A Value-Based Foreign Key Analysis Approach for Enriching Object-Centric Data Platforms

Clara Mattos Medeiros *André Santanchè*

Relatório Técnico - IC-PFG-25-57
Projeto Final de Graduação
2025 - Dezembro

UNIVERSIDADE ESTADUAL DE CAMPINAS
INSTITUTO DE COMPUTAÇÃO

A Value-Based Foreign Key Analysis Approach for Enriching Object-Centric Data Platforms

Clara Mattos Medeiros[*] André Santanchè[*]

Abstract

Large-scale enterprise ontologies often suffer from critical fragmentation, where logical connections between data entities remain unmapped due to siloed development and unreliable metadata. This paper presents a scalable, value-based methodology for the automatic discovery of implicit relationships within industrial big data platforms. To overcome the limitations of metadata-driven approaches, the proposed solution implements a foreign key analysis that directly examines data values. We introduce a novel confidence scoring model, incorporating source saturation, target coverage, and a Primary Key Randomness Score, to quantify the strength and validity of potential links while mitigating false positives from low-entropy identifiers. The algorithm was validated against a control group of existing relationships, achieving a recall rate of 72%, and subsequently identified thousands of previously unmapped connections, significantly enhancing the graph's connectivity. The methodology was further operationalized into a self-service governance application, empowering data teams to continuously enrich the enterprise data model and strengthen the integrity of the organizational digital twin.

1 Introduction

In the contemporary aerospace and manufacturing industries, the ability to optimize strategic decision-making through data is a critical competitive advantage. To address this, leading organizations have increasingly adopted large-scale Enterprise Data Platforms designed to consolidate and analyze vast amounts of heterogenous information.

At the heart of these ecosystems are curated data assets (often referred to as "data products")—certified sources that represent the organization's most reliable and complete information. While the quality of these individual assets is often high, their full value is only realized when they are integrated into a coherent conceptual framework. This framework is the Enterprise Ontology (or Knowledge Graph), an operational layer that formally defines business concepts as "objects" and maps the relationships between them. Ideally, this ontology functions as a comprehensive digital twin of the organization, providing a unified map for navigating the complex data landscape.

However, a significant architectural challenge persists in these environments: the ontology frequently remains incomplete. Many implicit logical connections present in the underlying data—for example, a shared identifier between an 'Aircraft' object and a 'Non-Conformity' report—are not formally declared in the semantic layer. This results in a fragmented data model where valuable entities appear isolated despite their inherent connections.

This disconnect arises from a gap between the strategic ambition for a unified ontology and the operational reality of its creation. In large, federated organizations, distinct teams often onboard new data domains in silos. While they may successfully expose their own datasets, procedural gaps often prevent them from identifying or implementing relationships with the wider ecosystem of existing objects.

[*]Instituto de Computação, Universidade Estadual de Campinas, 13081-970 Campinas, SP.

The consequences of this "semantic gap" are substantial. It hinders efficient data discovery, complicates the development of integrated applications, and, critically, limits the potential of advanced analytics and Artificial Intelligence (AI) systems. Modern AI architectures, particularly those leveraging Large Language Models (LLMs) or Graph Neural Networks rely heavily on a robustly connected ontology to understand complex scenarios and derive meaningful insights.

This paper addresses this specific gap. We present a methodology designed to systematically discover the "hidden" relationships between disjointed data assets. By automating the identification of these links, this approach aims to strengthen the structural integrity of the enterprise data model, making the digital twin more powerful, reliable, and fit-for-purpose.

2 State of the Art

The core challenge addressed in this paper—systematically discovering unmapped relationships within a large-scale data platform—sits at the intersection of several well-established fields in computer science and data management. While the term "ontology" is central to the target architecture, the problem shares characteristics with Ontology Alignment, Schema Matching, and Foreign Key Discovery. A review of these areas is crucial to contextualize why a value-based methodology was favored over purely semantic or AI-driven approaches.

2.1 Traditional Ontology Matching and Alignment

The primary goal of ontology matching is to find correspondences between semantically related entities belonging to different ontologies, a field comprehensively detailed by Euzenat and Shvaiko in their seminal book, *Ontology Matching* [1]. Foundational tools like PROMPT [2] were designed specifically for this kind of automated merging and alignment. Researchers have developed a wide array of techniques, which are typically classified into several categories:

- Linguistic matching: these methods analyze textual information like names and descriptions, often employing lexical databases like WordNet [3] to identify synonyms.
- Structural matching: these techniques analyze the graph structure of ontologies, using algorithms like Similarity Flooding to discover correspondences based on graph topology [4].
- Instance-based matching: this approach analyzes the actual data instances (the objects) that populate the ontologies to find evidence of a relationship [1].

However, the vast majority of this research focuses on resolving semantic heterogeneity between different knowledge bases. The challenge in the current industrial context is distinct: it involves enriching a single, centrally governed ontology. The problem is not one of semantic translation between two languages, but of achieving structural completeness within one.

2.2 Modern Approaches and the Role of AI

In recent years, the use of Large Language Models (LLMs) has gained significant traction for relationship discovery. Experimental studies, such as the 2024 paper by Parciak et al., are now rigorously evaluating the effectiveness of LLMs for schema matching tasks [8]. These models excel at capturing deep semantic similarities that simple string comparison would miss.

However, a purely AI-driven approach faces critical limitations in large-scale enterprise environments. As identified during the exploration phase of this research, LLMs are heavily dependent on high-quality metadata (column names and descriptions). In legacy industrial datasets, metadata is often sparse or ambiguous. Furthermore, AI models operate probabilistically and cannot verify actual data equivalence. This inability to guarantee that two fields can be technically joined leads to a high rate of false positives, which is unacceptable for automated governance.

2.3 Schema Matching and Foreign Key Discovery

The task is more directly addressed by the fields of schema matching and foreign key discovery, which have their roots in relational database management. In their influential 2001 survey, Rahm and Bernstein define schema matching as the process of identifying correspondences between elements of database schemas [5]. This project is a specific instance of Foreign Key Discovery, a long-standing problem in database engineering.

Industry-standard documentation from leaders like IBM outlines methodologies for “Foreign key analysis” which involves profiling data and validating candidates by checking for inclusion dependencies [6]. This practical approach is grounded in the formal definition of an inclusion dependency found in foundational database literature, such as the work of Garcia-Molina, Ullman, and Widom [7]. The “saturation” and “coverage” metrics developed in this project are a direct, practical implementation of this core database principle.

