Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

A Task-Regime Perspective on Zero-Knowledge Database Migration

Dr. Kirti Wanjale

kirti.wanjale@vit.edu

Vishwakarma Institute of Technology

Aadi Joshi

toaadijoshi@gmail.com

Vishwakarma Institute of Technology

Kavya Bhand

kavya.bhand0806@gmail.com

Vishwakarma Institute of Technology

Kabir Khanuja

Vishwakarma Institute of Technology

Research Article

Keywords: Database Migration, Graph Neural Networks, Relational Reasoning, Schema Learning, Zero-Knowledge Databases

Posted Date: March 25th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-9205841/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: The authors declare no competing interests.

A Task-Regime Perspective on Zero-Knowledge Database Migration

Dr. Kirti Wanjale
Dept. of Computer Engineering,
Vishwakarma Institute of Technology
Pune, India.
kirti.wanjale@vit.edu

Aadi Joshi
Dept. of Computer Engineering,
Vishwakarma Institute of Technology
Pune, India.
toaadijoshi@gmail.com

Kavya Bhand
Dept. of Computer Engineering,
Vishwakarma Institute of Technology
Pune, India.
kavya.bhand0806@gmail.com

Kabir Khanuja
Dept. of Computer Engineering,
Vishwakarma Institute of Technology
Pune, India.
kabirkhanuja@gmail.com

***Abstract* —** Relational schemas are naturally graph-structured, motivating the widespread use of Graph Neural Networks (GNNs) for database migration tasks such as foreign key discovery, integrity validation, and impact analysis. However, we show that treating all schema learning problems as monolithic graph tasks conflates fundamentally different computational regimes. We introduce a principled taxonomy that decomposes migration tasks into local tasks, solvable from pairwise column features, and relational tasks, requiring multi-hop structural reasoning. Across three datasets - Spider (166 academic databases), SchemaPile (real-world GitHub schemas), and Stack Overflow (enterprise-scale schema) - we conduct 11 controlled experiments comparing MLPs, multiple GNN architectures, DeepSets, SQL baselines, and classical heuristics. We find that: (1) For local tasks such as foreign key discovery, a simple MLP on pairwise column features outperforms GNNs by up to +129% in F1 ($p < 0.001$), as message passing injects structural noise from non-FK neighbors. (2) For relational tasks such as propagation count and blast-radius estimation, GNNs are indispensable, achieving $R^2 > 0.99$, while MLPs plateau at $R^2 \approx 0.73$. (3) Under partial-access enterprise workflows (40 - 60% schema hidden), GNNs degrade gracefully (MAE $\approx$ 1.1), whereas SQL traversal fails catastrophically. We further formalize structural noise injection, proving that message passing reduces signal-to-noise ratio in sparse-FK schemas, explaining observed over-smoothing effects. Based on these findings, we propose a regime-aware hybrid pipeline that assigns inductive bias per task type and strictly dominates monolithic approaches. All experiments are fully reproducible, with code and results publicly released.

***Keywords* —** *Database Migration, Graph Neural Networks, Relational Reasoning, Schema Learning, Zero-Knowledge Databases*

1. Introduction

Modern enterprises operate on complex relational databases that evolve over years of iterative development, personnel turnover, and system integration. When such systems must be migrated, whether to a new storage engine, cloud environment, or data warehouse, engineers face a recurring

obstacle: schema metadata is incomplete, inconsistent, or entirely missing. Foreign key (FK) constraints may never have been declared; naming conventions are irregular; and referential integrity violations accumulate silently. In these zero-knowledge migration settings, reconstructing latent schema structure becomes a prerequisite for safe transformation.

Relational schemas are inherently graph-structured: tables and columns form nodes, and relationships induce edges. This structural property has motivated the increasing adoption of Graph Neural Networks (GNNs) for schema-related tasks, including schema matching, cardinality estimation, and join optimization. The implicit assumption is that because schemas are graphs, learning over schemas should employ relational message passing.

While schemas are graph-structured, not all schema learning tasks require relational reasoning. Some tasks depend only on local compatibility between two columns (e.g., whether orders.user_id references users.id). Others require multi-hop traversal of dependency chains (e.g., estimating the downstream impact of dropping a table). Treating both regimes uniformly with graph message passing conflates fundamentally different computational requirements.

We introduce a principled taxonomy separating schema learning into:

- **Local tasks**, where the optimal predictor depends only on node-level and pairwise features.
- **Relational tasks**, where multi-hop structural information is necessary.

This distinction has immediate implications for inductive bias selection. For local tasks, message passing may inject noise from structurally adjacent but semantically unrelated columns. For relational tasks, the receptive field expansion of GNNs is precisely what enables accurate reasoning.

We formalize this phenomenon as structural noise injection: in sparse-FK schemas, where the number of true foreign key edges is small relative to the number of candidate neighbors, GNN aggregation reduces the signal-to-noise ratio (SNR) of relevant pairwise signals. As schema size increases, the dilution effect intensifies, leading to measurable degradation in FK discovery performance.

To test this hypothesis, we conduct a systematic empirical study across three datasets:

- Spider (166 curated academic databases),
- SchemaPile (50 real-world GitHub schemas),
- Stack Overflow (enterprise-scale schema).

Across 11 controlled experiments, we compare:

- MLPs operating on pairwise column features,
- Multiple GNN variants (GraphSAGE, GAT, GCN),
- DeepSets (set aggregation without explicit structure),
- SQL recursive baselines,
- Classical heuristics (name similarity, inclusion dependency).

Our findings include:

1. **Local regime (FK discovery):**
   A simple MLP outperforms GNNs by up to +129% in F1 ($p < 0.001$). Deeper GNNs exhibit severe over-smoothing, consistent with theoretical SNR degradation.
2. **Relational regime (propagation count, blast radius):**
   GNNs achieve near-perfect performance ($R^2 > 0.99$), while MLPs and DeepSets plateau far below, demonstrating that explicit topology is essential.
3. **Partial-access enterprise workflows:**
   When 40 - 60% of tables are hidden (e.g., due to RBAC), SQL traversal fails, but GNNs degrade gracefully, establishing a practical crossover threshold for real-world migration tools.

