OWNERSHIP IS THE MISSING DECLARATION: DECIDING RELATIONAL TARGET LEAKAGE FROM THE SCHEMA

Anonymous authors
Paper under double-blind review

ABSTRACT

Target leakage is treated as a statistical property of data: a column is leaky if a probe trained on it predicts the target too well, so no column can be judged before it is read. We ask instead which features the declared schema forces to determine the target on every database it admits, and what any schema-only method must leave undecided, where a task is a tree of many-to-one lookups over keys, foreign keys and copies. On a fragment allowing composite, cyclic and self-referencing foreign keys, a whole-key closure decides this soundly and completely, in time near-linear in the copy-closed view, and every negative verdict carries a counterexample database. One further SQL-declarable construct makes the question undecidable for feature sets, and targets aggregated over future rows, such as churn, are outside the fragment. Four such mechanisms cost models up to 22.4 accuracy points and 54.5 macro-F1, and the closure abstains on every one. Across 5,094 WikiDBs databases, 119 CTU databases and 23 benchmark tasks, keys and foreign keys alone certify no non-identifier leak at all; across 22,989 public schemas they certify 124 of 1,137,685 targets, every one a key cycle. Two open-source ERPs do declare which table owns a value, yet neither dump carries it, and one never emits a database-level foreign key at all. Blind language-model declarers recover 0 of the 90 columns designers removed by hand: the method certifies what a declaration forces, not leaks found unaided. Checking a declaration on a sample is cheap and carries a bound. On one RelBench task a name-blind generator proposes the declaration and ten sampled rows accept it; the closure certifies the removal, and re-inserting the column raises accuracy by 28.0 points. Given an ownership declaration, the rest of the fragment is decided before a row is read.

1 INTRODUCTION

Target leakage occurs when a feature available to a model encodes the value the model must predict (Kaufman et al., 2012), a leading cause of over-optimistic results (Kapoor & Narayanan, 2023). On multi-table data, we show, a schema that records which table *owns* a value settles the question before a row is read.[1] The problem is acute for multi-table data, where the training table is assembled by joining tables along keys, so a joined column can carry the target's value although no column of the target's own table does. Current practice finds such columns only by reading data. AutoML platforms probe each column and drop the ones that predict the target too well (H2O.ai, 2024; Amazon Web Services, 2024). Curators discard deterministic tasks (Grinsztajn et al., 2022), and relational benchmark designers "manually identify and remove columns that are highly correlated with the target" (Gu et al., 2026; Wang et al., 2024). None can judge a column before reading it, or say which verdicts the schema already forces. A schema could force them, but only if it records which table *owns* a value, and that is the one declaration standard formats omit. Meanwhile relational deep learning and relational foundation models consume exactly the columns reachable along foreign keys (FKs) (Fey et al., 2024; Ranjan et al., 2026; Truong et al., 2025), so their task pools must be screened before features are assembled. We ask:

[1] Unrelated to privacy leakage (Zhu et al., 2019), concept leakage (Havasi et al., 2022) and representation-level label leakage in relational transfer (Azevedo et al., 2026).

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

Which features does the declared schema force to determine the target on every database it admits, and what must any schema-only method leave undecided?

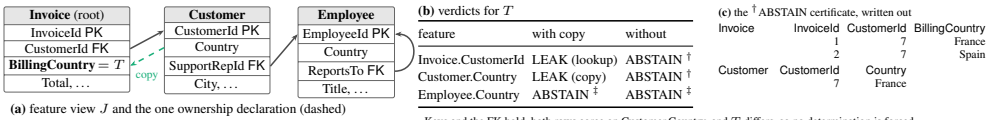
We model a task as a *feature view*: a tree of many-to-one lookups rooted at the entity table, with the target column withheld (§2). One-to-many children are excluded, reaching a model only through aggregates, which lie outside the fragment and are where a measured 22.4-point leak sits (§5.1). A feature receives the verdict LEAK when every admissible database makes the target a function of that feature, never because the two merely correlate in the data at hand. A feature that instead fixes the entity's own key receives IDENTIFIER. It is a leak in the sense of Kaufman et al. (2012) unless legitimately available at prediction time, which needs time stamps or domain knowledge and is out of scope; a certified feature must be excluded or the target reformulated. In Figure 1, whether an invoice's customer determines its billing country depends on one declaration: that the customer table *owns* the country and the invoice copies it.

