0.805 [.730,.868]

Table 4: Inter-rater agreement over doubly-rated rules. **Exact**: % rules where both reviewers gave same verdict. α: ordinal Krippendorff's α; AC2: Gwet's AC2; both with 95% bootstrap intervals. α is deflated relative to AC2 because `accept` accounts for 84.8% of all ratings.

Weight config	IDS_A	IDS_G
Equal	0.732	0.729
Sem.-heavy	0.648	0.648
Syn.-heavy	0.749	0.750
Scope-heavy	0.798	0.788

Table 5: Mean IDS across datasets ($n_A = n_G = 8$ runs per mode).

sign of Δ changes across weight configurations, indicating no consistent direction of difference.

Table 6 results support equivalence under every weight configuration, with small effect sizes ($|r| \leq 0.28$), moderate power (44–71%) and confidence intervals falling entirely within the a priori margin ($\Delta = 0.0202$). The result holds across all four configurations, highlighting how it isn't an artefact of any single similarity component of IDS. This provides a result in support of RQ2 and thus the system's capability to avoid redundancy while retaining exploratory parallelization capabilities of the complex information spaces that relational databases represent. Column space division across concurrently operating subagents doesn't measurably reduce semantic diversity of the resulting rule set relative to human specification.

6 Vision: Closed-Loop DQ Layer

The prototype implemented addresses rule discovery and governance. The broader vision extends this into a full closed-loop DQ layer with two additional components (full details and system diagram in Appendix §B.5, §B.6 and Figure 2).

Execution Dashboard and Reporting Agent (EDRA) runs checks idempotently on stable dataset snapshots, and writes immutable *ExecutionTraces* with violation counts, sample failing rows, runtime, and provenance (rule version, ontology snapshot, dataset snapshot). It then aggregates ExecutionTraces into interactive health dashboards, severity-weighted dataset scores, rule-stability trend views, and audit-ready exports, providing a single pane of glass for data owners, risk, compliance, and engineering.

Weights	HL	90% CI	r	Power
Equal	-0.003	$[-0.010, +0.012]$	$+0.16$	57%
Semantic	-0.005	$[-0.015, +0.008]$	$+0.22$	44%
Syntax	-0.005	$[-0.016, +0.010]$	$+0.25$	71%
Scope	$+0.003$	$[-0.004, +0.019]$	-0.28	53%

Table 6: Non-parametric TOST equivalence analysis for RQ2 IDS (Full Agentic − Guided); equivalence margin $\Delta = 0.0202$. HL: Hodges–Lehmann shift with exact CI from the inverted Mann-Whitney statistic; r: rank-biserial effect size. Runs required for 80% power: 11 (Equal), 13 (Semantic), 9 (Syntax), 11 (Scope).

Root Cause Analysis and Detective Agent (RCADA) reads ExecutionTraces, ontology risk metadata, dataset lineage, and profiler trends to produce ranked root-cause hypotheses using an explicit cause taxonomy (provider outage, pipeline regression, upstream drift, contextual event, ontology mismatch, rule mis-specification), bounded near-term failure forecasts, and non-binding prescriptive suggestions for steward review.

Together, these components close the feedback loop: EDRA executes and summarizes, RCADA diagnoses and proposes, humans decide, and the repository records updates for the next cycle. Because all rules and traces are linked to stable ontology node IDs, the system remains robust to schema and provider changes by reasoning over semantic entities rather than raw column names.

7 Discussion

We have proposed a vision for an ontology-grounded, versionable DQ system, taking a concrete first step by implementing and evaluating its agentic rule-discovery core and the HITL components needed for auditability/reproducibility requirements in regulated domains. Our results on two real-world databases show that the full-agentic mode produces rules of comparable diversity to the human baseline, supporting the semantic-coverage premise of the vision. However, the lack of established equivalence combined with the underpowered nature of the test for the first research question leaves it open, and we therefore make no current conclusions towards the quality parity between modes which will be further explored in future works. The diversity finding is consistent across two datasets from different domains (financial/retail), which is encouraging but preliminary evidence of domain-agnostic transferability.

Limitations and future steps We identify five limitations that qualify these results. First, rule *complexity* is not yet measured: our metrics capture quality and diversity but not whether automatically discovered rules are systematically simpler than human-specified ones, which would represent a meaningful gap in expressive power. Future evaluations will focus on quantifying also this dimension building on top of IDS and carrying out more comprehensive analysis on the metric's correlation to human judgment. Second, both datasets might share characteristics across domains (ex both have a transactional nature) that have not been controlled for and which may not highlight the true limits of domain transferability; broader domain coverage is needed to test the system's limits. Expanding to datasets across existing relational benchmarks and synthetic data generation benchmarks is a priority for future work, with the longer-term goal of curating a benchmark pack including data snapshots, evolution scripts, governance text, and gold rule catalogs to share with the broader research community to support re-playable, lifecycle-aware evaluation specifically of Agentic DQ systems. Third, scaling reviewers is inherently expensive, while small groups introduce issues with statistical power, as seen for RQ1 where a low power does not make it possible to draw conclusions on equivalence, or for RQ2 where a higher power would allow to draw even more confident conclusions. Future work will thus complement a larger reviewer set with automated evaluation based on rule execution outcomes and expert-validated reference gold rule sets. Fourth, compute budgets for these tests restricted us to a single LLM and temperature setting. We recognize the need for future model/temperature ablation test to better understand the robustness and transferability of the proposed harness. Finally, the current evaluation covers only the rule-discovery component; the full vision (execution+dashboarding, and root-cause analysis) remains to be implemented and evaluated end-to-end, thus at present we do not claim any empirical demonstration of the full architecture that we leave for future work.

Open problems. Grounding noisy governance text into calibrated ontology updates, mapping abstract concepts across heterogeneous schemas, handling schema evolution without destabilizing linked rules, and measuring false positives at scale remain key open challenges we hope this framework can help start to address systematically.

8 Acknowledgments

The authors would like to thank Aniket Joshi for having sourced the databases and prepared them for use, while Michael Santarossa and Roberto Varinelli for their feedback and precious advice on the work carried out for this paper.