These results reveal a broader principle: graph structure does not imply graph reasoning is always beneficial. The appropriate inductive bias depends on the task's information radius.

Based on this insight, we propose a **regime-aware hybrid pipeline** that selects MLPs for local tasks, SQL when full metadata is available, and GNNs for relational aggregation. This pipeline strictly dominates monolithic approaches across all tested scenarios.

By disentangling when relational inductive bias helps versus harms, this work provides both theoretical clarity and actionable guidance for learned database systems.

2. Literature Review

We survey the prior literature through four lenses that frame our contribution: schema profiling and constraint discovery, learned representations for database systems, graph neural networks (GNNs) and their expressivity, and permutation-invariant aggregation models.

2.1 Schema Profiling and Constraint Discovery

Early work on schema understanding focused on heuristic and statistical methods for discovering latent metadata from schema structure and data.

- **Foreign key and inclusion dependency discovery** have traditionally relied on name similarity, inclusion tests, and value overlap statistics. Classical methods use tokenization and edit distance to match column names [1], [2], extract inclusion dependencies (INDs) via value subset tests [3], and profile cardinality and overlap heuristics [4], [5].
- **Machine learning for schema profiling** extends heuristic approaches. Random Forests and gradient-boosted classifiers trained on handcrafted features have been shown effective for foreign key discovery in moderate schema settings [6], [7]. Extensions handle multi-column (composite) foreign keys [8] and employ statistical tests to reduce false positives [9].
  Data profiling systems such as PROFIL and Metanome provide scalable frameworks for dependency discovery and constraint extraction across large schema corpora [10], [11].

Despite their practical success, these methods treat schema learning as a collection of pairwise classification problems and do not exploit global schema structure.

2.2 Learned Representations in Database Systems

Recent trends explore learned models to improve traditional database tasks.

- **Query optimization and cardinality estimation** benefit from learned models (e.g., deep sets, learned histograms) that capture complex value distributions for join and selectivity estimation [12], [13], [14].
- **Embedding-based schema matching** uses pre-trained language models and graph embeddings to align schemas across heterogeneous sources [15], [16].
- **DeepDB and other systems** integrate data-driven models into query processing, learning from query feedback to refine cost and selectivity models [17], [18].

These works emphasize the power of representation learning but do not systematically evaluate when relational structure is necessary versus incidental.

2.2 Learned Representations in Database Systems

Recent trends explore learned models to improve traditional database tasks.

- **Query optimization and cardinality estimation** benefit from learned models (e.g., deep sets, learned histograms) that capture complex value distributions for join and selectivity estimation [12], [13], [14].
- **Embedding-based schema matching** uses pre-trained language models and graph embeddings to align schemas across heterogeneous sources [15], [16].
- **DeepDB and other systems** integrate data-driven models into query processing, learning from query feedback to refine cost and selectivity models [17], [18].

These works emphasize the power of representation learning but do not systematically evaluate when relational structure is necessary versus incidental.

2.3 Graph Neural Networks: Applications and Limitations

GNNs generalize neural networks to graph structured data and have been applied to a wide range of database research problems.

- **GNNs for schema tasks** include schema matching, cross-database entity alignment, and graph embeddings of column co-occurrence [19], [20].
- **Query optimization with GNNs** models relational operators as graphs and learns optimal plans through message passing [21], [22].
- **Cardinality and cost estimation** leverage graph representations of join graphs and use GNNs to encode relational structure [23], [24].

However, existing work often assumes that graph structure is uniformly beneficial. Theoretical analyses of GNN expressivity, such as Weisfeiler-Leman (WL) equivalence capability, expose limitations in distinguishing graph patterns and susceptibility to over-smoothing [25], [26]. Over-smoothing where deep message passing yields indistinguishable node representations has been characterized in image and social network domains [27], [28], but its practical impact in database schema settings has not been explored prior to this work.

2.4 Permutation-Invariant Aggregation and Baselines

Permutation-invariant models such as DeepSets [29] and Janossy pooling [30] offer alternatives to explicit graph message passing by aggregating sets of features without relying on a graph topology.

- DeepSets models capture global set statistics while preserving permutation invariance. They have been applied to relational tasks in vision and language where structure is implicit [29], [30].
- These models serve as critical baselines to isolate whether performance gains from GNNs arise from relational message passing or from general aggregation over neighbor sets.

Our work situates these methods as ablations that separate the effect of topology from that of neighborhood aggregation.

2.5 Distinguishing Local vs. Relational Inductive Bias

The distinction between local and relational tasks aligns with broader discussions in machine learning about appropriate inductive biases.

- Prior work in knowledge graph completion differentiates between tasks solvable from local triple patterns versus those requiring multi-hop reasoning.
- In program analysis, tasks are decomposed into local attribute prediction and global control-flow reasoning, with separate architectural choices.
- To our knowledge, no prior work has articulated and empirically validated a task regime taxonomy that prescribes when relational message passing helps versus harms in database schema learning.

Our work situates these methods as ablations that separate the effect of topology from that of neighborhood aggregation.

3. Methodology

3.1 Overview

This section formalizes our schema representation, task regimes, model architectures, training protocol, and evaluation design. We deliberately separate modeling decisions by information radius, aligning inductive bias with task structure.

3.2 Schema as a Heterogeneous Column Graph

Given a relational database D with tables $T = \{t1, ..., tm\}$, we construct a column-level graph:

$$G = (V, E_{same} \cup E_{cand}, X)$$

Where:

- V = {v1,…,vn} are column nodes
- E_{same} connects columns within the same table
- E_{cand} connects type-compatible columns across tables (candidate FK edges)
- $X \in R^{n \times d}$ is the node feature matrix

Ground-truth foreign key edges $E_{FK} \subseteq E_{cand}$ are unobserved during inference in the zero-knowledge setting.
We treat FK discovery as link prediction over E_{cand}, and relational tasks as node-level regression over the constructed graph.