Results in brief. (Φ_0 and (S1)–(S3) are defined in §2.) **(i) Decide.** On Φ_0 a whole-key closure decides declared determination soundly and completely, finite and unrestricted implication coincide, and one construct more, an FK into a `UNIQUE` superkey, is undecidable; reading (S2) relative to the referenced key restores decidability (Theorems 1, 2, 5). Every declared single-feature determination of a non-constant target is an equality, or a lookup through an *authority* whose whole key the feature fixes (Corollary 2). **(ii) Sound, and complete up to a forced abstention.** Every negative verdict carries a machine-verifiable counterexample database, so abstention outside the closure is forced for every sound schema-only method (Corollary 3).

§5 tests the theory with one detector on public data. Related work is deferred to §6, where the closure and its verdicts let us restate the key–FK redundancy and separability lines in those terms. On leaks planted into six real databases the detector flags every declared leak and abstains, with a certificate, on every undeclared twin. As Corollary 2 predicts, what must be declared is ownership: two fully audited open-source ERPs already write it, 555 copies over 838 tables in one and 169 over 655 in another, and a third confirms the same split on a mined lower bound. Public exports declare far less: guessed copies hold no more often than matched controls, and in the two systems whose source we can read, the declarations exist upstream of the dump (Appendix Q).

2 SETTING: DECLARED LEAKAGE ON THE FEATURE VIEW

Definition 1 (Declared schema, fragment Φ_0)**.** A schema S consists of relations R with attributes $\mathrm{att}(R)$, one primary key $\mathrm{pk}(R) \subseteq \mathrm{att}(R)$ and optional secondary keys, declared `UNIQUE` sets. It has foreign keys $f = R[X] \subseteq R'[\mathrm{pk}(R')]$, with X a list of non-null attributes of R aligned with $\mathrm{pk}(R')$. It also has *copy declarations* $c = (f, a, b)$ with $a \in \mathrm{att}(R)$, $b \in \mathrm{att}(R')$, meaning $r.a = f(r).b$ for every tuple r of R, where $f(r)$ is the tuple that r references, with a and b of one declared domain. Declared domains and semantic types are otherwise typing only. An attribute is *descriptive* if it lies neither in the primary key nor in an FK of its relation, and *free* if moreover no copy declaration targets it. $S \in \Phi_0$ iff: **(S1)** every FK references exactly the primary key of its target. **(S2)** every secondary key contains a free attribute; without it a `UNIQUE` on a child's attribute would let an aggregate of it determine the target (Proposition 2). **(S3)** copy targets are descriptive, each has at most one copy declaration, and the relation-level copy graph, where $R \to R'$ if a copy runs along an FK from R to R', is acyclic. Adding declarations can therefore move a schema out of



(a) feature view J and the one ownership declaration (dashed)

(b) verdicts for T

feature	with copy	without
Invoice.CustomerId	LEAK (lookup)	ABSTAIN †
Customer.Country	LEAK (copy)	ABSTAIN †
Employee.Country	ABSTAIN ‡	ABSTAIN ‡

(c) the † ABSTAIN certificate, written out

	InvoiceId	CustomerId	BillingCountry
Invoice	1	7	France
	2	7	Spain
	CustomerId	Country	
Customer	7	France	

Keys and the FK hold, both rows agree on Customer.Country, and T differs, so no determination is forced.
‡ two customers of one support rep in different countries

Figure 1: Deciding one target on the public Chinook database (`github.com/lerocha/chinook-database`). With the declaration, the closure certifies two leaks. Without it, every verdict is ABSTAIN, the two-row counterexample being (c); Appendix A traces the closure on this schema. The copy holds on 100% of shipped invoices, yet guessing copies by shared column name finds 1 of the 12 renamed copies in Chinook and Northwind, and matching against Schema.org types finds none.

Φ_0 (§5.2). An instance is *admissible* if it satisfies all declared keys, FKs and copies. Domains are infinite unless stated otherwise.