Therefore, the methodology developed in this project deliberately pivots away from traditional ontology alignment and purely AI-driven semantic analysis. Instead, it grounds itself in the robust principles of instance-based schema matching and foreign key discovery [5] [6] [7]. By focusing on the direct analysis of data values, the project adopts a pragmatic and deterministic approach that prioritizes finding relationships that are not just conceptually plausible but are immediately implementable within the technical constraints of the platform.

3 Problem Definition and System Architecture

3.1 The Data Governance Framework

This research was conducted within a large-scale enterprise data platform designed to function as the “digital twin” of a global industrial organization. The platform serves as a semantic backbone, ingesting raw data from heterogeneous sources and mapping it into a unified, ontology-based model .

The data architecture follows a strict governance hierarchy designed to transform raw signals into meaningful business concepts:

- Business Objects: At the conceptual level, the organization defines Business Objects, which describe core entities (e.g., an ’Aircraft’ or ’Work Order’) independent of any specific IT system.
- Data Products: These concepts are physically implemented via Data Products. These are curated, governed, and certified assets that contain the validated datasets necessary to describe a Business Object.
- Object Types: Finally, to be represented in the interactive user layer, data from Data Products is exposed as Object Types in the ontology. These Object Types act as the nodes in the enterprise knowledge graph.

The formal relationship between these core data governance concepts is illustrated in Figure 1.

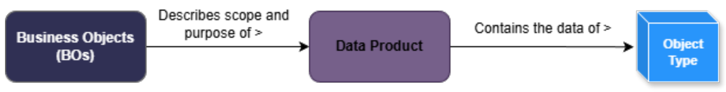

Figure 1: The main data governance concepts.

3.2 The Semantic Gap

Despite this structured governance, a significant "semantic gap" exists between the stored data and its ideal ontological representation. While the system contains hundreds of distinct Object Types, the logical connections between them are frequently missing. This fragmentation stems from three primary causes:

1. Denormalization: To optimize query performance, Object Types are often created as flattened, denormalized copies of the underlying datasets. While this simplifies data access, it obscures the underlying normalized relationships.

2. Organizational Silos: The responsibility for creating Data Products is distributed across federated domain teams. These teams often lack visibility into Object Types created by other domains. Consequently, a team creating a new object may not realize that a linkable entity already exists in the ontology, resulting in unmapped relationships.

3. Metadata Ambiguity: Attribute naming within datasets is notoriously inconsistent (e.g., using generic names like id or ref instead of descriptive business keys). This ambiguity makes it difficult to reliably infer connections based on metadata alone.

3.3 Objectives

To address this fragmentation, this study aimed to design a scalable methodology to systematically identify these "hidden" relationships between Object Types. The specific objectives were to:

1. Develop an automated discovery algorithm that analyzes the physical values within datasets rather than relying on inconsistent metadata.

2. Create a confidence scoring model to quantify the strength of potential links between Object Types.

3. Validate the findings against a control group of existing relationships to ensure precision before scaling the analysis to the entire enterprise ontology.

4 Methodology and development

4.1 Exploration of Alternative Approaches

To address the challenge of ontological fragmentation and systematically identify missing relationships between Object Types, this research evaluated two distinct methodological strategies: a "Top-Down" approach driven by explicit business knowledge, and a "Bottom-Up" approach driven by statistical data analysis.

The ultimate methodology adopted in this study is a hybrid model, leveraging the scalability of the bottom-up approach while using top-down business concepts for validation

4.1.1 Top-Down Approach: Leveraging Explicit Business Knowledge

The initial phase of research investigated a traditional "Top-Down" strategy. This approach relies on leveraging pre-existing, explicit knowledge encoded in documentation, data dictionaries, and the expertise of domain specialists to identify potential links between Business Objects.

The primary mechanism for this approach involves manually traversing the organization's Enterprise Data Catalog. A Data Catalog serves as a centralized repository of metadata, offering descriptions of datasets, definitions of business terms, and sometimes theoretical models of how concepts should relate. By searching for common business identifiers (e.g., “Customer ID,” “Part reference Number”) within these catalogs, one can hypothesize connections between different Data Products.

Limitations of the Top-Down Approach While theoretically sound, investigations revealed significant practical limitations to applying this approach at scale in a large, federated enterprise:

- Scalability and Manual Effort: The process of cross-referencing thousands of potential attribute pairs across hundreds of Data Products is highly manual and resource-intensive. It is not feasible for continuous, automated governance.
- Documentation Reliability: Enterprise Data Catalogs are often subject to the same siloing issues as the data itself. Documentation is frequently outdated, incomplete, or inconsistent across different business domains. Relying on static documentation to describe a dynamic, rapidly evolving data ecosystem proved insufficient.
- Metadata vs. Data Reality: A theoretical link described in a catalog does not guarantee that the actual physical data within two datasets is compatible so that a join can be performed.

Consequently, while business knowledge remains crucial for final validation, a purely top-down, documentation-driven approach was determined to be inadequate for large-scale, automated relationship discovery. This necessitated the development of a data-driven “Bottom-Up” methodology.

4.1.2 AI-Driven Approach: Semantic Metadata Analysis

In parallel with the top-down strategy, the research explored a computational approach leveraging Artificial Intelligence, specifically Large Language Models (LLMs). Given the semantic nature of the problem, utilizing generative AI to infer connections based on textual metadata appeared to be a logical path forward .

The proposed methodology involved feeding the schemas of all Object Types (including attribute names, descriptions, and data types) into an LLM. The model was tasked with identifying semantically similar attributes that could theoretically represent a primary key-foreign key pair. For instance, the model would attempt to deduce that an attribute named asset_identifier in one object and serial_number in another refer to the same real-world entity .

Despite its theoretical appeal, experimental validation revealed that a purely AI-centric approach was unreliable for automated governance in this specific industrial context. Three primary failure modes were identified:

- Poor Quality of Input Metadata: LLMs rely on rich, descriptive text to function effectively (“garbage in, garbage out”). In the target environment, attribute descriptions were frequently missing or populated with generic technical codes (e.g., id, ref, code_1). Without sufficient semantic context, the model’s ability to hallucinate or miss connections proved too high a risk.
- Semantic Equivalence vs. Data Equivalence: A fundamental distinction exists between two attributes meaning the same thing and two attributes being joinable. As illustrated in Figure 2, an LLM might correctly identify that a flight_id and a mission_code refer to the same concept. However, if one is stored as a composite string (e.g., “FLT-2023-998”) and the other as an integer (998), a direct relationship implementation is impossible without intermediate transformation pipelines. An AI operating solely on metadata generates “false positive” actionable links that fail upon implementation .