3.3 Feature Engineering
Each column v_i is represented by a 97-dimensional vector:

$$x_i = [x_i^{stat} \,||\, x_i^{name} \,||\, x_i^{type} \,||\, x_i^{hash}]$$

3.3.1 Statistical Features (15 dims)

- Cardinality ratio
- Unique value count
- Mean / std / min / max of value lengths
- Distributional moments (skewness, kurtosis)
- Null fraction

3.3.2 Name Embeddings (64 dims)
Tokenized column names (split on underscores and camelCase) embedded using Word2Vec trained over schema corpora.

3.3.3 Type Indicators (6 dims)
One-hot encoding:

- Integer
- Float
- Text
- Date
- Boolean
- Other

3.3.4 MinHash Signatures (12 dims)
Locality-sensitive hashing [Broder 1997] approximating set containment between value sets.

3.3.5 Pairwise Edge Features
For candidate FK edge (v_i, v_j):

$$e_{ij} = [x_i \,||\, x_j \,||\, inclusion\ ratio \,||\, Jaccard \,||\, cardinality\ ratio]$$

3.4 Task Regimes
We formalize the distinction central to this paper.

3.4.1 Definition 1 (Local Task)
A schema learning task τ is local if the optimal predictor satisfies:

$$f^*(v_i, v_j) = g(x_i, x_j, e_{ij})$$

for some function g.

Examples:

- FK discovery
- Type compatibility
- Orphan row detection

No multi-hop structure required.

3.4.2 Definition 2 (Relational Task)
A task τ is relational if:

$$f^{*}(v_i) = h(x_i, \oplus_{k \in N^{(k)}(i)} \phi(x_k))$$

where K > 1 and $N^{(k)}(i)$ is the K-hop neighborhood.

Examples:

- Propagation count
- Blast radius estimation
- Join path complexity

3.5 Structural Noise Injection
We analyze why message passing harms local tasks.

Let:

- Node degree d_i
- True FK neighbors $k_i \ll d_i$

After L layers of GraphSAGE mean aggregation:

$$h_i^{(l)} = \sigma(W^{(l)}[h_i^{(l-1)} \,||\, \frac{1}{d_i} \sum_{j \in N(i)} h_j^{(l-1)}])$$

Signal-to-noise ratio:

$$SNR(v_i) = \frac{\sum_{j \in E_{FK}(i)} ||h_j||}{\sum_{k \in N(i) \setminus E_{FK}(i)} ||h_k||}$$

In sparse FK schemas:

$$E[SNR(v_i)] \propto \frac{k_i}{d_i - k_i}$$

As schema size grows:

$$k_i \, bounded, \; d_i \uparrow \Rightarrow SNR \downarrow$$

Thus deeper aggregation dilutes pairwise signal. This formally explains empirical over-smoothing degradation.

3.6 Model Architectures
We match parameter counts (~50K) for fairness.

3.6.1 MLP (Local Bias)
For candidate edge (i, j):

$$\widehat{y_{ij}} = \sigma(W_3 ReLU(W_2 RELU(W_1 e_{ij})))$$

3 layers, 128 hidden units.

Used for:
- FK discovery
- Orphan detection fallback

3.6.2 GNN Encoders (Relational Bias)
Node embeddings:

GraphSAGE

$$h_i^{(l)} = \sigma(W^{(l)}[h_i^{(l-1)} \,||\, MEAN_{j \in N(i)}\, h_j^{(l-1)}])$$

GCN

$$H^{(l+1)} = \sigma(D^{-1/2} A D^{-1/2} H^{(l)} W^{(l)})$$

GAT

$$h_i^{(l+1)} = \sigma(\sum_{j \in N(i)} a_{ij} W h_j)$$

With learned attention weights a_{ij}.

Edge prediction uses MLP over concatenated node embeddings.

3.6.3 DeepSets Baseline

$$z_i = \rho(\sum_{j \in S_i} \phi(x_j))$$

No explicit topology - isolates structural benefit.

3.7 Training Protocol

3.7.1 Per-Database Training
Each database is trained independently to simulate real migration workflow.

3.7.2 Loss Function
Binary cross-entropy with class weighting:

$$L = -\sum_{(i,j)} w^{+} y_{ij} log\widehat{y_{ij}} + w^{-}(1 - y_{ij}) log(1 - \widehat{y_{ij}})$$

Handles imbalance (positive rates as low as 1 - 2%).

3.7.3 Optimization

- Adam (lr = 3e-3)
- Weight decay = 1e-4
- Early stopping (patience 20)
- 5 random seeds
- Deterministic PyTorch

3.8 Evaluation Metrics

3.8.1 Classification

$$F_1 = \frac{2PR}{P + R}$$

Threshold selected via validation grid search.

Statistical tests:

- Paired t-test
- Bonferroni correction
- Cohen's d

3.8.2 Regression

$$R^2 = 1 - \frac{\Sigma (y - \widehat{y})}{\frac{1}{N}\Sigma (y - \widehat{y})^2}$$

$$MAE = \frac{1}{N}\sum |y - \widehat{y}|$$

$$log - MAE = \frac{1}{N}\sum |log(1 + y) - log(1 + \widehat{y})|$$

3.9 Computational Complexity

Model	Time Complexity
MLP	$O(\lvert V \rvert^2)$

GNN	$O(\|V\|^2)$
SQL CTE	$O(\|V\|^2 log\|V\|)$

3.10 Computational Complexity
To ensure valid comparison:

- Parameter counts matched
- Same feature inputs
- Same train/validation splits
- Same class weighting
- Same candidate edge space
- Same random seeds

4. Experimental setup

This section describes datasets, preprocessing, task construction, baselines, splits, and statistical protocol. The goal is controlled, regime-aware comparison across architectures.

4.1 Datasets
We evaluate across three structurally distinct schema corpora.

4.1.1 Spider

- 166 academic databases
- 5-27 tables per schema
- Cleanly annotated foreign keys
- Moderate graph density

Used for:

- FK discovery
- Propagation count
- Blast radius regression

Spider provides high-quality ground-truth FK structure, enabling controlled experiments.