Composite, self-referencing and cyclic FKs are all allowed. A copy declaration is an *ownership* declaration: the referenced table owns the value and $r.a$ is its row-aligned copy. A shared domain supplies neither.

Definition 2 (Feature view)**.** A feature view J is a finite rooted tree whose nodes are labelled by relations, with the entity table E at the root. Every edge $u \to v$ binds all of $\mathrm{pk}(\mathrm{rel}(v))$, either to the attributes of an FK of $\mathrm{rel}(u)$ (a *lookup*) or to attributes of u and constants (a *keyed join*, e.g. a description table on $(code, lang{=}\texttt{'EN'})$). A view row of an instance I picks a tuple $\rho(v) \in I(\mathrm{rel}(v))$ for every node, consistently with the bindings (inner join). $V_J(I)$ is the set of view rows and $\mathrm{Pos}(J)$ the set of positions (v, a). We write $S \models_J X \to T$ if on every admissible I, any two rows of $V_J(I)$ that agree on X agree on T.

V_J is a select–project–join view in the classical sense (Klug & Price, 1982; Abiteboul et al., 1995): one training row per entity, features joined many-to-one. One-to-many children are not positions of V_J. They reach models only through aggregates.

Definition 3 (Task and verdicts)**.** A task $\tau = (S, J, T, F)$ has a target position $T = (root, t)$ with $t \notin \mathrm{pk}(E)$ and features $F \subseteq \mathrm{Pos}(J) \setminus (\{T\} \cup \{(root, k) : k \in \mathrm{pk}(E)\})$. A position L is an IDENTIFIER if $S \models_J L \to (root, k)$ for all $k \in \mathrm{pk}(E)$. L *leaks* T, verdict LEAK, if $S \models_J L \to T$ and L is not an identifier. F is a JOINT leak of T if $S \models_J F \to T$ and F does not determine the root key. $L \in F$ is *essential* if $S \not\models_J F \setminus \{L\} \to T$. Every other pair receives ABSTAIN.

Because the view is keyed by the root entity, three kinds of columns are excluded: root-key positions, IDENTIFIER features, and columns of tables that reference E (including their FK back to E, join handles). T is withheld and every other view position is available.

Example and practice. In Figure 1, E is Invoice and T is BillingCountry. CustomerId is not an identifier, as a customer has many invoices, and the copy $(f_{\mathrm{CustomerId}}, \mathrm{BillingCountry}, \mathrm{Country})$ keeps the schema in Φ_0 and makes CustomerId leak T. SQL DDL, SchemaPile, CTU and RelBench declare keys and FKs, WikiDBs FKs and a property per column. None declares copies, so copies enter as designer declarations or recorded assumptions.

3 Theory: what the declared schema decides

The closure. Let $\bar{J} \supseteq J$ be the least view in which every node u has an f-lookup child for each copy declaration (f, a, b) of $\mathrm{rel}(u)$. By (S3) it is finite, and restricting rows from $\bar{J}$ to J is a bijection on admissible instances (Lemma 1). Row equivalence $\equiv$ is the least equivalence on positions of $\bar{J}$ and constants closed under: **(E1)** $v.k \equiv u.\beta(k)$ for every edge $u \to v$ binding $k \in \mathrm{pk}(\mathrm{rel}(v))$ to $\beta(k)$. **(E2)** $u.a \equiv v.b$ for every copy (f, a, b) and f-edge $u \to v$. **(E3, merge)** If $u \neq v$, $\mathrm{rel}(u) = \mathrm{rel}(v) = R$ and $u.k \equiv v.k$ for all k of a declared key of R, then $u.a \equiv v.a$ for all $a \in \mathrm{att}(R)$. The closure $\mathrm{cl}(X)$ is the least set of $\equiv$-classes that contains $[x]$ for $x \in X$ and every constant class and is closed under the **key rule**: if $[v.k] \in \mathrm{cl}(X)$ for every k of a declared key K of $\mathrm{rel}(v)$, then $[v.a] \in \mathrm{cl}(X)$ for all $a \in \mathrm{att}(\mathrm{rel}(v))$. The key rule fires on whole keys only. We call J *satisfiable* if no $\equiv$-class contains two distinct constants, which is necessary and sufficient for $V_J(I) \neq \emptyset$ on some admissible I (Appendix A).