References

BakerHostetler. Us crypto exchange consents to $100m dfs settlement for AML compliance failures. https://www.bakerlaw.com/insights/us-crypto-exchange-consents-to-100m-dfs-settlement-for-aml-compliance-failures/. Accessed: 2026-02-11.

Alla Chepurova, Aydar Bulatov, Yuri Kuratov, and Mikhail Burtsev. 2025. Wikontic: Constructing wikidata-aligned, ontology-aware knowledge graphs with large language models. *arXiv preprint arXiv:2512.00590*.

Sijie Dong, Soror Sahri, Themis Palpanas, and Qitong Wang. 2025. Automated data quality validation in an end-to-end gnn framework. *arXiv preprint arXiv:2502.10667*.

Santosh Durgam. 2025. Scalable data quality frameworks for record keeper aggregation in financial platforms-proposes a framework to standardize and enhance the quality of financial datasets across heterogeneous record keepers. *International journal of data science and machine learning*, 5(01):70–97.

Hui Feng, Yuntzu Yin, Emiliano Reynares, and Jay Nanavati. 2025. Ontologyrag: Better and faster biomedical code mapping with retrieval-augmented generation (rag) leveraging ontology knowledge graphs and large language models. In *International Workshop on Knowledge-Enhanced Information Retrieval*, pages 71–86. Springer.

Wenbin Guo, Xin Wang, Jiaoyan Chen, Zhao Li, and Zirui Chen. 2025. Ontology-enhanced knowledge graph completion using large language models. In *International Semantic Web Conference*, pages 61–80. Springer.

Aleksey Lapin, Igor Hromov, Stanislav Chumakov, Mile Mitrovic, Dmitry Simakov, Nikolay O Nikitin, and Andrey V Savchenko. 2025. Lightautods-tab: Multi-automl agentic system for tabular data. *arXiv preprint arXiv:2507.13413*.

Jisoo Lee, Raeyoung Chang, Dongwook Kwon, Harmanpreet Singh, and Nikhil Verma. 2025. Gemmas: Graph-based evaluation metrics for multi agent systems. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 1522–1532.

Jinyang Li, Binyuan Hui, Ge Qu, Jiaxi Yang, Binhua Li, Bowen Li, Bailin Wang, Bowen Qin, Ruiying Geng, Nan Huo, and 1 others. 2023. Can llm already serve as a database interface? a big bench for large-scale database grounded text-to-sqls. *Advances in Neural Information Processing Systems*, 36:42330–42357.

Jeshwanth Reddy Machireddy. 2023. Data quality management and performance optimization for enterprise-scale etl pipelines in modern analytical ecosystems. *Journal of Data Science, Predictive Analytics, and Big Data Applications*, 8(7):1–26.

Pete Schroeder Michelle Price and Tatiana Bautzer. U.s. bank regulators fine citi $136 million for failing to address long-standing data issues. https://www.reuters.com/business/finance/us-bank-regulators-fine-citi-136-million-failing-address-longstanding-data-2024-07-10/. Published: 2024-07-10. Accessed: 2026-02-11.

Fan Nie, Junlin Wang, Harper Hua, Federico Bianchi, Yongchan Kwon, Zhenting Qi, Owen Queen, Shang Zhu, and James Zou. 2026. Dsgym: A holistic framework for evaluating and training data science agents. *arXiv preprint arXiv:2601.16344*.

Victor Ojewale, Harini Suresh, and Suresh Venkatasubramanian. 2026. Audit trails for accountability in large language models. *arXiv preprint arXiv:2601.20727*.

Manisha Padala, Lokesh Nagalapatti, Atharv Tyagi, Ramasuri Narayanam, and Shiv Kumar Saini. 2025. Tab-shapley: Identifying top-k tabular data quality insights. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 12435–12442.

Matteo Pancini, Matteo Camilli, Giovanni Quattrocchi, and Damian Andrew Tamburri. 2025. Engineering mlops pipelines with data quality: A case study on tabular datasets in kaggle. *Journal of Software: Evolution and Process*, 37(9):e70044.

Hima Patel, Nitin Gupta, Naveen Panwar, Ruhi Sharma Mittal, Sameep Mehta, Shanmukha Guttula, Shashank Mujumdar, Shazia Afzal, Srikanta Bedathur, and Vitobha Munigala. 2022. Automatic assessment of quality of your data for ai. In *Proceedings of the 5th Joint International Conference on Data Science & Management of Data (9th ACM IKDD CODS and 27th COMAD)*, pages 354–357.

Shuofei Qiao, Yanqiu Zhao, Zhisong Qiu, Xiaobin Wang, Jintian Zhang, Zhao Bin, Ningyu Zhang, Yong Jiang, Pengjun Xie, Fei Huang, and 1 others. 2025. Scaling generalist data-analytic agents. *arXiv preprint arXiv:2509.25084*.

Joshua Robinson, Rishabh Ranjan, Weihua Hu, Kexin Huang, Jiaqi Han, Alejandro Dobles, Matthias Fey, Jan E Lenssen, Yiwen Yuan, Zecheng Zhang, and 1 others. 2024. Relbench: A benchmark for deep learning on relational databases. *Advances in Neural Information Processing Systems*, 37:21330–21341.

Devender Saini, Bhavika Jain, Nitish Ujjwal, Philip Sommer, Dan Romuald Mbanga, and Dhagash Mehta. 2025. A unified ai system for data quality control and dataops management in regulated environments. *arXiv preprint arXiv:2512.05559*.

Timur Sattarov, Dayananda Herurkar, and Jörn Hees. 2022. Explaining anomalies using denoising autoencoders for financial tabular data. *arXiv preprint arXiv:2209.10658*.

Avinash Kumar Singh, Bhaskarjit Sarmah, and Stefano Pasquali. 2025. Finch: Financial intelligence using natural language for contextualized sql handling. *arXiv preprint arXiv:2510.01887*.

Ji Sun, Guoliang Li, Peiyao Zhou, Yihui Ma, Jingzhe Xu, and Yuan Li. 2025. Agenticdata: An agentic data analytics system for heterogeneous data. *arXiv preprint arXiv:2508.05002*.

Katherine Thornton, Harold Solbrig, Gregory S Stupp, Jose Emilio Labra Gayo, Daniel Mietchen, Eric Prud'Hommeaux, and Andra Waagmeester. 2019. Using shape expressions (shex) to share rdf data models and to guide curation with rigorous validation. In *European Semantic Web Conference*, pages 606–620. Springer.