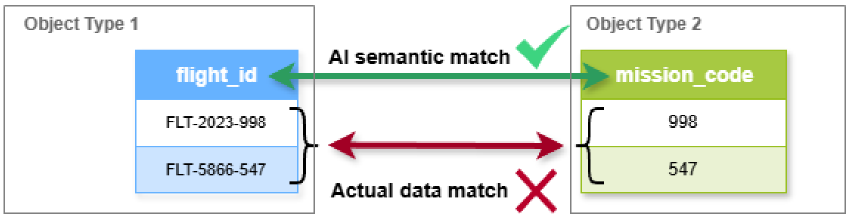

Figure 2: AI‘s limitation: semantic match vs. actual data match.

- Computational Inefficiency: A brute-force AI analysis requires a pairwise comparison of every attribute across the entire ontology. With hundreds of objects and thousands of attributes, this results in a combinatorial explosion. The computational cost and latency of millions of LLM inference calls were deemed unjustifiable when compared to deterministic methods leveraging the platform's native indexing engine .

It is important to note that while AI was deemed unsuitable as the primary discovery engine in this low-metadata context, it holds significant potential as an auxiliary layer in a hybrid architecture. Future work could explore using LLMs as a pre-filtering mechanism to narrow the search space before executing expensive data scans, or conversely, as a post-validation step to semantically verify the technical matches found by deterministic algorithms. However, these hybrid integrations remained outside the scope of this specific project, which prioritized a deterministic Value-Based Foreign Key Analysis.

4.2 Proposed Methodology: Value-Based Foreign Key Analysis

Following the evaluation of alternative strategies, this research implemented a "Bottom-Up" Foreign Key Analysis. Unlike metadata-driven approaches, this methodology grounds its discovery process in the most reliable resource available: the actual data values stored within the Object Types.

This approach offers a deterministic and scalable method for uncovering relationships with a high degree of confidence. The solution was architected as a fully automated pipeline within the enterprise's distributed computing environment, leveraging server-side APIs to efficiently query the ontology's underlying index structures.

4.2.1 Architecture of the Identification Engine

The discovery engine is designed for scalability and minimal manual intervention. The architecture is divided into two primary phases: an automated daily preparation phase and the core analysis phase.

4.2.1.1 Phase 1: Automated Data Indexing and Sampling

To decouple the heavy data-gathering task from the complex analysis, a preparatory pipeline runs daily to establish a "Global Key Index."

1. Cataloging: The system scans the platform to generate a master registry of all active Data Products and Object Types.

2. Primary Key Sampling: For every identified Object Type, the system programmatically retrieves a random sample of up to 1,000 primary key (PK) values. This creates a unified, lightweight reference dataset containing representative keys for every entity in the enterprise. This sample serves as the "fingerprint" used to probe for connections .

This ensures the main algorithm can run efficiently without needing to query billions of records for every execution. The flowchart in Figure 3 describes this process.

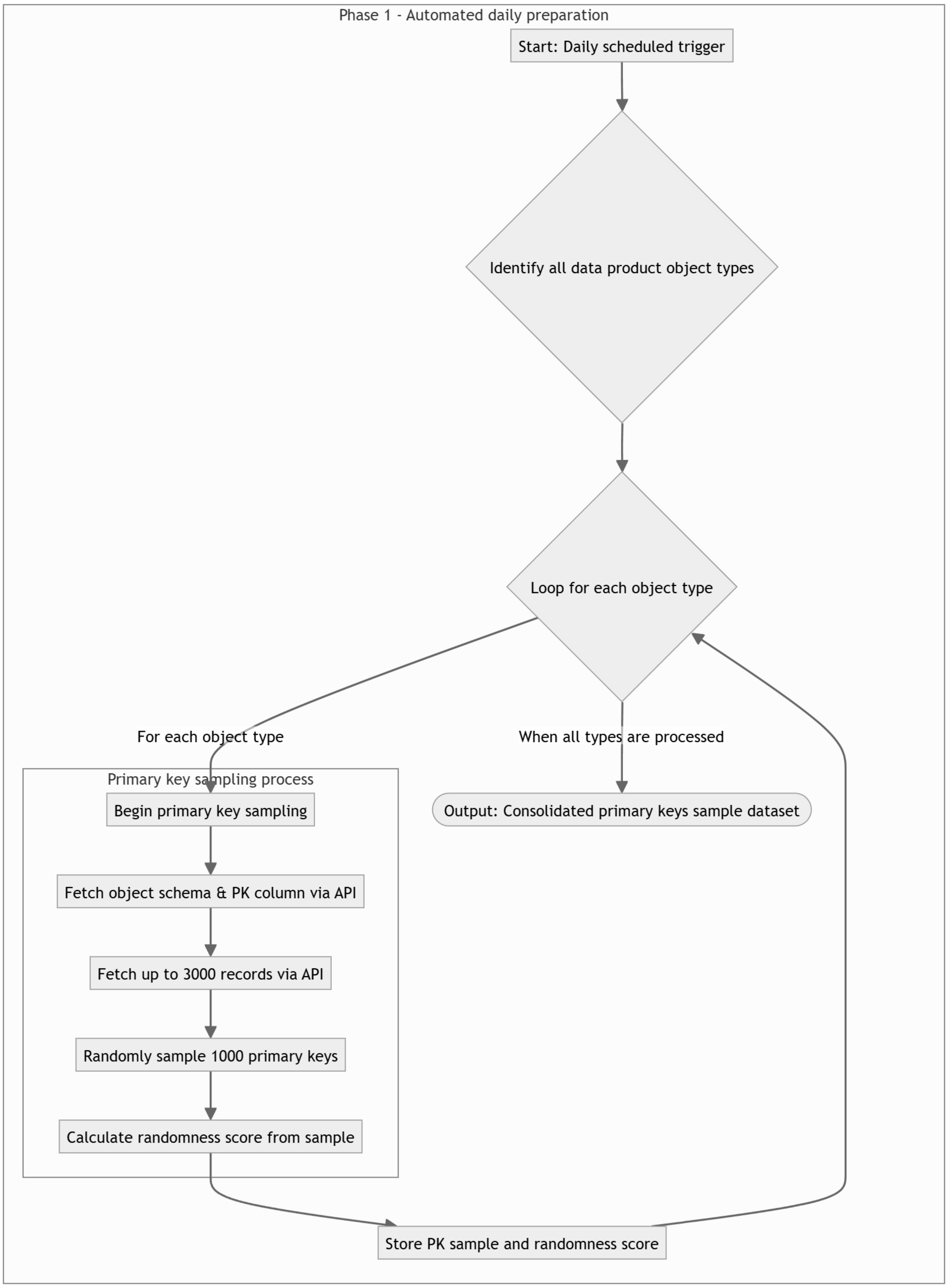
Phase 1 - Automated daily preparation
Start: Daily scheduled trigger
Identify all data product object types
Loop for each object type
For each object type
When all types are processed
Primary key sampling process
Begin primary key sampling
Fetch object schema & PK column via API
Fetch up to 3000 records via API
Randomly sample 1000 primary keys
Calculate randomness score from sample
Output: Consolidated primary keys sample dataset
Store PK sample and randomness score