Theorem 1 (Decision)**.** *Let $S \in \Phi_0$, J a satisfiable feature view, $X \subseteq \mathrm{Pos}(J)$ and $T \in \mathrm{Pos}(J)$. The following are equivalent: (a) $[T] \in \mathrm{cl}(X)$; (b) $S \models_J X \to T$ over all instances; (c) $S \models_J X \to T$ over finite instances. If (a) fails, a finite admissible counterexample is constructed explicitly, with at most $2\|\bar{J}\| + \sum_R (m+1)^{|\mathrm{pk}(R)|}$ tuples, where $m = 2|\mathrm{Pos}(\bar{J})|$. FKs that label no edge of $\bar{J}$ are irrelevant, so cyclic and self-referencing FKs need no chase-termination argument. $\mathrm{cl}(X)$ is computable in $O(\|\bar{J}\| \log \|\bar{J}\|)$; Corollary 1 avoids materialising $\bar{J}$ by truncating (E3).*

Proof idea (Appendix A). Soundness is induction over the rules. For completeness, freeze two copies of every row class, equal on $\mathrm{cl}(X)$ and distinct elsewhere (the merge rule (E3) makes both satisfy every declared key). Complete dangling FKs with fresh values and one shared filler. Fill copy targets from the sinks of the copy graph upward. An executable checker certified all 58,975

non-implications it generated and found no soundness violation in 88,059 random and 2,462,911 exhaustively enumerated admissible instances (Appendix D). Finite domains change nothing, provided they are used only by attributes outside keys, FKs and join bindings, and copy-linked attributes share a domain. A binary label is the typical case: map T's two frozen values to two distinct elements, and everything else to a third (Remark 2).

Inherited and new. For keys alone, (a)⇔(b) is FD propagation to select–project–join views, polynomial with infinite domains (Klug & Price, 1982; Abiteboul et al., 1995; Fan et al., 2008); that part is inherited. What is new: copies and FKs, cyclic or not, do not interact with keys under (S1)–(S3), and finite and unrestricted implication coincide for view goals, known previously only for key and FK goals (Fan & Siméon, 2003).

Corollary 1 (Memoised closure, schema-polynomial at fixed merge depth)**.** *A fixpoint over the copy* DAG*, memoised on (relation, known-attribute set), decides the same verdicts without materialising $\bar{J}$, in time $O\big((\|J\|\beta^{\ell} + \|S\|2^{w}) \cdot \mathrm{poly}(\|S\|)\big)$: polynomial in $\|S\| + \|J\|$ at bounded arity w and fixed merge depth ℓ, applying (E3) only among the explicit nodes (Appendix L).*

Dropping merges only removes derivations, so the truncation under-approximates cl and can only abstain: at $\ell = 2$ it reproduces every verdict on 48,787 pairs and decides Theorem 1's blow-up family at $k = 200$ in 1.1 ms, where $\bar{J}$ would have $3.2 \cdot 10^{60}$ nodes. The exact complexity on Φ_0 is open, with a single-exponential upper bound and no lower bound.

Corollary 2 (Authority characterization)**.** *Let $S \in \Phi_0$ and let L be a feature that is not an identifier. Then $S \models_J L \to T$ iff either (equality) $L \equiv T$ (a join binding, a declared copy chain or a merge), or (constant) $[T]$ is a constant class, or (lookup) there are a node w of $\bar{J}$ with $T \equiv w.a$ for some a, an* authority *for T, and a declared key K of $\mathrm{rel}(w)$ with $[w.k] \in \mathrm{cl}(\{L\})$ for all $k \in K$. The lookup crosses tables if $K = \mathrm{pk}(\mathrm{rel}(w))$ and is a reverse lookup if K is a secondary key. Every derivation is a DAG of whole-key authority firings rooted at $[L]$.*