Manish Tomar, Vasudevan Ananthakrishnan, and Muthuraman Saminathan. 2024. Agentic ai-powered data quality guardians for regulated industries. *Journal of Artificial Intelligence General science (JAIGS) ISSN: 3006-4023*, 3(1):507–531.

Ziting Wang, Shize Zhang, Haitao Yuan, Jinwei Zhu, Shifu Li, Wei Dong, and Gao Cong. 2025. Fdabench: A benchmark for data agents on analytical queries over heterogeneous data. *arXiv preprint arXiv:2509.02473*.

Boming Xia, Qinghua Lu, Liming Zhu, Zhenchang Xing, Dehai Zhao, and Hao Zhang. 2025. Evaluation-driven development and operations of llm agents: A process model and reference architecture. *Available at SSRN 5775317*.

Appendix

A Beyond Traditional Data-Quality Checks

Most production pipelines start with generic checks that do not depend on the domain. Common examples of traditional data quality checks are completeness and missingness, uniqueness and duplicate detection, data type and parsing validity, format checks such as date and identifier patterns, value ranges, basic outliers, and referential integrity across keys. These checks are necessary, but they often miss errors that only become visible when the data is interpreted in its real business meaning.

We refer to the domain-specific sophisticated data quality checks as *policy grounded semantic rules*. These are domain dependent rules that use real world constraints and governance text such as policies, data dictionaries, and reporting definitions. They are often conditional, time aware, and cross table. They also need provenance so reviewers can see which document or evidence justified the rule. Some examples are shown in Table 7.

Rule type	Finance example	Grounding
Calendar and market hours	No equity trades on Sunday. Trades outside market hours must be flagged or explained.	Market calendar, policy
Temporal consistency	Settlement date must be after trade date. For many products it follows a fixed lag such as T plus 2, with exceptions.	Policy, product spec
Cross field meaning	Notional should match quantity times price within tolerance and currency scaling.	Domain math, evidence
Entity state	Counterparty must be active on trade date. Trades for blocked accounts must be rejected.	Reference data, policy
Cross table reconciliation	Trade totals must reconcile with positions and ledger aggregates at the portfolio level.	Reporting definition

Table 7: Examples of policy grounded semantic rules that go beyond traditional checks.

In addition, we also frame our data quality system around the four classic analytics types: descriptive, diagnostic, predictive, prescriptive that are a standard in the field. Table 8 maps the full envisioned agents stack to said analytics types.

B Formal Artifact Specifications

B.1 Ontology specifications

We propose a clear, minimal, and extensible ontology specification for the ontology-enhanced agentic data quality system. The ontology is designed to be LLM-friendly, human-readable, and agent-operable, while remaining robust across domains. Design goals include stability (a fixed structural contract so agents can interoperate), expressiveness (rich semantics captured primarily in natural language), partial observability (nodes may be incomplete or uncertain by design), domain generality (the same ontology spine works across tabular domains), and LLM-first reasoning (semantics optimized for textual reasoning).

The ontology is a typed semantic graph: each node represents a semantic unit in tabular data, and relationships between nodes are captured explicitly or via references. The ontology supports exactly five node types, and this set is closed by design; new semantic distinctions should be expressed via `semantic_role`, not by adding node types (Table 9).

Every ontology node must conform to the minimal schema below.

```
{
  "node_id": "string",
  "node_type": "Dataset | Entity | Attribute |
   Relationship | Vocabulary",
  "name": "string",
  "semantic_role": "string",
  "confidence": "float",
  "sources": ["data | domain | human
  | external"],
  "version": "string",
  "meaning": {
    "description": "string"
  }
}
```

Field meanings are summarized in Table 10.

Beyond `meaning.description`, we support a soft semantic envelope for natural-language keys such as `constraints`, `typical_behavior`, `edge_cases`, and `risk_notes`. For example:

```
"meaning": {
 "description": "Executed equity trades table.",
  "constraints": "Each row is one executed
trade.",
 "typical_behavior": "High volume during mar-
ket hours.",
 "edge_cases": "After-hours trades and correc-
tions.",
  "risk_notes": "Used for risk and regulatory
reporting."
}
```

Agents may add new keys such as `regulatory_context` or `temporal_behavior`, but must not remove or overload existing keys.

B.2 Rules specifications

A structured, versioned rule repository works in tandem with the ontology. It is where data quality

Agent	Descriptive	Diagnostic	Predictive	Prescriptive
Ontology Gen.	Grounds columns into entities, measures, and units.	Adds semantic context for anomaly explanation.	Marks forecastable measures and derived metrics.	Enables domain-aware imputation and unit normalization.
Rule Discovery	Drafts checks with clear summaries and coverage metrics.	Encodes conditional/composite rules for diagnosis.	Proposes drift-sensitive and stability-aware rules.	Suggests adaptive thresholds and remediation-aware rules.
Implementer	Translates approved rules into executable code.	Generates diagnostic checks capturing failure evidence.	Produces code artifacts that accumulate execution history for forecasting.	Emits validated, controlled fixes to apply safeguards/remediation.
Human review	Confirms expected vs. concerning signals.	Confirms or rejects proposed causes.	Decides which predicted risks are actionable.	Approves, edits, or blocks remediation strategies.
EDRA	Executes checks and visualizes quality metrics and trends.	Captures failure evidence and supports drill-down/lineage views.	Collects execution history and displays risk scores and drift forecasts.	Presents recommended actions for review.
RCADA	Consumes descriptive trace summaries.	Produces ranked RCA hypotheses with evidence.	Forecasts future failures from recurring patterns.	Suggests targeted interventions and rule updates.

Table 8: How agents and human review support descriptive, diagnostic, predictive, and prescriptive behavior.

Node type	Purpose
Dataset	Represents a table, file, or view as a whole.
Entity	Represents a real-world object described by rows.
Attribute	Represents a column or derived field.
Relationship	Represents row-to-row or table-to-table linkage.
Vocabulary	Represents a value space or code system.

Table 9: Exhaustive ontology node types.