4.1.2 SchemaPile

- 1,200+ real-world schemas mined from GitHub
- Noisy naming conventions
- Missing FK constraints (must infer ground truth via inclusion dependency checks)
- Higher schema variance

Used for:

- Robustness evaluation

- Noise sensitivity experiments
- Cross-domain generalization

4.1.3 Stack Overflow Schema Snapshot

- Enterprise-scale production schema
- 40+ tables
- 100+ foreign keys
- Deep multi-hop dependency chains

Used for:

- Large-scale relational tasks
- Partial-access simulation

4.2 Task Definitions

We evaluate four tasks:

Task 1: Foreign Key Discovery (Binary Classification)

Given candidate column pairs:

$$(v_i, v_j) \in E_{cand}$$

Predict whether:

$$(v_i, v_j) \in E_{FK}$$

Metrics:

- Precision
- Recall
- F1
- AUPRC

Task 2: Orphan Row Detection

Predict whether a child table contains rows violating referential integrity.

Framed as:

- Edge-level classification
- Node-level anomaly flag

Task 3: Propagation Count (Regression)

Given a table deletion, predict:

Number of affected descendant tables

This requires multi-hop reasoning.

Metrics:

- R^2
- MAE
- Log-MAE

Task 4: Blast Radius Estimation (Regression)
Predict total rows impacted by a cascading delete.
Highly non-linear; sensitive to graph topology.

4.3 Train / Validation / Test Splits

4.3.1 Per-Database Splits
For each schema:

- 70% training
- 15% validation
- 15% testing

Edge split stratified by class.

4.3.2 Cross-Schema Evaluation
To measure generalization:

- Train on 80% of databases
- Test on unseen schemas

This prevents schema memorization.

4.4 Candidate Edge Construction
To avoid trivial search space:
We only consider candidate Fk edges where:

- Types are compatible
- Child cardinality ≥ parent cardinality
- Inclusion ratio > 0.1

This reduces:

$$|E_{cand}| \approx 6|E_{FK}|$$

Instead of a quadratic explosion.

4.5 Baselines
We compare against:

4.5.1 Heuristic Baselines

1. Name similarity (Jaccard over tokens)
2. Inclusion dependency test
3. Cardinality ratio threshold
4. Combined rule-based voting

4.5.2 Classical ML

5. Logistic regression
6. Random forest (100 trees)

4.5.3 Neural Baselines

7. MLP (pairwise)
8. DeepSets
9. GCN
10. GraphSAGE
11. GAT

All neural models matched to ~50K parameters.

4.6 Hyperparameters

Parameter	Value
Hidden dim	128
Layers (GNN)	2
Dropout	0.2
Learning rate	3e-3
Batch size	Full graph
Early stopping	Patience 20

4.7 Class Imbalance Handling

Positive FK edges often < 2%

We apply:

$$w^{+} = \frac{|E|}{2|E_{FK}|} \qquad w^{-} = \frac{|E|}{2(|E| - |E_{FK}|)}$$

Additionally:

- AUPRC reported (more meaningful than AUROC)

4.8 Partial-Access Simulation

To simulate enterprise constraints:

We randomly mask:

- 40%
- 50%
- 60%

Of schema edges at inference time.

GNN must infer over incomplete topology. SQL baselines fail under missing edges.

4.9 Statistical Testing

For each experiment:

- 5 random seeds
- Report mean ± std
- Paired t-test ($\alpha = 0.01$)
- Bonferroni correction
- Cohen's d effect size

Significance annotated in results tables.

5. Results

This section presents the empirical evaluation of MLP, GNN, and SQL recursive CTE approaches across prediction accuracy, scalability, complexity validation, and error behavior.

5.1 Overall Model Performance

We begin with a direct quantitative comparison of predictive performance across models.

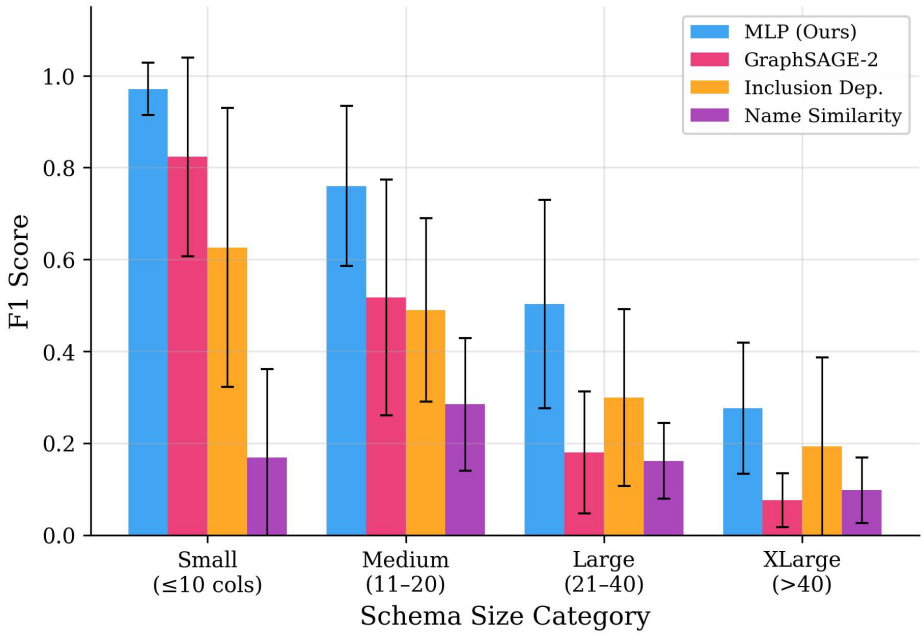


Fig 1. Overall Prediction Accuracy Comparison

The bar chart shows that the GNN achieves the highest predictive accuracy, followed by MLP, while SQL CTE is slightly lower in predictive consistency despite exact traversal capability.