Proposition 1 (Co-key membership is not a determination)**.** *Let $S \in \Phi_0$, R a relation, $t \in \mathrm{pk}(R)$, and read R : as "on every admissible R". Then $S \not\models R : (\mathrm{att}(R) \setminus (\{t\} \cup \mathrm{Free}(R) \cup \mathrm{Cp}(R))) \to t$, where $\mathrm{Free}(R)$ are the free attributes and $\mathrm{Cp}(R)$ the copy targets of R. In particular $S \not\models R : (\mathrm{pk}(R) \setminus \{t\}) \to t$, and no co-member of a composite key determines another. Composite keys contribute determinations only through whole keys: if $[w.k] \in \mathrm{cl}(F)$ for every k of a declared key of $\mathrm{rel}(w)$ and $T \equiv w.a$, then $S \models_J F \to T$, which is a JOINT leak if F does not determine the root key. A member $L \in F$ is essential iff $[T] \notin \mathrm{cl}(F \setminus \{L\})$. It is inessential whenever $[L] \in \mathrm{cl}(F \setminus \{L\})$, but not only then.*

Proposition 2 (One-to-many aggregates)**.** *Let C reference E through $C[Y] \subseteq E[\mathrm{pk}(E)]$ with $Y \subsetneq \mathrm{pk}(C)$, let c be a free attribute of C such that every declared secondary key of C containing c contains another free attribute of C, let φ be any aggregate of c over an entity's children, and assume $S \not\models_J \emptyset \to T$. Then there is a finite admissible instance with two entities, each with exactly one child, that agree on φ and differ on T, so the declared constraints do not imply $\varphi \to T$, even restricted to entities that have a child. Without the condition on secondary keys this fails: under $\mathtt{UNIQUE}(c)$, $\min(c)$ determines T on entities that have a child. Conversely, a child attribute declared as a copy of T determines T through every member-returning aggregate.*

Likewise, a description-table text determines its code iff a secondary key inside $\{lang, txt\}$ is declared (Proposition 4). Metadata that declares types but no owner declares no authority, so no derivation exists (Corollary 2).

Corollary 3 (Completeness and forced abstention)**.** *Call a procedure* schema-only *if its output depends only on (S, J, F, T), and* sound *on Φ_0 if it flags (L, T) only when L leaks T (Definition 3). (i) The closure detector D^* (flag iff $[T] \in \mathrm{cl}(\{L\})$ and L is not an identifier) is sound. (ii) (Completeness) Every pair that a sound schema-only procedure flags is flagged by D^*, so D^* is the greatest such procedure. (iii) For every non-identifier pair D^* does not flag, a finite admissible instance on which L does not determine T is constructed explicitly. In particular, if S and S' are isomorphic tasks and the data of either violate $L \to T$, every sound procedure abstains on both.*

Part (ii) restates Theorem 1's completeness for procedures. The substance is (iii), a checkable certificate for every forced abstention. Abstention is not safety: data may hold determinations the schema does not imply (§5.1).

Corollary 4 (Monotonicity and locality)**.** *Adding declarations never removes an implied determination. A declaration changes the verdicts of a task only if it is a key, an FK edge or a copy declaration of a relation that occurs in the task's copy-closed view.*

Theorem 2 (Undecidability, one declarable construct beyond Φ_0)**.** *Let Φ^+ extend Φ_0 by FKs that reference a declared* `UNIQUE` *superkey of the primary key (*`PRIMARY KEY(X)`*,* `UNIQUE(X,Y)`*,* `FOREIGN KEY(A,B) REFERENCES R'(X,Y)`*), equivalently, by copy declarations into key attributes (Remark 1). Deciding $S \models_J F \to T$ for $S \in \Phi^+$ is undecidable, for finite and for unrestricted implication, already when J is the root alone (for feature sets F, whether single features suffice is open).*

Corollary 5 (Boundary)**.** *Φ_0 (decidable, Theorem 1) and Φ^+ (undecidable) differ in one SQL-declarable construct: whether an FK may reference a non-minimal* `UNIQUE` *set. The intermediate case turns on one condition. Read (S2) relative to the referenced key rather than the primary key, and call the result (**S2$^+$**). Under it, FKs may reference* any *declared key and Theorem 1 holds verbatim, construction and certificates unchanged, so finite and unrestricted implication still equal the closure. It also forces referenced keys to be minimal, and the construct of Theorem 2 violates it (Theorem 5, Appendix A.1). (S2$^+$) is therefore sufficient for decidability, and Algorithm 1's restriction to primary-key lookups is conservative rather than required; whether (S2$^+$) is necessary is open (Appendix A.1). The full results map is Table 3 (Appendix B).*