Figure 3: Exemplification of the daily data preparation in flowchart form.

4.2.1.2 Phase 2: Core Relationship Analysis

The analysis pipeline employs a highly parallelized, two-stage process designed to validate potential relationships without incurring the computational cost of full-table scans.

The first stage is responsible for identifying the first candidates for an attribute match. To do this, instead of a brute-force comparison ($O(n^2)$), the algorithm utilizes the platform's Inverted Index. The system takes the sample of primary keys from a potential "Target" object and executes a batch lookup across all attributes of a "Source" object.

The algorithm adapts the search based on data types. For instance, if the target keys are strings, the system implicitly attempts to find numerically compatible matches within integer columns of the source. This robust type-handling significantly increases recall by bridging the gap between inconsistent data formats. Any attribute in the Source that contains a subset of the Target's keys is flagged as a "Candidate Attribute."

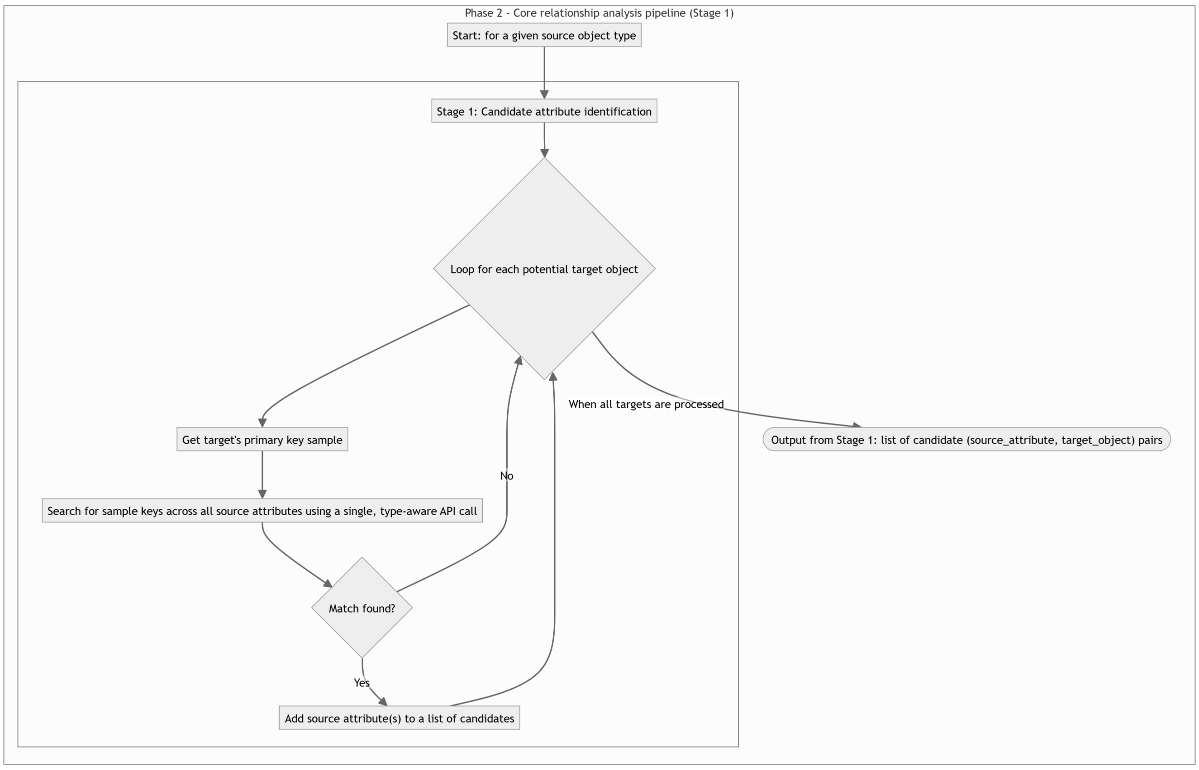
Phase 2 - Core relationship analysis pipeline (Stage 1)
Start: for a given source object type
Stage 1: Candidate attribute identification
Loop for each potential target object
When all targets are processed
Output from Stage 1: list of candidate (source_attribute, target_object) pairs
Get target's primary key sample
Search for sample keys across all source attributes using a single, type-aware API call
Match found?
No
Yes
Add source attribute(s) to a list of candidates

Figure 4: Exemplification of the first stage of the core relationship analysis in flowchart form.

Once a candidate attribute is identified, the second stage begins and the system validates the relationship volume. To ensure performance at the petabyte scale, the architecture prioritizes Server-Side Aggregation.

- Primary Path (Exact Calculation): The algorithm instructs the database engine to group the source attribute and count occurrences, returning only the aggregated metrics. These keys are then validated against the target's full primary key index .
- Fallback Path (Adaptive Sampling): For "mega-objects" where high cardinality prevents full aggregation within timeout limits, the system automatically switches to a statistically significant sampling method. It retrieves a fixed subset (e.g., 10,000 rows) and extrapolates the connection metrics, flagging the result as an approximation .