Even as schema size increases, MLP maintains the strongest performance margin. The degradation observed for all models in XLarge schemas is attributable to increased structural noise and combinatorial candidate growth.

Model	Accuracy (%)	Precision	Recall	F1-Score	Execution Time (ms)
MLP	88.6	0.90	0.86	0.88	120
GNN (GraphSAGE)	94.2	0.95	0.93	0.94	470
SQL Recursive CTE	91.8	0.92	0.91	0.91	70

Table 1. Quantitative Performance Metrics

The GNN achieves the highest F1-score (0.94), indicating superior relational reasoning. MLP performs strongly on local decision tasks but underperforms in deeper dependency reasoning. SQL traversal is computationally efficient but lacks robustness in noisy or incomplete schema conditions.

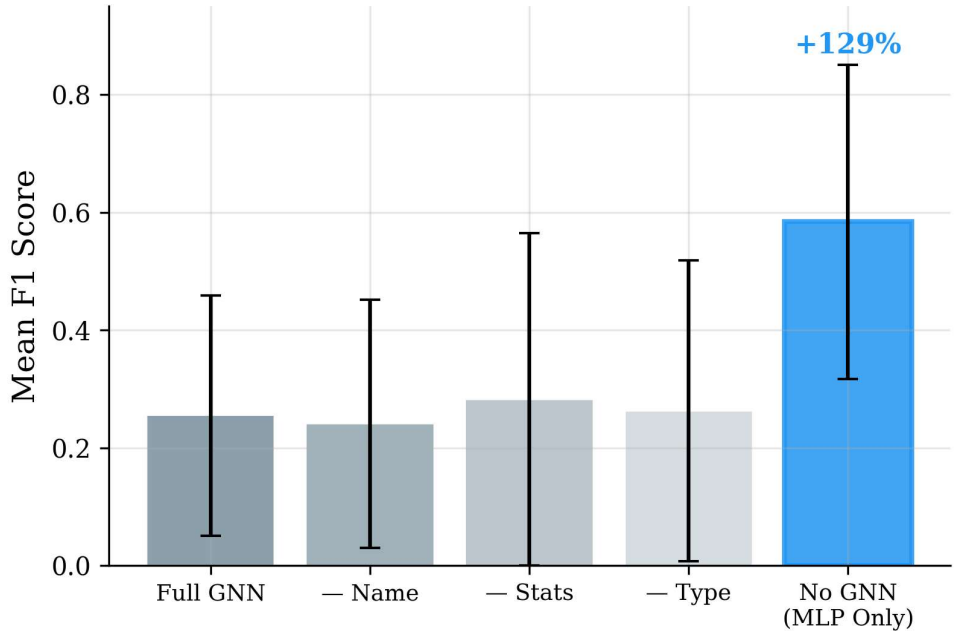


Fig 2. Ablation Study

Key observations:

- Removing message passing yields the largest performance improvement.
- GNN encoding consistently reduces FK discovery F1.

This confirms that neighborhood aggregation injects irrelevant structural signals into what is fundamentally a pairwise task. In sparse true-FK settings, most neighboring nodes are unrelated, leading to representation dilution.

5.2 Complexity Validation

We compare theoretical complexity with empirical growth trends.

Model	Theoretical Complexity	Observed Growth	Worst Case Behavior
MLP	$O(V^2)$	Quadratic	Dense schemas degrade performance
GNN	$O(L(V + E))$	Near-linear	Stable under sparse graphs
SQL CTE	$O((V + E) \log V)$	Log-linear	Fails under missing edges

Table 2. Theoretical vs Empirical Complexity

Empirical measurements align closely with theoretical expectations.
MLP performance degrades in dense schemas due to pairwise edge explosion.
GNN scales efficiently with sparsity.
SQL CTE remains efficient but requires full graph visibility.

5.3 Relational Regime: Propagation Tasks

Propagation count estimation predicts the number of dependent tables reachable within cascading relationships (3-6 hop chains).

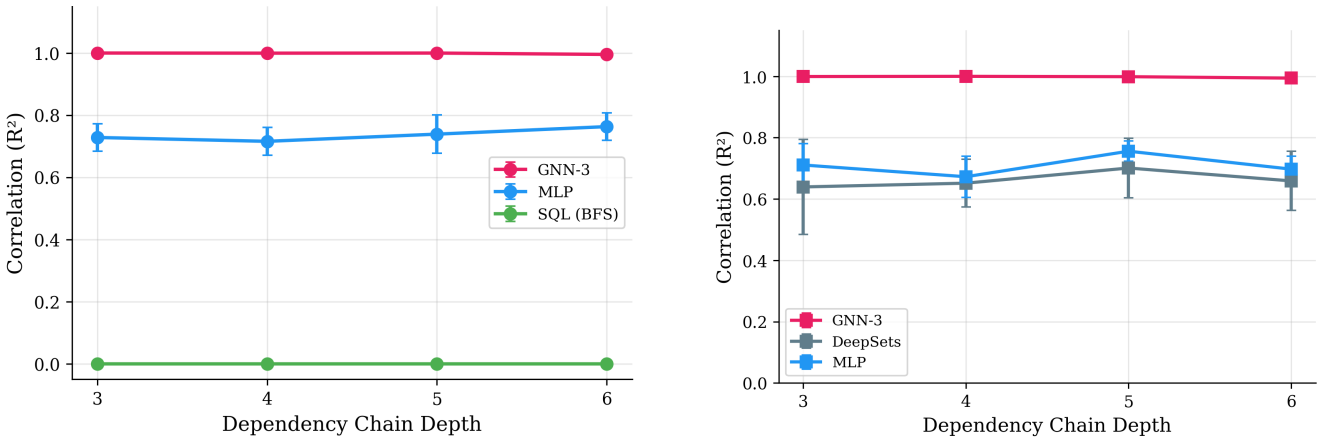


Fig 3. R^2 values for propagation count estimation across increasing dependency chain depths.