Proposition 3 (Colour refinement is not enough)**.** *There are S_A (an FK 6-cycle) and S_B (two FK 3-cycles) in Φ_0 whose labelled schema graphs receive identical stable colourings under colour refinement, with identically defined tasks (follow the FK three times from C_0, does the reached d determine the root's d?), such that the declared determination holds in S_B and fails in S_A. Call a schema-only procedure, operating on the schema graph alone,* colour-refinement-invariant *if its output on (S, J, L, T) depends only on the stable colours of the labelled schema graph of S at the root relation, at the attributes of L and T, and along the walk of FK edges that defines J. Hence no colour-refinement-invariant procedure, message passing over schema graphs (Xu et al., 2019; Morris et al., 2019; Scheidt & Schweikardt, 2025) included, is a sound and complete schema-only detector. Only schema-graph readers at colour-refinement power are ruled out; message passing over the* data *graph, and higher-order or positionally encoded readers, are not.*

A declared determination also makes $\mathcal{V}$-usable information (Xu et al., 2020; Ethayarajh et al., 2022) maximal, but not conversely (Appendix A).

4 THE DETECTOR

Algorithm 1 (Appendix K) copy-closes J, closes each feature with a witness pointer for every key firing, and emits one of the four verdicts. It computes exactly D^* of Corollary 3, with $O(\|\bar{J}\| \log \|\bar{J}\|)$ preprocessing and $O(\|\bar{J}\|)$ per closure. Every LEAK carries its derivation: the authority, the key, and the declarations used. Every ABSTAIN carries a finite counterexample database of two rows per distinct row class of $\bar{J}$. Both can be audited. Adapters read declared keys and FKs from each corpus. Constructs outside Φ_0, such as FKs into `UNIQUE` sets, are ignored or kept as flagged lookups, which by Corollary 4 can only lose determinations (Appendix C). The implementation agrees with Proposition 6's specification on every verdict tested, including IDENTIFIER, JOINT and essential-member cases. A pass over all 22,989 SchemaPile schemas and their 1,137,685 candidate targets under three copy sources takes 107 s with 4 processes (median 0.46 ms per schema) and reads 0 rows.

Sample-verified declarations. A declaration the schema does not carry can still be verified on a sample, cheaply and with a bound, though the zero-violation rule costs recall (0.620 to 0.255, Appendix M).

Theorem 3 (Sample-verified declarations)**.** *Propose K candidate copy declarations from the schema alone and test each on the* same *uniform sample of n root rows, accepting a candidate iff it has zero sampled violations. If $n \geq \lceil \ln(K/\delta)/\varepsilon \rceil$, then with probability $\geq 1-\delta$ every accepted declaration has true violation rate $\leq \varepsilon$, and every closure-certified LEAK whose derivation uses m accepted declarations holds on at least a $1 - m\varepsilon$ fraction of the root rows all m cover. The sample size depends on K only through $\log K$, and never on the number of features, tasks or (feature, target) pairs the accepted declarations go on to certify through the closure.*

A Clopper–Pearson variant accepts a declaration that holds only up to a certified defect rate, which is what a real ownership rule usually is: an overridable default rather than an exact equality (Appendix M).

Theorem 4 (Declarations make verification per-row)**.** *On the one-step lookup class (N root rows, D distinct foreign-key values, $D \leq \varepsilon N$), verifying K declarations on one shared sample costs $\Theta(\log(K/\delta)/\varepsilon)$ rows, with matching upper and lower bounds. Every tester that sees only the (F, T) projections of sampled root rows costs $\Theta(\sqrt{D^*}/\varepsilon)$, $D^* = \min(D, \varepsilon N/2)$.*

The gap is structural, and it prices two access models rather than two implemented methods. By Corollary 2 a certified determination has a witness position in $\bar{J}$, and checking a witness is a *unary* predicate that one sampled row refutes. Without declarations $F \to T$ is *pairwise*, refuted only by two rows colliding on F, a birthday bound, by the standard collision argument (Canonne, 2020; Paninski, 2008), in the row-sampling form of Cormode et al. (2009). The pairs the closure must abstain on are exactly those with no witness (Appendix N, Figure 8).