Field	Meaning
`node_id`	Stable unique identifier.
`node_type`	One of the five exhaustive types.
`name`	Human-readable label.
`semantic_role`	Open-ended semantic classification.
`confidence`	Belief score in correctness, from 0 to 1.
`sources`	Provenance of semantic evidence.
`version`	Ontology version tag.
`meaning. description`	Canonical semantic explanation.

Table 10: Field semantics for the minimal ontology node schema.

knowledge is proposed, reviewed, executed, audited, and evolved. It serves as a staging ground for agent-generated rules, a human control point for approval and refinement, a persistent memory of data quality intent, a bridge from ontology to execution, and an audit log for governance. We propose core principles of being ontology-linked, text-first, lifecycle-aware, provenance-rich, and executable without being bound to any one runtime. Every rule must conform to a minimal schema.

```
{
  "rule_id": "string",
  "rule_name": "string",
  "linked_nodes": ["ontology_node_id"],
  "rule_type": "descriptive | diagnostic | predictive | prescriptive",
  "status": "draft | approved | deprecated",
  "confidence": "float",
  "generated_by": "agent | human",
  "version": "string",
  "logic": {
   "intent": "string"
  }
}
```

The fields carry the following meanings: `linked_nodes` ties the rule to one or more ontology nodes; `rule_type` captures the analytics orientation; `status` captures lifecycle state; `confidence` captures belief in correctness; and `generated_by` records origin.

Rule logic is stored as typed natural-language blocks.

```
"logic": {
  "intent": "What quality property this rule enforces",
  "condition": "When the rule applies",
  "expectation": "What must hold true",
  "severity": "Impact of violation",
  "remediation": "Suggested corrective action",
  "rationale": "Why this rule exists"
}
```

Agents may add blocks such as `regulatory_basis`, `statistical_basis`, or `exceptions`, but must not redefine existing keys. Rules move through explicit states (Proposed, Approved, Rejected). State transitions are recorded as provenance events. Every rule must reference at least one ontology node; rules inherit meaning from linked nodes, and ontology updates may trigger rule re-evaluation. Example:

Module	What it outputs
Structural	Schema facts such as column names, physical types, counts, declared keys, partitions.
Distributional	Global numeric and categorical summaries from full scan or distributed aggregation.
Temporal	Frequency, gaps, weekday or holiday patterns, rolling rates with confidence.
Relationship	Dependencies, candidate foreign keys, correlations, ordering constraints.
Quality signals	Null rates, duplicates, declared constraint counts, drift indicators.

Table 11: Profiler modules and their evidence outputs.

`"linked_nodes": ["col:trade_amount", "entity:trade"]`.

B.3 Profiler subsystem specifications

The Profiler Subsystem is a deterministic, non-LLM component that extracts complete, coverage-aware evidence summaries from raw tabular data. It is the epistemic foundation of the system. Agents reason over profiler outputs rather than raw rows, which can be too large to route through an agent loop. The intended division of labor is that profilers compute facts, and agents interpret meaning.

Profiler inputs are raw tables or files, file-format metadata, execution configuration, and optional calendars or controlled business context. The profiler emits a *Statistical Evidence Pack*, a structured artifact saved to the Artifact repository and consumed as immutable evidence by Ontology generation. We summarize the main profiling modules in Table 11. The Statistical Evidence Pack is machine-readable; examples are shown below.

```
{
  "row_count": 125000000,
  "columns": {
   "trade_id": "string",
   "trade_date": "date",
   "trade_amount": "decimal"
  }
}

{
  "trade_amount": {
   "p50": 120000,
   "p99": 5000000,
   "max": 87000000,
   "negative_rate": 0.0
  }
}

{
  "trade_date": {
   "inferred_frequency": "daily",
   "weekday_gaps": ["Sunday"],
   "max_gap_days": 2,
   "calendar_alignment": "NYSE",
   "confidence": 0.97
  }
}
```

The profiler records coverage and sampling policy explicitly.

```
{ "profiling_mode": "full_scan",
 "coverage": "100%", "sampling":
 "none" }
```

If sampling is required, the policy and limitations are recorded. Profiler outputs are deterministic for a given dataset version. Profiling code versions are logged. The Evidence Pack is saved to the Artifacts Repository and reused by downstream agents. Profilers never consume ontology or rules.

B.4 Ontology generation specifications

Ontology generation transforms profiler evidence into a semantic understanding of the dataset and emits ontology creation or update operations. It never reasons over raw rows at scale and instead consumes coverage-aware evidence summaries from the Statistical Evidence Pack. Ontology generation reads the Statistical Evidence Pack from the Artifacts Repository. Mandatory inputs include structural metadata, distributional profiles, temporal profiles, and relationship and dependency profiles. Optional inputs include user hints, and controlled documentation. A key design principle is that Ontology generation never ingests raw rows; it reasons only over deterministic profiler evidence to avoid sampling bias and preserve temporal semantics.

The system emits a set of Ontology Operations as structured outputs to force the LLM to conform to the node schema (transactional `CREATE_NODE` and `UPDATE_NODE` operations that update the ontology namespace). For example, from a column profile for `trade_amount`, the system can infer a monetary measure role and emit a manifest entry plus a node operation.

```
{
  "column": "trade_amount",
  "semantic_role": "monetary_measure",
  "meaning": {
   "description": "Executed notional value of a
trade.",
   "constraints": "Must be non-negative.",
   "risk_notes": "Errors impact exposure and
reporting."
  },
  "confidence": 0.92
}
```

Ontology operations are selected based on existing ontology match confidence. A new ontology namespace is created if no match is strong enough or isolation is requested. Otherwise Ontology generation updates existing nodes when new evidence refines meaning, constraints, units, or relationships.