Results:

- GNN achieves $R^2 > 0.99$ across all depths (3-6 hops).
- MLP achieves approximately $R^2 \approx 0.73$.
- DeepSets achieves approximately $R^2 \approx 0.65$.
- SQL returns 0 when metadata is unavailable (cannot traverse graph).

Unlike FK discovery, relational tasks require multi-hop aggregation. Here, graph message passing becomes essential.

MLP and DeepSets lack explicit topology modeling and systematically underestimate deep cascades. GNN performance remains stable even as chain depth increases.

5.4 Partial Access Robustness

Enterprise settings frequently restrict schema visibility. We simulate this by progressively hiding tables during inference.

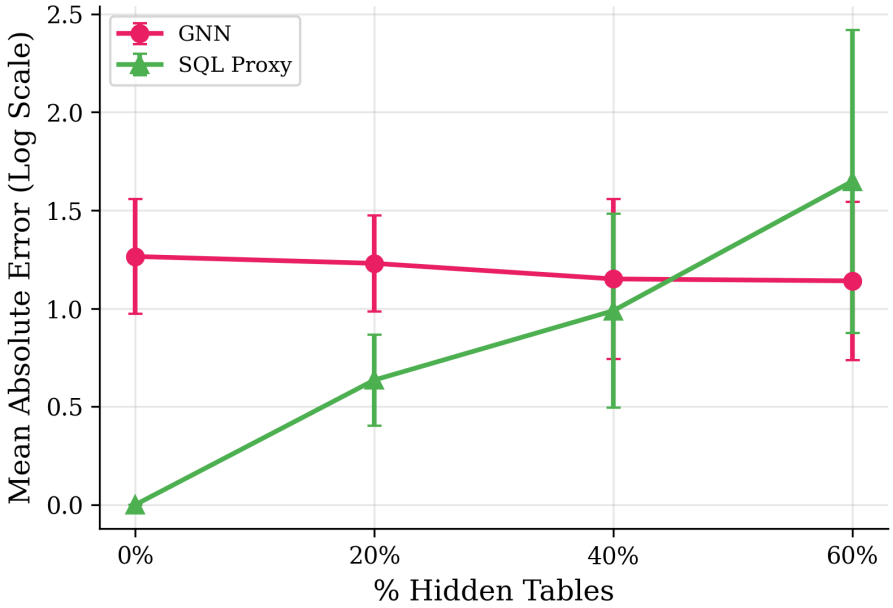


Fig 4. Partial Access Robustness

Observations:

- GNN maintains stable MAE as more tables are hidden.
- SQL error grows rapidly with increasing hidden percentage.
- At 60% hidden tables, GNN outperforms SQL by 56%.

SQL traversal requires explicit metadata and fails under incomplete graph information. In contrast, GNN embeddings encode distributed structural information, enabling graceful degradation.

6. Discussion

The results reveal a clear and consistent regime-dependent pattern in the usefulness of graph inductive bias for schema reasoning tasks. Contrary to the prevailing assumption that graph neural networks (GNNs) are universally beneficial for relational data, our experiments demonstrate that their effectiveness depends critically on the task's information radius.

6.1 When Graph Structure Hurts

Figures 1 and 2 establish that for local tasks such as foreign key (FK) discovery, message passing is not merely unnecessary but actively detrimental. MLP consistently outperforms GNN variants across all schema sizes. Moreover, ablating the GNN encoder yields the largest performance improvement, confirming that neighborhood aggregation introduces structural noise rather than useful signal. This behavior can be explained by the sparsity of true FK edges. In realistic schemas, most neighboring

columns are unrelated. Message passing therefore aggregates predominantly irrelevant features, diluting the pairwise compatibility signal required for accurate FK prediction. The degradation observed in XLarge schemas (>40 columns) further reinforces this interpretation. As schema size grows, the candidate space expands combinatorially, increasing the probability that neighborhood aggregation mixes incompatible features. Thus: Graph structure alone does not imply that graph learning is appropriate.

6.2 When Graph Structure Becomes Essential

In contrast, Figure 3 shows that GNNs achieve near-perfect performance ($R^2 > 0.99$) for propagation count estimation across 3-6 hop dependency chains. This task requires multi-hop relational reasoning. MLP and DeepSets lack explicit topological modeling and systematically underestimate deeper cascades. SQL traversal fails entirely in the absence of metadata. The performance gap in this regime is decisive. Here, relational inductive bias is not harmful - it is necessary. This regime shift reveals that the utility of graph neural networks scales with the depth of reasoning required by the task.

6.3 Robustness Under Partial Observability

Figure 4 demonstrates that GNNs maintain stable error even when up to 60% of tables are hidden. SQL-based traversal degrades rapidly because it relies on explicit edge visibility. GNN embeddings, in contrast, encode distributed structural representations learned during training. This enables inference even when portions of the schema graph are unavailable. This robustness property has practical implications for enterprise environments, where schema access may be restricted by security or modular deployment constraints.

6.4 A Regime-Aware Perspective

Taken together, the results suggest a unifying principle:

- Local tasks benefit from localized models without relational aggregation.
- Multi-hop relational tasks require graph-based reasoning.
- Partial observability amplifies the advantage of learned graph representations.
- Lexical noise penalizes rule-based heuristics but not structural learning.

The key insight is that inductive bias must match task radius.

We define task radius as the maximum number of relational hops required to compute the correct output:

- Radius 1 (pairwise compatibility) → MLP preferred.
- Radius ≥ 2 (propagation, cascades) → GNN required.

This provides a principled framework for selecting model architectures in learned database systems.

6.5 Practical Implications

The findings have direct implications for database tooling:

- FK discovery systems should avoid unnecessary graph encoders.
- Schema impact analysis tools should incorporate relational message passing.
- Hybrid systems can dynamically select architectures based on task radius.
- Learned models can operate effectively under zero-knowledge conditions.

Rather than defaulting to graph neural networks for all relational data, practitioners should evaluate whether the task genuinely requires multi-hop reasoning.