5 Experiments

We test the theory with Algorithm 1 on SchemaPile (22,989 schemas) (Döhmen et al., 2024), CTU (the 119 of its 145 databases that declare keys) (Motl & Schulte, 2015), RelBench v2 (23 tasks) (Gu et al., 2026), 5,094 WikiDBs databases (Vogel et al., 2024) and THE JOIN (639 databases, 10,612 tasks; Appendix V). Copies come from a stated source: C0 none (keys and FKs as declared); C1 a same-name, same-type column (on WikiDBs, the same property) copied along one FK; C2 two columns referencing the same key. Intervals are 95% bootstrap; Appendix Z maps every number to a script and output file.

5.1 One declaration, measured harm: RelBench's hand-removed columns

RelBench's designers remove 90 (task, column) pairs by hand (Table 1): 62 other tasks' labels, 19 same-row attributes and 9 free-text columns. Of these, 15 live on one-to-many children, outside the view. Only 7 determine the target in the rows: 2 copies, 2 lookups through check tables and 3 age bounds (Appendix G). As released, the schemas certify none of the 7. One ownership declaration per copy recovers the 2 copies, and Algorithm 1 also flags the item's FK to its header and the header key. Row-reading baselines do better here than Algorithm 1. On the 7, a probe reaches 4 at precision 0.80, HyFD 3 at 1.00, PPS and TANE all 7 at 0.58 and 0.03, Algorithm 1 2 at 0.50 with declarations and none without; a zero-row rule refusing another task's label gets 60 of the 90 and 2 of the 7 (Table 1). Two baselines do better still: a preliminary structural detector scoring declared copies directly reports 3 of the 90 at precision 1.00, and on CTU an $|\rho| > 0.3$ screen reaches F1 0.75 against the join-handle rule's 0.55 (Appendices G, J). The first is schema-only and simply better on this list; the rest read rows. What the closure adds is warrant: those baselines read 93.2M rows and return no certificate, and 15,891 of 22,989 schemas ship no rows for them to read. *Without hindsight,* a name-blind generator proposes the item-incoterms declaration and 10 sampled rows accept it (Theorem 3), reproducing the certified determination with nothing hand-written; results-position has no such analogue (Appendix M). Six blind language-model declarers, run in isolation, wrote declarations that mostly hold in the rows, yet none reaches a target and Algorithm 1 abstains on all 23 tasks. A language model asked directly which columns are illegitimate finds 24 of the 90 at precision 0.80 over that whole list, but every named run recognised RelBench, whose removal lists are public (Appendix G).

Do the removed columns matter? We re-insert removals into their official tasks, keep RelBench's temporal split and train a gradient-boosted model on the one-hop view (5 seeds; Table 2). Under the distributed metadata Algorithm 1 flags none of them. On rel-salt item-incoterms, the item's Incoterms is *not* owned by its header: the dataset's own paper says the field "can be defined independently on item and header level" (Appendix G). The field is an overridable default, one of the four declaration kinds below, so its certificate is a sampled defect bound rather than an exact equality. On 10,000 root rows, 7 violations bound the defect rate at 0.1314% with 95% confidence; the true rate is 0.0733%, and the bound has 0.985 empirical coverage over 200 seeds. Non-zero violations put this under Theorem 3's Clopper–Pearson variant rather than its zero-violation rule (Table 15). Algorithm 1 yields LEAK under either certificate. Re-inserting the copy raises held-out

			Alg. 1		read rows (100k sample)			
removed columns	n	det.	rel.	+2	probe	PPS	HyFD	TANE
other tasks' labels	62	2	0	1	1	3	0	21
same-row attributes	19	5	0	1	4	7	3	17
free text	9	0	0	0	0	0	0	1
all 90	90	7	0	2	5	10	3	39
determine T	7	7	0	2	4	7	3	7
other flags			0	2[a]	1	5	0	222
Algorithm 1 + blind declarations (6 isolated runs): 0/90, 0 det., 0 other flags[b]								

Table 1: RelBench v2's 90 hand-removed columns by reason for removal. det.: determine T in the rows (held-out lookup ≥ 0.95 or exact FD). Algorithm 1 as released (rel.; C0, C1) and with two copy declarations (+2). Probe ≥ 0.95, PPS ≥ 0.5. Precision is det. over all flags. [a]The item's FK to its header and the header key. [b]Appendix G.