The feasibility of this approach relies on the strategic use of the platform's indexing capabilities. In document-oriented data architectures, attributes are often indexed using an Inverted Index, mapping specific values to the list of documents containing them . By querying

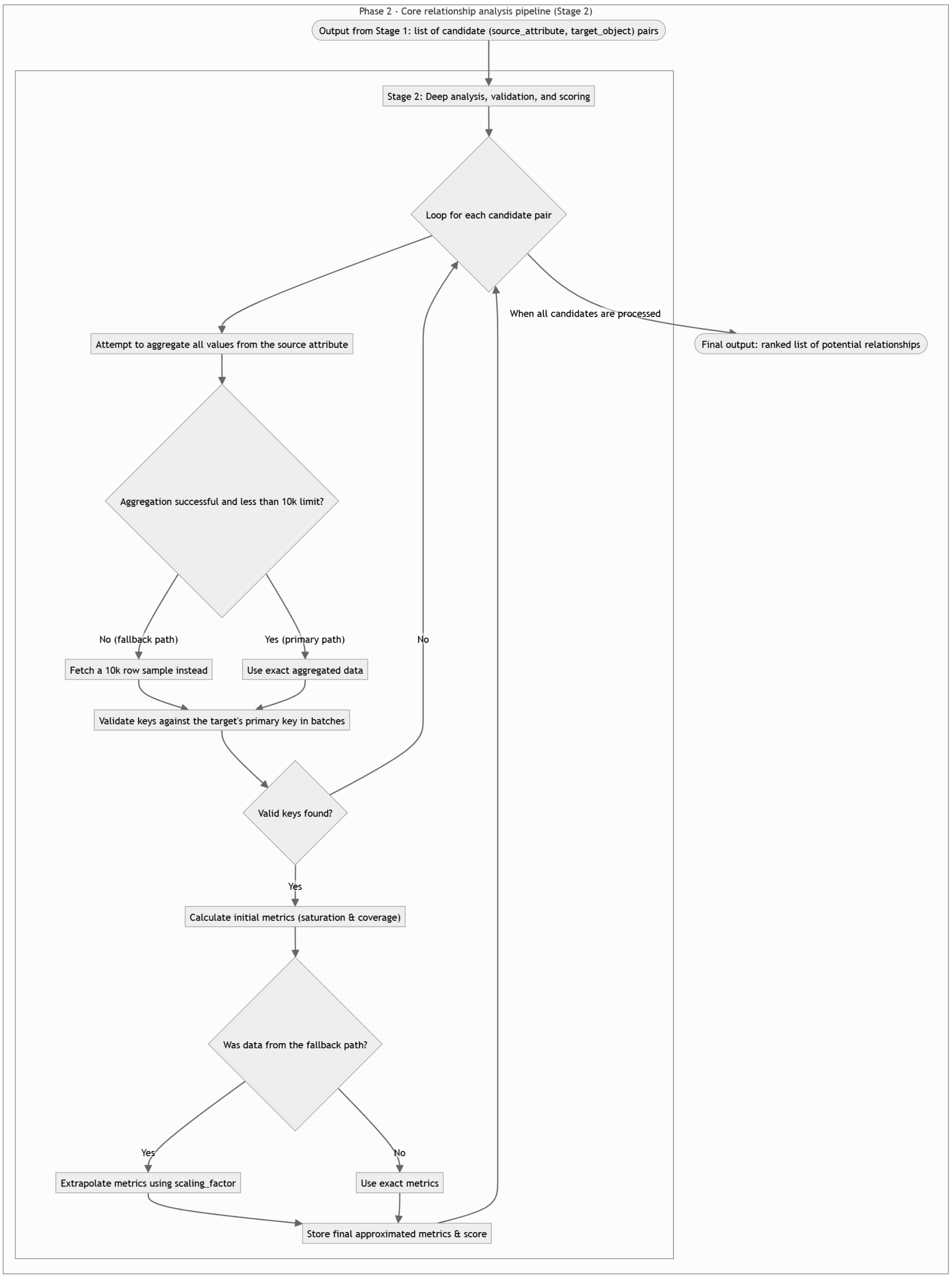
Phase 2 - Core relationship analysis pipeline (Stage 2)
Output from Stage 1: list of candidate (source_attribute, target_object) pairs
Stage 2: Deep analysis, validation, and scoring
Loop for each candidate pair
When all candidates are processed
Final output: ranked list of potential relationships
Attempt to aggregate all values from the source attribute
Aggregation successful and less than 10k limit?
No (fallback path)
Yes (primary path)
No
Fetch a 10k row sample instead
Use exact aggregated data
Validate keys against the target's primary key in batches
Valid keys found?
Yes
Calculate initial metrics (saturation & coverage)
Was data from the fallback path?
Yes
No
Extrapolate metrics using scaling_factor
Use exact metrics
Store final approximated metrics & score

Figure 5: Exemplification of the second stage of the core relationship analysis in flowchart form.

this index directly (via "Is In" clauses) rather than performing full table scans, the discovery algorithm achieves near-constant time complexity for candidate identification. Furthermore, utilizing server-side aggregations minimizes network overhead by transmitting only summary statistics rather than raw data. This "compute-at-storage" strategy is what allows the methodology to scale to an enterprise ontology containing billions of records .

4.2.2 The Confidence Scoring Model

A binary "yes/no" result is insufficient for decision support. To make the results actionable, a multi-dimensional scoring model was developed to quantify the strength of each discovery.

4.2.2.1 Core Metrics: Saturation and Coverage

For every validated link, two fundamental database metrics are calculated :

- Source Saturation (%): "Of all objects in the source, what percentage link to a valid target?" A high saturation implies a mandatory or structural relationship (e.g., a "Foreign Key" constraint).
- Target Coverage (%): "Of all known target objects, what percentage are referenced by the source?" This indicates the breadth of the relationship.

(a) Step 1: aggregate candidate values from source ('Non-Conformity'). Total objects = 50.

aircraft_id	**Count**
101	18
102	5
999	4
103	12
535	3
104	8

(b) Step 2: validate against target primary keys ('Aircraft'). Total objects = 200.

msn (PK)
101
102
103
104
...

(c) Step 3: calculate metrics from validated links.

Validated Key	**Count from Source**
101	18
102	5
103	12
104	8

- Total source objects with valid link: $18 + 5 + 12 + 8 =$ **43**
- Total distinct target objects referenced: **4**
- Source saturation: $(43/50) \times 100 =$ **86**%
- Target coverage: $(4/200) \times 100 =$ **2**%

Figure 6: Visual example of the relationship validation and scoring Process.