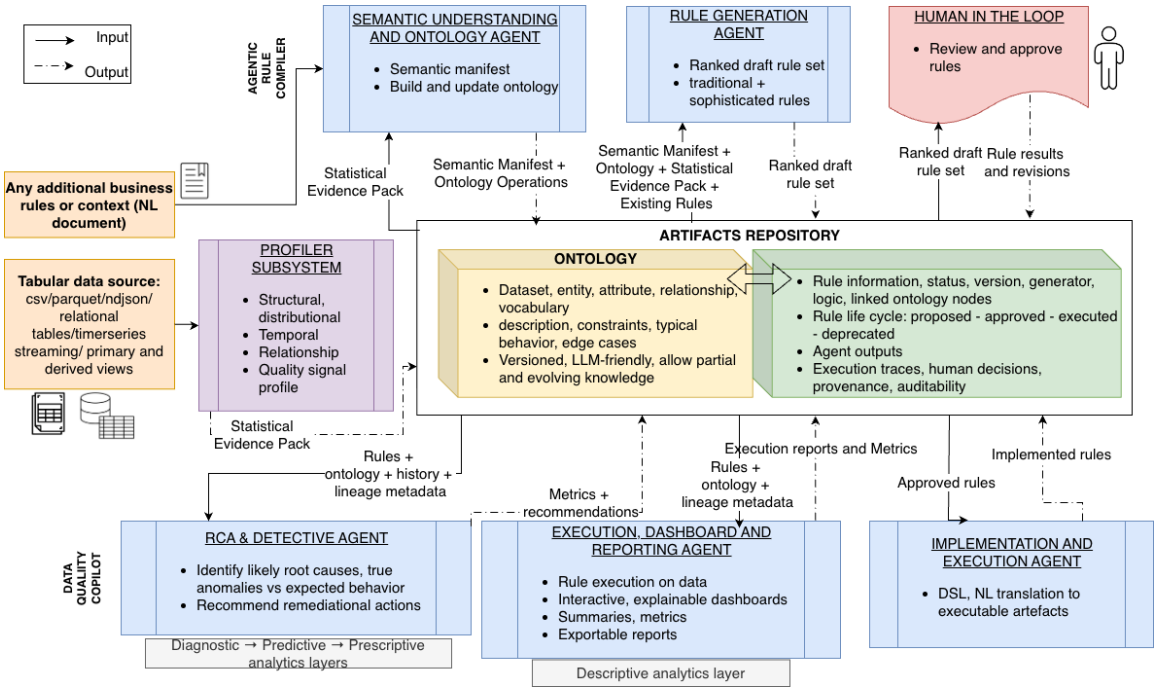

Figure 2: Full envisioned system overview: all components sharing a central Artifacts Repository across the full DQ lifecycle.

B.5 EDRA specifications (Vision)

EDRA is the interpretation and decision interface of the system. It is read-heavy, analytics-heavy, and explanation-focused. EDRA does not create rules and does not execute checks. Instead, it aggregates artifacts and human review in the Artifacts Repository, runs checks idempotently on stable dataset snapshots, writes immutable *Execution-Traces* with violation counts, sample failing rows, runtime, and provenance (rule version, ontology snapshot, dataset snapshot), computes higher-level analytics, and presents actionable, auditable views to stakeholders (data owners, risk, compliance, engineering).

EDRA's primary goal is to provide a single pane of glass for data-quality health, make failures explainable and traceable, support governance and audit, enable trend analysis and prioritization, and close the loop from rule failure to decision and remediation.

EDRA consumes read-only artifacts from the Artifacts Repository: rule definitions and metadata, ExecutionTraces and execution timelines, ontology references (node IDs, meanings, risk notes), and human decisions (approve, reject, modify with comments). Optional lineage metadata can be used to surface source-to-transformation context in investigations.

EDRA produces interactive dashboards, exportable audit-ready reports (e.g., JSON or CSV), alerts and summaries, dataset-level risk scores and trend forecasts, and non-binding prescriptive suggestions such as remediation playbooks or rule-tuning proposals. It organizes insights explicitly as descriptive (what happened), diagnostic (why), predictive (what is likely next), and prescriptive (what to do) analytics over the data-quality system itself.

A minimal EDRA UI includes (i) an Overview Dashboard with dataset health, top failing rules, and affected ontology nodes; (ii) a Rule Catalog with filters by dataset, ontology node, severity, status, and last execution; (iii) a Failure Explorer that shows the rule definition and rationale, violation statistics and samples, linked ontology meaning and risks, and supporting context such as lineage when available; (iv) Temporal and trend views for failure rate, recurrence, and stability; and (v) an Audit and governance panel that replays the rule lifecycle from draft to approved, modified, and deprecated with evidence exports.

EDRA computes severity-weighted dataset health, failure rate and recurrence, rule stability over time, coverage (the fraction of rows evaluated by active rules), and ontology risk exposure by aggregating the risk weights of affected ontology nodes.

B.6 RCADA specifications (Vision)

RCADA closes the loop. Given rule failures and execution traces, it explains why violations happened and what to do next by combining historical repository context, profiler trends, lineage, and optional external signals. RCADA primarily operates in the diagnostic, predictive, and prescriptive analytics layers; descriptive summaries are surfaced by EDRA.

For a set of failing rules and datasets, RCADA (i) identifies and ranks likely root causes, (ii) distinguishes true anomalies from contextual or expected behavior, and (iii) recommends concrete, human-controlled remediation actions with calibrated confidence.

RCADA reads (a) ExecutionTraces (failure rates, example rows, timestamps, runtime), (b) rule metadata (severity, linked ontology nodes, prior stability), (c) ontology metadata (semantic meaning, risk notes, provider dependencies), (d) artifact history (past outcomes and human decisions on similar failures), (e) dataset lineage (provider, pipeline stage, schema versions, deploy tags), and (f) profiler time series (null rates, distributions, cardinality trends). Optional external context includes calendars/holidays, outage logs, and governance or regulatory change notes.

RCADA writes a structured RCA Report artifact (root-cause candidates with evidence, classification such as contextual anomaly vs. likely incident, recommended actions with owners and urgency, and confidence/uncertainty). It can also emit bounded forecasts of near-term recurrence risk and non-binding prescriptive suggestions (e.g., rule refinement proposals or temporary usage restrictions) for human review.

RCADA uses an explicit and explainable cause taxonomy (e.g., provider outage, pipeline regression, upstream drift, contextual event, ontology mismatch, rule mis-specification). Candidates are scored using temporal alignment, cross-source agreement when comparable signals exist, historical recurrence, and ontology risk alignment. RCADA also uses human decisions as weak labels to estimate likely false positives and to flag systematically over-strict rules.