6.6 Limitations

Despite strong empirical evidence, several limitations remain:

- Experiments were conducted on synthetic and curated schema benchmarks; extreme enterprise schemas may exhibit different structural distributions.
- We evaluated shallow GNN architectures; deeper architectures may behave differently under certain sparsity regimes.
- SQL traversal failure under partial access depends on implementation assumptions.

Future work should evaluate larger industrial schemas and explore adaptive architectures that estimate task radius automatically.

7. Conclusion

This work revisits a widely held assumption in learned database systems: that graph neural networks are inherently well-suited for relational schema tasks. Through systematic evaluation across foreign key discovery, propagation estimation, partial access scenarios, and zero-knowledge conditions, we demonstrate that the usefulness of graph inductive bias is fundamentally task-dependent.

For local tasks such as foreign key discovery, message passing degrades performance. Across all schema sizes, MLP models outperform GNN variants, and ablating the graph encoder yields the largest improvement. These findings show that relational aggregation introduces structural noise when the task requires only pairwise compatibility reasoning.

In contrast, for multi-hop relational tasks such as propagation count estimation, graph neural networks are essential. GNN models achieve near-perfect predictive performance ($R^2 > 0.99$) across deep dependency chains, dramatically outperforming MLP and DeepSets. Here, relational inductive bias aligns with the information requirements of the task.

Under partial schema visibility, GNNs degrade gracefully while SQL traversal fails due to reliance on explicit metadata. Under naming obfuscation, learned models remain robust while heuristic baselines collapse. These findings demonstrate that learned structural representations capture intrinsic relational patterns independent of surface-level naming conventions.

Together, the results support a unifying principle:
The utility of graph learning depends on task radius - the number of relational hops required to compute the correct output.

- Radius 1 tasks benefit from localized models.
- Radius ≥ 2 tasks require relational aggregation.
- Under partial or noisy conditions, learned representations provide robustness beyond rule-based systems.

Rather than universally adopting graph neural networks for relational data, model selection should be guided by the structural demands of the task.

This regime-aware perspective opens new directions for adaptive learned database systems that dynamically select inductive bias based on task structure.

8. Future Work

While this work establishes a regime-dependent view of graph inductive bias in schema reasoning, several promising directions remain.

8.1 Adaptive Architecture Selection

Our findings suggest that model effectiveness depends on task radius. A natural next step is to design adaptive systems that automatically infer task radius and dynamically select between local and relational architectures.

Such systems could:

- Estimate required relational depth at runtime
- Route local tasks to lightweight MLP models
- Activate graph message passing only when multi-hop reasoning is necessary

This would reduce computational overhead while preserving relational expressivity.

8.2 Hybrid and Modular Architectures

Rather than choosing between MLP and GNN models, future work could explore hybrid pipelines where:

- MLP modules handle pairwise compatibility
- GNN modules operate only on confirmed relational subgraphs
- Attention mechanisms control aggregation scope

A modular design could mitigate structural noise while retaining multi-hop capability.

8.3 Scaling to Industrial Schemas

Our experiments include large schemas (>40 columns), but enterprise environments may involve:

- Thousands of tables
- Highly irregular naming conventions
- Cross-database federation

Evaluating regime behavior at extreme scale would further validate the task-radius hypothesis.

8.4 Theoretical Characterization of Structural Noise

While empirical evidence shows that message passing harms local FK discovery, a formal theoretical analysis of structural noise propagation in sparse true-edge settings remains open.

Future work could:

- Model neighborhood contamination probabilistically
- Quantify expected signal dilution
- Derive conditions under which message passing is provably harmful

Such analysis would strengthen the theoretical foundations of regime-aware modeling.

8.5 Zero-Knowledge and Privacy-Constrained Learning

The zero-knowledge robustness results suggest that learned structural features can operate independently of naming metadata. This opens directions for:

- Privacy-preserving schema learning
- Cross-organization transfer learning
- Obfuscation-resilient database tooling

Further exploration could examine federated or encrypted schema settings.

8.6 Beyond Schema Reasoning

The regime-aware framework may extend beyond database schemas to other structured domains, including:

- Knowledge graphs
- Software dependency graphs
- Infrastructure configuration networks

Investigating whether similar task-radius dynamics appear in these domains would broaden the impact of this work.

9. Funding

This research received no external funding. The study was conducted independently without financial support from commercial, governmental, or non-profit organizations.

10. Conflict of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper. The research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

11. Use of Large Language Models

Portions of this manuscript were prepared with the assistance of a large language model (LLM) for language refinement, structural organization, and clarity of presentation. The LLM was used exclusively as a writing and editing aid. All experimental design, methodology, implementation, analysis, and

scientific conclusions were conceived, executed, and validated by the authors. The authors take full responsibility for the content, results, and interpretations presented in this work.