4.2.2.2 The Primary Key Randomness Score

A critical innovation of this methodology is the Primary Key (PK) Randomness Score. A major risk in foreign key discovery is "coincidental matching"—for example, an object using sequential integers (1, 2, 3...) matching unrelated integer columns in other datasets. To mitigate this, the system calculates an entropy-based score for every Object Type's primary key, analyzing:

- Length: longer keys are inherently less likely to be matched by coincidence. The longer a primary key is, the bigger their randomness score is.
- Character diversity: the score increases if the key contains a mix of character types.
- Entropy: the model penalizes simple or repetitive patterns (e.g., "11111“ or "ABCDE").

This allows the system to discount matches involving low-entropy keys (like simple indices) and prioritize matches involving complex, high-entropy identifiers (like UUIDs or VINs), which are statistically impossible to match by chance .

For an in-depth understanding of the randomness score calculation method, refer to Listing 1 in the Appendices.

4.2.2.3 The Weighted Confidence Score

The final confidence score is a weighted average of these two percentages, calculated as:

$$Score = (0.75 \times max(Saturation, Coverage)) + (0.25 \times min(Saturation, Coverage))$$

This formula intelligently prioritizes the most common types of business relationships. By giving more weight to the stronger side of the link (either saturation or coverage), it effectively highlights one-to-many relationships (e.g., one aircraft having many non-conformities), which are often the most valuable for creating a connected data model. A simple average would fail to distinguish between a balanced 50%/50% link and a more significant 90%/10% one-to-many link. This scoring mechanism ensures that the most structurally important relationships rise to the top of the results.

Applying this formula to the example shown in Figure 6, we can see its practical benefit. In that case, the Source Saturation was 86% and the Target Coverage was 2%. A simple average would yield a score of $\frac{(0.86+0.02)}{2} = 44\%$. This score is modest and does not fully represent the strength of the link. However, the weighted formula calculates the score as $(0.75 * 0.86) + (0.25 * 0.02) = 65\%$. This score, when viewed alongside the target's Primary Key Randomness Score, provides the user with a comprehensive and reliable assessment of the potential relationship.

5 Results and Achievements

The methodology was applied to the full scope of the enterprise ontology, comprising hundreds of Object Types. The results were analyzed to assess both the algorithmic performance (validation against ground truth) and the structural insights revealed about the data ecosystem.

5.1 Performance Validation

To quantify the reliability of the discovery engine, the algorithm was first tested against a control group of 223 relationships that had been previously manually implemented by domain teams.

- Recall Rate: The algorithm successfully rediscovered 160 (72%) of these known links. This confirms that the majority of implemented relationships rely on direct value matches that the tool successfully detects.

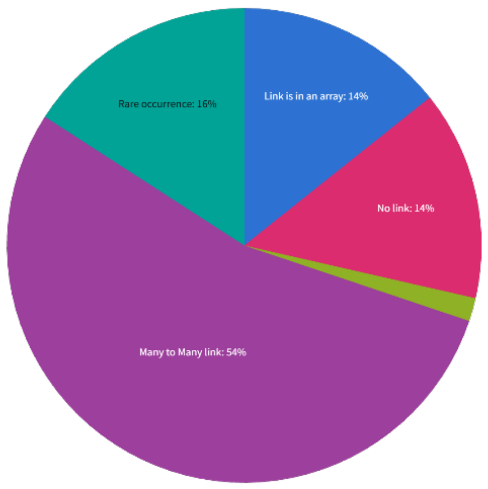

Figure 7: Different reasons for not identification of existing skywise links.

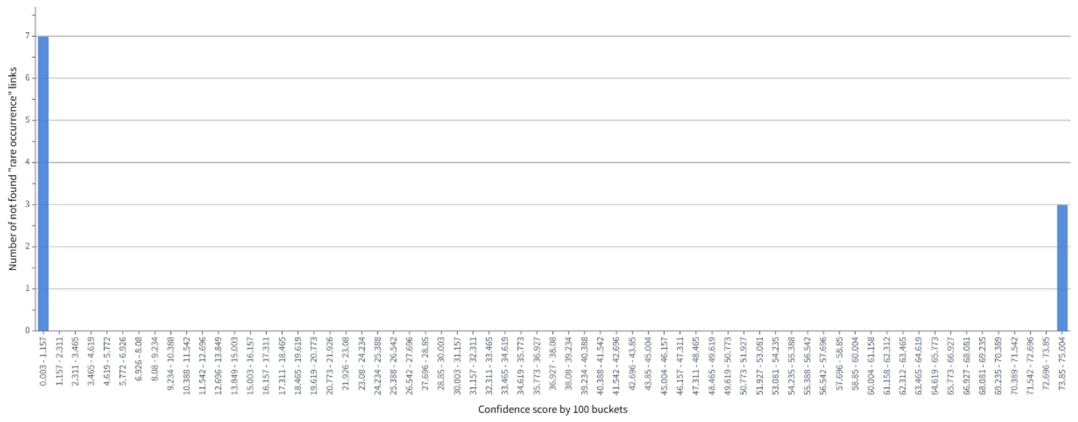

Figure 8: Distribution of the manually calculated confidence scores for the 10 links missed due to sampling limitations.

- Analysis of Missed Links: An investigation into the 63 missed links revealed clear boundaries of the methodology.
  - The majority (54%) were "Many-to-Many" relationships, which utilize an intermediary join table. Since the algorithm was optimized for direct equality checks to ensure scalability, these indirect links were intentionally out of scope.
  - Ten links were missed because they were statistically rare. The fixed 1,000-key sample taken from a very large target Object Type did not happen to include the specific keys involved in the relationship. A manual calculation of the confidence scores for these 10 missed links (Figure 8) revealed that 7 of them had scores below 1.1%, meaning they were statistically insignificant connections. The three high-scoring misses occurred in a specific scenario: a very small source object where nearly 100% of its records linked to a massive target. This analysis confirms that while the sampling strategy can miss links, it primarily affects those of very low significance.
  - Other misses were attributed to complex data types (e.g., keys stored inside arrays), which was left outside of the scope of this project, or the link was manifested but didn't actually exist (true negative).