Example: Rule `R-042` (`trade_amount` non-negative) shows a failure spike after `2026-01-25`. RCADA attributes 97% of violations to `source=OTC_FEED_X`, notes ontology risk for PnL and regulatory reporting, and recommends an engineering escalation plus a steward-reviewed source-conditional exception with confidence.

C Blinded Rule Review Protocol

This appendix specifies the blinded, mixed-mode, overlapping-assignment review design referenced in §5, and the reviewer pool that carried it out. The design was built to satisfy three requirements: every rule is rated by exactly two different reviewers (enabling inter-rater agreement to be reported), no reviewer ever rates a rule they authored and both generation modes are present in every batch (so mode is never confounded with batch or rater identity).

C.1 Block design

Rules from each dataset were split into four blocks (A–D). Each block contains the rules authored by exactly two of the four reviewers and is rated by the two reviewers whose rules it does *not* contain:

Block	Authored by	Rated by
A	R2, R3	R1, R4
B	R3, R4	R1, R2
C	R4, R1	R2, R3
D	R1, R2	R3, R4

Critically, both modes produced by a given author travel into the same block together: Block A, for instance, contains both R2's and R3's Full Agentic *and* Guided rules. An author's agentic output is never routed to a different block than their guided output, since splitting them would let a between-author skill difference masquerade as a between-mode effect. Within each author's allotment, rules are sorted by mode and dealt alternately across the author's two permitted blocks, so that each mode is split evenly rather than concentrated in one block. Presentation order within a batch is shuffled independently per (block, reviewer) pair, and the resulting position of each rule is logged.

This yields, per dataset, a connected co-rating cycle across reviewers, (R1,R4), (R1,R2), (R2,R3), (R3,R4), equal review workload across reviewers, and every rule rated by exactly two people who did not write it. One limitation follows directly from having four reviewers and two raters per block: each run is seen by three of the four reviewers rather than all four, and the pairs (R1,R3) and (R2,R4) never co-rate directly. Krippendorff's α remains valid over the connected graph, but we do not report a complete 4×4 pairwise agreement matrix, since two of the six pairs were never jointly observed.

C.2 Validation

Before release, every batch was checked against five automated assertions: exhaustive, non-overlapping coverage of all rules across blocks; exactly two ratings per rule; no reviewer assigned to a block containing their own authored rules; both modes present in every block; and both modes present for every author within every block they occupy.

C.3 Inter-rater agreement

The design yields 541 doubly-rated rules and 1,082 ratings. Table 4 reports the coefficients; this section records the disagreement breakdown and two reporting choices. The verdict scale for disagreement is ordered: reject < modify < accept, since and accept/modify disagreement is less severe than an accept/reject disagreement, treating them as unordered categories would understate the true agreement. Reviewers agree exactly on 77.3% of rules, while they differ by one step on the ordinal scale on 13.1%, and differ at the extremes (`accept/reject`) on 9.6%. Per subset the split is 76.0/15.1/8.8 on Berka, 79.0/10.3/10.7 on H&M, 76.8/13.8/9.4 on agentic rules and 78.1/11.9/10.0 on guided rules. We report AC2 alongside α because the verdict distribution is concentrated: with 84.8% of ratings in one category, α is deflated by construction, while AC2 is prevalence-robust.

D Benchmarking Agenda

The experiments in §5 focus on rule-generation quality and diversity. Comprehensive valuation dimensions for the full envisioned system extend beyond these to include: execution correctness (reproducibility of ExecutionTraces from artifact snapshots); ontology grounding quality (semantic role accuracy, confidence calibration); HITL efficiency (accept/modify/reject rates, iterations-to-convergence, reviewer effort proxies); temporal adaptation (portfolio stability under schema drift, MTTD/MTTR); and explanation quality (usefulness of RCA hypotheses and prescriptive suggestions).

As mentioned in the discussion, we thus envision benchmarks that ship not only tables, but the *governance-and-audit context* that the system consumes and produces:

- **Data + evolution timeline**: snapshots/streams of related tables (finance as anchor), with scripted schema/provider/policy evolution and injected drifts/anomalies.
- **Ontology spine**: a typed Dataset/Entity/Attribute/Relationship/Vocabulary ontology linking columns/joins to meaning envelopes (`constraints`, `typical_behavior`, `risk_notes`).
- **Governance corpus**: policy/regulatory/dictionary text mapped (at least coarsely) to ontology nodes, enabling provenance links from rules back to intent.
- **Repository event log**: an append-only history of rule versions, HITL decisions, and ExecutionTraces, so adaptation and auditability are testable rather than asserted.

We aim, together with the broader community to build these shared benchmark packs and scoring harnesses around this contract, with license-clean datasets, governance corpora, gold rule catalogs, and standard scoring for precision/recall, coverage, stability, and explanation quality.

E Rule Taxonomy

To characterize what the system actually mines rather than only how well the rules it mines are received, we manually tagged a randomly drawn sample of 85 AI-mined rules from the Full Agentic mode (25% of all AI-mined rules) against eight standard relational and data-quality rule categories. Table 12 reports the resulting category shares. We provide also representative examples of AI-mined rules for each of the classes present in the table.

Range / statistical


```
{
 "rule_id": "rule:client_district_id_range_validity",
  "rule_name": "District ID Range Constraint",
  "linked_nodes": ["col:client_district_id",
 "rel:client_to_district"],
  "logic": {
   "intent": "Ensure district_id falls within the
 valid administrative range, protecting referen-
 tial integrity.",
   "condition": "All non-null district_id values
 in client.",
   "expectation": "Integers between 1 and 77
 inclusive.",
   "supporting_evidence": "Ontology states in-
 teger 1–77; profiling shows min=1, max=77,
 distinct_count=77."
  },
  "executable_code": "SELECT * FROM client
 WHERE
    district_id < 1 OR district_id > 77
    OR district_id IS NULL"
 }
```


Key / uniqueness

{
"rule_id": "rule:client_client_id_uniqueness",
"rule_name": "Client ID Uniqueness Constraint",
"linked_nodes": ["col:client_client_id", "tbl:client"],
"logic": {
"intent": "Enforce that client_id serves as a valid primary key by ensuring each value appears exactly once.",
"condition": "All rows where client_id is present.",
"expectation": "Every client_id unique across all 5369 rows, no duplicates permitted.",
"supporting_evidence": "Profiling shows distinct_count=5369 equals row_count=5369, confirming uniqueness."
},
"executable_code": "SELECT * FROM client WHERE client_id IN
(SELECT client_id FROM client GROUP BY client_id
HAVING COUNT(*) > 1)"
}