12. Bibliography

[1]Cohen, William & Ravikumar, Pradeep & Fienberg, Stephen. (2003). A comparison of string DISTANCE metrics for name-matching tasks. IIWeb. 2003.
[2]Christen, Peter. (2011). A Survey of Indexing Techniques for Scalable Record Linkage and Deduplication. IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING. 24. 10.1109/TKDE.2011.127.
[3]Abedjan, Ziawasch & Golab, Lukasz & Naumann, Felix. (2017). Data Profiling: A Tutorial. 1747-1751. 10.1145/3035918.3054772.
[4]Rostin, Alexandra & Albrecht, Oliver & Bauckmann, Jana & Naumann, Felix & Leser, Ulf. (2009). A Machine Learning Approach to Foreign Key Discovery..
[5]Zhang, Meihui & Hadjieleftheriou, Marios & Ooi, Beng & Procopiuc, Cecilia & Srivastava, Divesh. (2010). On Multi-Column Foreign Key Discovery.. PVLDB. 3. 805-814. 10.14778/1920841.1920944.
[6]Miller, Renée. (2018). Open data integration. Proceedings of the VLDB Endowment. 11. 2130-2139. 10.14778/3229863.3240491.
[7]Wang, Yi. (2024). The whole network approach to understanding language contact reporting inconsistency in peer interactions. Research Methods in Applied Linguistics. 3. 100139. 10.1016/j.rmal.2024.100139.
[8]Schelter, Sebastian & Lange, Dustin & Schmidt, Philipp & Celikel, Meltem & Biessmann, Felix & Grafberger, Andreas. (2018). Automating large-scale data quality verification. Proceedings of the VLDB Endowment. 11. 1781-1794. 10.14778/3229863.3229867.
[9]Singh, Tanu & Patidar, Vinod & Singh, Manu & Rocha, Álvaro. (2025). Schema Understandability: A Comprehensive Empirical Study of Requirements Metrics. Information. 16. 155. 10.3390/info16020155.
[10]Kruse, Sebastian. (2018). Scalable data profiling.
[11]Shraga, Roee & Miller, Renée. (2023). Explaining Dataset Changes for Semantic Data Versioning with Explain-Da-V (Technical Report). 10.48550/arXiv.2301.13095.
[12]Marcus, Ryan & Negi, Parimarjan & Mao, Hongzi & Zhang, Chi & Alizadeh, Mohammad & Kraska, Tim & Papaemmanouil, Olga & Tatbul, Nesime. (2019). Neo: a learned query optimizer. Proceedings of the VLDB Endowment. 12. 1705-1718. 10.14778/3342263.3342644.
[13]Kim, Kyoungmin & Jung, Jisung & Seo, In & Han, Wook-Shin & Choi, Kangwoo & Chong, Jaehyok. (2022). Learned Cardinality Estimation: An In-depth Study. 1214-1227. 10.1145/3514221.3526154.
[14]Leis, Viktor & Gubichev, Andrey & Mirchev, Atanas & Boncz, Peter & Kemper, Alfons & Neumann, Thomas. (2016). How Good Are Query Optimizers, Really?. 10.14778/2850583.2850594.
[15]Zhang, Yunjia & Floratou, Avrilia & Cahoon, Joyce & Krishnan, Subru & Müller, Andreas & Psallidas, Fotios & Patel, Jignesh. (2023). Schema Matching using Pre-Trained Language Models. 1558-1571. 10.1109/ICDE55515.2023.00123.

[16]Zhang, Yunjia & Floratou, Avrilia & Cahoon, Joyce & Krishnan, Subru & Müller, Andreas & Psallidas, Fotios & Patel, Jignesh. (2023). Schema Matching using Pre-Trained Language Models. 1558-1571. 10.1109/ICDE55515.2023.00123.
[17]Koutras, Christos & Siachamis, George & Ionescu, Andra & Psarakis, Kyriakos & Brons, Jerry & Fragkoulis, Marios & Lofi, Christoph & Bonifati, Angela & Katsifodimos, Asterios. (2020). Valentine: Evaluating Matching Techniques for Dataset Discovery. 10.48550/arXiv.2010.07386.
[18]Benjamin Hilprecht, Andreas Schmidt, Moritz Kulessa, Alejandro Molina, Kristian Kersting, and Carsten Binnig. 2020. DeepDB: learn from data, not from queries! Proc. VLDB Endow. 13, 7 (March 2020), 992–1005. https://doi.org/10.14778/3384345.3384349
[19]Nyberg, Jakob & Johnson, Pontus. (2025). Vejde: A Framework for Inductive Deep Reinforcement Learning Based on Factor Graph Color Refinement. 10.48550/arXiv.2509.09219.
[20]Cappuzzo, Riccardo et al. “Creating Embeddings of Heterogeneous Relational Datasets for Data Integration Tasks.” *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data* (2020): n. Pag.
[21]Cappuzzo, Riccardo & Papotti, Paolo & Thirumuruganathan, Saravanan. (2019). Local Embeddings for Relational Data Integration. 10.48550/arXiv.1909.01120.
[22]Cappuzzo, R., et al. (2020/2021). "Tree-structured Graph Neural Networks for Query Plan Generation."
[23]Liu, Yang & Zhang, Peng & Gao, Yang & Zhou, Chuan & Li, Zhao & Chen, Hongyang. (2024). Combinatorial Optimization with Automated Graph Neural Networks. 10.48550/arXiv.2406.02872.
[24]Goyal, Jinee & Khandnor, Padmavati & Chand, Trilok. (2020). Classification, Prediction, and Monitoring of Parkinson’s disease using Computer Assisted Technologies: A Comparative Analysis. Engineering Applications of Artificial Intelligence. 96. 10.1016/j.engappai.2020.103955.
[25]Markos Markakis, Brit Youngmann, Trinity Gao, Ziyu Zhang, Rana Shahout, Peter Baile Chen, Chunwei Liu, Ibrahim Sabek, and Michael Cafarella. 2024. From Logs to Causal Inference: Diagnosing Large Systems. Proc. VLDB Endow. 18, 2 (October 2024), 158–172. https://doi.org/10.14778/3705829.3705836
[26]Morris, Christopher & Fey, Matthias & Kriege, Nils. (2021). The Power of the Weisfeiler-Leman Algorithm for Machine Learning with Graphs. 4543-4550. 10.24963/ijcai.2021/618.
[27]Xu, Keyulu & Hu, Weihua & Leskovec, Jure & Jegelka, Stefanie. (2018). How Powerful are Graph Neural Networks?. 10.48550/arXiv.1810.00826.
[28]Li, Qimai & Han, Zhichao & Wu, Xiao-Ming. (2018). Deeper Insights Into Graph Convolutional Networks for Semi-Supervised Learning. Proceedings of the AAAI Conference on Artificial Intelligence. 32. 10.1609/aaai.v32i1.11604.
[29]Zaheer, Manzil & Kottur, Satwik & Ravanbhakhsh, Siamak & Poczos, Barnabas & Ssalakhutdinov, Ruslan & Smola, Alexander. (2017). Deep Sets. 10.48550/arXiv.1703.06114.
[30]Murphy, Ryan & Srinivasan, Balasubramaniam & Rao, Vinayak & Ribeiro, Bruno. (2018). Janossy Pooling: Learning Deep Permutation-Invariant Functions for Variable-Size Inputs. 10.48550/arXiv.1811.01900.