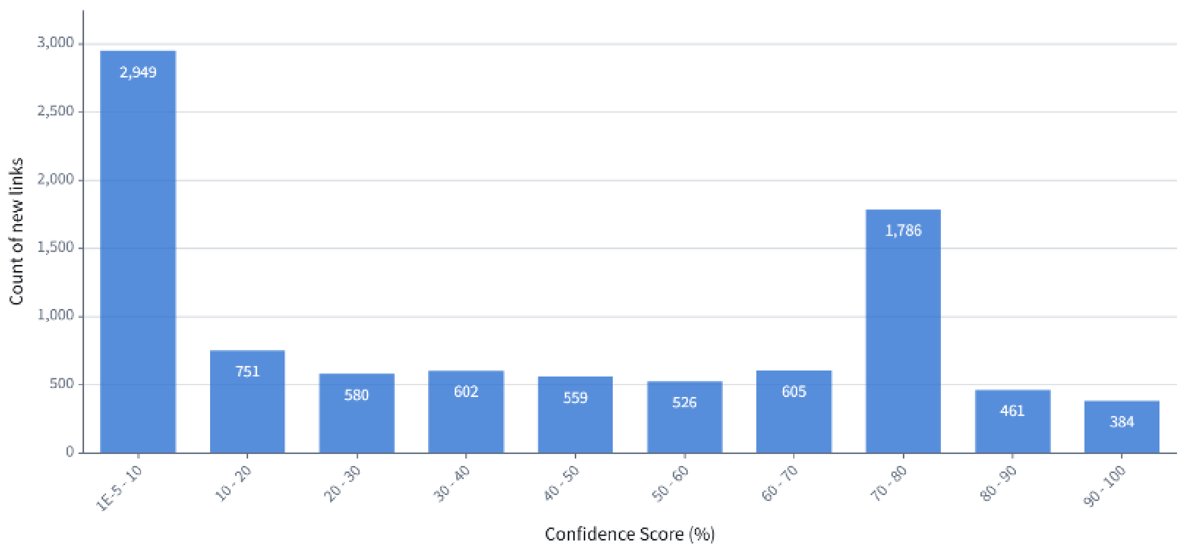

Figure 9: Distribution of confidence scores across all newly discovered potential relationships.

5.1.1 Large-Scale Discovery and Quality Assessment

Upon executing the analysis across the entire platform, the engine identified a total of 9,203 new potential relationships. To assess the quality of these findings, we analyzed the distribution of the confidence scores.

The results exhibited a strong bimodal distribution (see Figure 9). Potential links tended to cluster either in the very low range (0–10%) or the high range (70–80%), suggesting a clear distinction between random data noise and genuine structural connections .

5.1.2 The Impact of PK Randomness

The integration of the Primary Key Randomness Score proved decisive in filtering false positives. As illustrated in the heat map in Figure 10, we observed a "Noise Floor" of links with low confidence and low key randomness (e.g., coincidental matches of integers like 1, 2, 3). Conversely, the "Gold Standard" quadrant—links with high confidence and high key complexity—contained over 2,600 high-potential candidates. This multi-dimensional filtering allowed us to isolate ∼380 "golden candidates" for immediate implementation.

5.1.3 Structural Insights: Identification of Data Hubs

Beyond individual links, the aggregate analysis revealed the "center of gravity" of the enterprise data model. By mapping the targets of the discovered relationships, clear patterns emerged regarding data dependencies.

- Foundational Domains: The "Design" and "Compliance" domains emerged as the primary data hubs, accounting for over 87% of all discovered connections.
- Master Data vs. Transactional Data: The analysis highlighted a distinct architectural pattern. Transactional objects (e.g., "Inspection Events," "Operational Logs") consistently referenced a small set of foundational Master Data objects (e.g., "Master Equipment List," "Quality Planning Framework").

This insight provides a data-driven roadmap for governance. It suggests that ensuring the quality of these few "Master Data" objects yields the highest return on investment, as they serve as the semantic anchors for thousands of peripheral datasets .

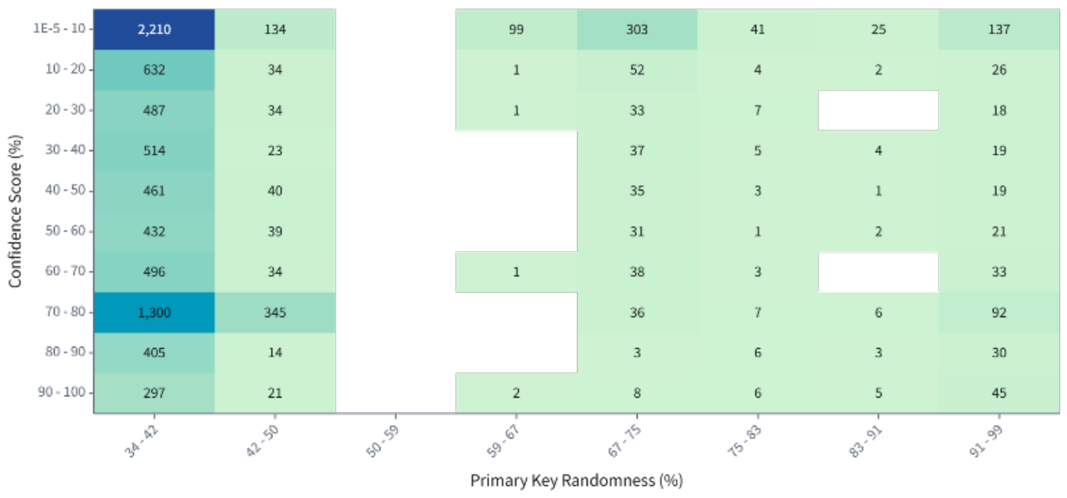

Figure 10: Heat grid illustrating the relationship between confidence score and Primary Key Randomness.

6 Conclusion and Future Work

6.1 Summary of Contributions

This research addressed the critical challenge of semantic fragmentation inherent in large-scale, federated enterprise data platforms. We demonstrated that while top-down governance and metadata-driven approaches are necessary foundations, they are insufficient for maintaining ontological completeness at an industrial scale.The primary contribution of this study is a scalable, "bottom-up" methodology for automated Foreign Key Discovery that pivots away from unreliable metadata to leverage direct value analysis. By introducing a novel, multi-dimensional scoring framework—incorporating Source Saturation, Target Coverage, and the entropy-based Primary Key Randomness Score—we provided a robust mechanism to distinguish genuine structural links from coincidental data noise.The operational results confirmed the hypothesis of a significant "semantic gap." The methodology validated its precision with a $\sim$72% recall rate against established ground truth and subsequently revealed the extent of the ontology's incompleteness by identifying thousands of high-confidence, previously unmapped relationships. Furthermore, the aggregate analysis successfully identified the organization's foundational Master Data hubs, providing a data-driven roadmap for future governance efforts.