{
"rule_id": "rule:loan_date_format_iso8601",
"rule_name": "Loan Date ISO 8601 Format Validation",
"linked_nodes": ["col:loan_date"],
"logic": {
"intent": "Ensure date values conform to ISO 8601 (YYYY-MM-DD) and are valid calendar dates.",
"condition": "All rows in the loan table.",
"expectation": "4-digit year, month 01–12, day 01–31, and the combination forms a valid calendar date.",
"supporting_evidence": "Ontology requires fixed-length ISO 8601; profiling shows all 682 values of length 10.0 with std_length=0.0."
},
"executable_code": "SELECT * FROM loan WHERE date !~
'^\d{4}-(0[1-9]|1[0-2])-(0[1-9]|[12]\d|3[01])$' OR date ≠ TO_DATE(date, 'YYYY-MM-DD')::text"
}

Categorical domain / vocabulary

Completeness / NOT NULL

{
"rule_id": "rule:card_type_vocabulary_conformance",
"rule_name": "Card Type Vocabulary Conformance",
"linked_nodes": ["col:card_type", "vocab:card_type_codes"],
"logic": {
"intent": "Enforce that card type values conform to the controlled vocabulary of valid card tier codes.",
"condition": "All rows in card where type is not null.",
"expectation": "Every value one of: classic, junior, gold.",
"supporting_evidence": "Profiling shows exactly 3 distinct values: classic (659), junior (145), gold (88), matching vocab:card_type_codes."
},
"executable_code": "SELECT * FROM card WHERE type NOT IN
('classic', 'junior', 'gold')"
}

{
"rule_id": "rule:trans_account_id_non_null_positive",
"rule_name": "Account ID Non-Null and Positive",
"linked_nodes": ["col:trans_account_id"],
"logic": {
"intent": "Ensure account_id is always populated and valid as a foreign key reference to accounts.",
"condition": "For all rows in the trans table.",
"expectation": "account_id must be non-null and a positive integer (≥ 1).",
"supporting_evidence": "Ontology states account_id cannot be null and must be a positive integer; profiling confirms 0 nulls, min=1, across all 1,056,320 rows."
},
"executable_code": "SELECT * FROM trans WHERE account_id
IS NULL OR account_id < 1"
}

Syntactic format

Functional dependency

{
"rule_id": "rule:perceived_colour_master_id_name_consistency",
"rule_name": "Perceived Colour Master ID-Name Consistency",
"linked_nodes": ["col:article_perceived_colour_master_id", "col:article_perceived_colour_master_name"],
"logic": {
"intent": "Enforce that each master colour id maps to exactly one master colour name (id → name).",
"condition": "Every distinct id value, excluding the -1 sentinel for unknown.",
"expectation": "Distinct (id, name) pairs equal distinct id values (20); each id maps to a single name.",
"supporting_evidence": "Both columns have exactly 20 distinct values across 105,542 rows; matching distinct counts indicate a controlled vocabulary relationship."
},
"executable_code": "WITH inconsistent_ids AS (SELECT id FROM
article WHERE id ≠ -1 GROUP BY id HAVING
COUNT(DISTINCT name) > 1) SELECT * FROM article
WHERE id IN (SELECT id FROM inconsistent_ids)"
}

Inclusion dependency

{
"rule_id": "rule:account_district_referential_integrity",
"rule_name": "Account District Referential Integrity",
"linked_nodes": ["rel:account_to_district", "rel:district_id_to_name"],
"logic": {
"intent": "Ensure every account.district_id references a valid district record in the district table.",
"condition": "Every row in the account table.",
"expectation": "district_id must match an existing A1 value in district (1–77 inclusive).",
"supporting_evidence": "Profiling shows 77 distinct district_id values matching the district table's 77 rows."
},
"executable_code": "SELECT * FROM account WHERE district_id
IS NULL OR district_id NOT IN (SELECT A1 FROM district)"
}

Denial constraint

{
"rule_id": "rule:district_A5_A6_A7_demographic_coherence",
"rule_name": "District A5-A6-A7 Demographic Coherence",
"linked_nodes": ["col:district_A5", "col:district_A6", "col:district_A7", "rel:demographic_metrics_coherence"],
"logic": {
"intent": "Ensure internal consistency among related
demographic metrics A5, A6 and A7.",
"condition": "Every row where A5, A6 and A7 are all non-zero.",
"expectation": "Pairwise ratios remain within the bounds implied by observed means (A5:A6:A7 ≈ 8:4:1); extreme deviations indicate quality issues.",
"supporting_evidence": "All three columns share similar zero rates (5–6%); ontology asserts internal coherence."
},
"executable_code": "SELECT * FROM district WHERE A5 > 0 AND
A6 > 0 AND A7 > 0 AND ((A5/A6 < 0.67 OR A5/A6 > 6)
OR (A5/A7 < 2.57 OR A5/A7 > 23.1)
OR (A6/A7 < 1.3 OR A6/A7 > 11.7))"
}

F Additional Figures

Below we provide a series of additional figures and plots to accompany and complement visually the results discussed in §5.4:

Category	Applied to	Share
Range / statistical	numeric bounds, thresholds, sign, outlier bounds	24.2%
Key / uniqueness	primary-key or unique-column assertions	23.2%
Categorical domain / vocabulary	enumerated valid-value list	22.1%
Syntactic format	regex, length, type, encoding, calendar-validity of a date string	10.5%
Completeness / NOT NULL	presence assertions	9.5%
Functional dependency	$X \to Y$ within a table	6.3%
Inclusion dependency (IND)	foreign-key / referential integrity across tables	3.2%
Denial constraint (DC)	forbidden predicate combination not reducible to the categories above	1.1%

Table 12: Taxonomy of AI-mined rule types, tagged on a random 25% sample ($n = 85$) of all AI-mined rules.

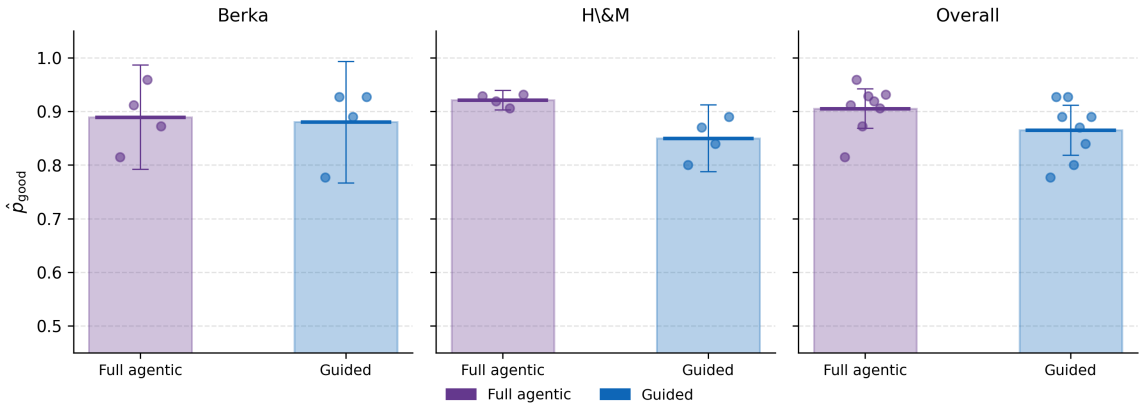

Figure 3: Good-rule rate $\hat{p}_{\text{good}}$ per mode and dataset (dots = individual runs, bar = mean, error bars = 95% CI-interval on mean).

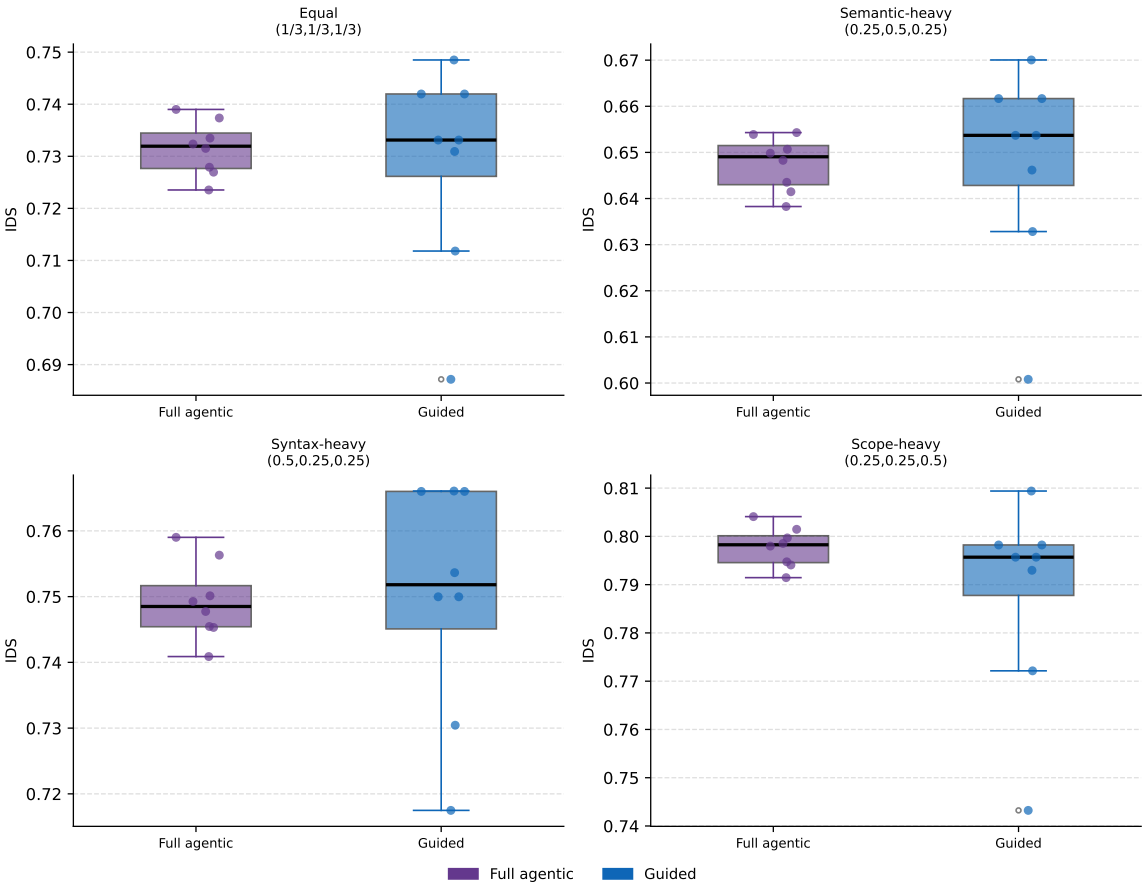

Figure 4: IDS distributions per mode across weight configurations ($n_1 = n_2 = 8$ runs per mode, pre-review rules).

Dataset	**Mode**	N_{rules}	N_{ratings}	Accept	Modify	Reject
Berka	Full agentic	216	432	363 (84.0%)	42 (9.7%)	27 (6.2%)
	Guided	101	202	168 (83.2%)	18 (8.9%)	16 (7.9%)
H&M	Full agentic	124	248	222 (89.5%)	13 (5.2%)	13 (5.2%)
	Guided	100	200	164 (82.0%)	12 (6.0%)	24 (12.0%)
Overall	Full agentic	340	680	585 (86.0%)	55 (8.1%)	40 (5.9%)
	Guided	201	402	332 (82.6%)	30 (7.5%)	40 (10.0%)

Table 13: HITL review outcomes per mode and dataset (pooled across 8 runs). N_{rules}: rules proposed; $N_{\text{ratings}} = 2N_{\text{rules}}$, since the blinded block design has every rule rated by exactly two reviewers who did not author it (§C). Accept/Modify/Reject are counts over ratings, with row percentages. Good-rule rates are reported in Table 2 as the mean across good rule rates of reviewer runs. The Full Agentic mode proposes more rules than Guided because the orchestrator enforces a per-column coverage criterion, whereas a human expert might target a narrower set of constraints. The matched-scope control in §5.4 re-runs the RQ1 analysis with this volume asymmetry removed.