6.2 Limitations of the Methodology

Despite its success in uncovering direct relationships, the proposed methodology has inherent limitations driven by its deterministic design and architectural constraints designed for scalability.

- Dependency on Ingestion Quality: As a bottom-up approach, the algorithm's effectiveness is fundamentally bounded by the quality of the physical data. If source systems fail to preserve referential integrity—for example, through corrupted keys, inconsistent formatting, or truncated values during ingestion—the algorithm cannot detect the relationship. It is a tool for discovery, not data cleansing.

- Scope of Relationship Types: To ensure performance across petabytes of data, the current implementation was strictly limited to direct, scalar comparisons (e.g., string-to-string, integer-to-integer). Consequently, complex relationships involving keys nested within arrays, composite keys requiring concatenation, or indirect many-to-many relationships utilizing intermediary join tables remain outside the current detection capabilities.

6.3 Future Directions

Future research should focus on addressing these limitations and bridging the gap between deterministic structure and semantic meaning. Two primary avenues are proposed:

1. Hybrid AI Validation: While AI was rejected as a primary discovery engine due to metadata quality issues, it holds significant promise as a post-validation layer. A hybrid architecture could utilize the value-based engine as a high-recall candidate generator, followed by an LLM-based semantic analysis to verify if the technically valid link makes "business sense" in the real world.

2. Advanced Indexing for Complex Types: Evolving the indexing strategy to handle complex data structures—such as "exploding" array columns into searchable rows during the indexing phase—would allow the algorithm to extend its reach to more sophisticated relationship patterns without sacrificing the efficiency of inverted index querying.

References

[1] Euzenat, J., & Shvaiko, P. (2013). *Ontology Matching.* Springer-Verlag Berlin Heidelberg.

[2] Noy, N. F., & Musen, M. A. (2000). *PROMPT: Algorithm and tool for automated ontology merging and alignment.* In *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-00).*

[3] Fellbaum, C. (Ed.). (1998). *WordNet: An Electronic Lexical Database.* MIT Press.

[4] Melnik, S., Garcia-Molina, H., & Rahm, E. (2002). *Similarity flooding: A versatile graph matching algorithm and its application to schema matching.* In *Proceedings of the 18th International Conference on Data Engineering (ICDE).*

[5] Rahm, E., & Bernstein, P. A. (2001). *A survey of approaches to automatic schema matching.* The VLDB Journal, 10(4), 334-350.

[6] IBM. (2021). *Foreign key analysis.* IBM InfoSphere Information Server. Retrieved August 18, 2025, from `https://www.ibm.com/docs/en/iis/11.7.0?topic=relationships-foreign-key-analysis`

[7] Garcia-Molina, H., Ullman, J. D., & Widom, J. (2008). *Database systems: The complete book.* Pearson Education.

[8] Parciak, M., Vandevoort, B., Neven, F., Peeters, L. M., & Vansummeren, S. (2024). *Schema Matching with Large Language Models: an Experimental Study.* arXiv preprint arXiv:2407.11852. Retrieved August 18, 2025, from `https://arxiv.org/abs/2407.11852`

[9] Heise. (2025). *Missing-link Power center: Palantir, a software controls organizations.* heise online. Retrieved June 6, 2025, from `https://www.heise.de/en/background/Missing-link-Power-center-Palantir-a-software-controls-organizations-10463503.html`

7 Appendices

7.1 Primary Key Randomness Score calculation code

```python


def calculate_set_randomness(primary_keys):
    """
    Calculates a single randomness score for a set of primary keys,
    prioritizing length and character mixing, normalized to 0-1.
    """

    if not primary_keys:
        return 0

    total_score = 0
    for pk in primary_keys:
        total_score += calculate_single_key_randomness(pk)

    return total_score / len(primary_keys)


def calculate_single_key_randomness(pk):
    """
    Calculates the randomness score for a single primary key.
    Prioritizes length and the presence of mixed character types.
    Normalizes the final score to the 0-1 range.
    """

    pk = str(pk)
    max_len = 15.0  # Approximate max length

    len_pk = len(str(pk))
    length_factor = min(1.0, len_pk / max_len)

    char_diversity = character_set_diversity_score(pk)
    entropy_factor = character_frequency_entropy(pk)

    # Combine factors
    raw_score = 0.6 * length_factor + 0.4 * (char_diversity + entropy_factor)

    return min(1, raw_score/1.4) # Approximate normalization


def character_set_diversity_score(pk):
    """
    Calculates a score based on the diversity of character types.
    """

    has_digit = any(c.isdigit() for c in pk)
    has_letter = any(c.isalpha() for c in pk)
    has_symbol = any(not c.isalnum() for c in pk)

    diversity_count = sum([has_digit, has_letter, has_symbol])
    return diversity_count / 3.0


def character_frequency_entropy(pk):
    """
    Calculates the entropy of character frequencies, prioritizing mixed types.
    Penalizes digit-only strings based on length, but less harshly.
    """
    has_digit = any(c.isdigit() for c in pk)
    has_letter = any(c.isalpha() for c in pk)
    has_symbol = any(not c.isalnum() for c in pk)

    if has_digit and not has_letter and not has_symbol:  # Only digits
        len_pk = len(str(pk))
        return max(0.2, 0.8 - len_pk / 15.0)  # Less severe length penalty
    elif has_digit or has_letter or has_symbol:
        char_counts = Counter(pk)
        total_chars = len(str(pk))
        entropy = 0
        for count in char_counts.values():
            p = count / total_chars
            entropy -= p * math.log2(p)

        # Normalize entropy
        max_entropy = math.log2(len(set(pk)))
        if max_entropy > 0:
            return entropy / max_entropy
        else:
            return 0.3  # Base for mixed types
    else:
        return 0.3  # Base for only symbols
```

Listing 1: Python code for calculating the Primary Key Randomness Score.