task (metric)	re-inserted column(s)	CLEAN	LEAKY	Δ [95% CI]	placebo Δ	(a) / (b)	evid.
item-incoterms (acc.)	header Incoterms (FK copy)	71.8	99.9	$+28.0$ [$+27.8$, $+28.3$]	-0.2 to 0.0	– / LEAK	C
results-position (R^2)	positionOrder (row copy)	36.7	99.6	$+62.9$ [$+60.0$, $+66.0$]	0.0 to $+12.5$	– / LEAK *	C*
sales-incoterms (acc.)	items' Incoterms (1:N)	77.3	99.8	$+22.4$ [$+22.2$, $+22.7$]	0.0 to $+0.2$	– / –	M

Table 2: Re-inserting hand-removed columns into official RelBench tasks (temporal split, GBM, 5 seeds, 95% CI; Appendix G). Every Δ is computed before rounding. Placebo: Δ for non-removed columns of the same table; one of results-position's 2 moves R^2 by $+12.5$, against a null permutation control (Fig. 4). Algorithm 1 under (a) distributed metadata, (b) one added declaration. Evid.: C certified, M measured only. *Outside Φ_0; Theorem 1 does not cover it.

accuracy by 28.0 points, against near zero for matched header columns (Table 2). Item and header Incoterms agree on 99.93% of items. On rel-f1 results-position, positionOrder raises R^2 by 62.9 points, but certifying it needs a same-row equality, which lies outside Φ_0 and holds on 99.91% of rows. Over all 56 ordered pairs of rel-salt's removed attributes the closure certifies the one functional determination, abstains by proof on two one-to-many aggregates, and certifies no non-leaking cell (Proposition 2, Appendix W.2). RelBench's heterogeneous GNN is inflated as much or more, $+32.1$ against the GBM's $+28.0$ (Table 12).

Who writes the declaration. A dataset author declares ownership once in a sidecar beside the manifest, in one of four kinds (exact copy, overridable default, value map, same-row equality) with provenance and a verification block, or mines it from an ORM layer as §5.2's ERPs do; seven declarations, 0.30 per task, span RelBench v2's 90-column audit (Appendix O).

Outside the class. On rel-salt sales-incoterms, one of RelBench v2's own hand-removed columns, the leak enters through the mode of the child items' Incoterms, a one-to-many aggregate, inflating accuracy by $+22.4$ [$+22.2$, $+22.7$]; a heterogeneous GNN, whose message passing computes exactly such aggregates, loses $+32.8$ [$+32.3$, $+33.3$] to it against a null permutation control. CTU Accidents supplies a second official case via a fitted, not declared, aggregate, and two 4DBInfer cases behave alike, one (mag/venue) under a post-hoc 20-class restriction; all carry standalone associations Algorithm 1 cannot flag, and it abstains on all four (Appendix S). On sales-incoterms no generator proposes a single candidate, so the verifier never forms the hypothesis it would have to reject (Proposition 2, Appendix S). The total score of rel-ratebeer, an undeclarable sum of five removed sub-scores, is flagged by no sound schema-only method (Corollary 3).

5.2 WHAT DO PUBLIC EXPORTS DECLARE?

With keys and FKs only (C0), Algorithm 1 flags 124 of SchemaPile's 1,137,685 non-key targets. In every one, a key cycle routes the feature back to the target's own column. It flags none of CTU's 12,856 non-key columns, none of RelBench's 23 official tasks, and none of THE JOIN's 17,110 candidate targets, whose format has no field for a copy (Table 10, Appendices F and V). Authorities are declared: Algorithm 1 returns IDENTIFIER for 421,270 pairs on 115,520 SchemaPile targets. The copies tying a feature to an authority are not, and no target has a JOINT leak. Under C0, 60.9% of SchemaPile schemas (52.4% with an FK) and 63.9% of CTU databases lie in Φ_0; outside it the

closure is sound but not complete (Appendix C). Of the 1,137,685 targets, 381,560 lie inside Φ_0, where abstention is proved forced rather than silence, 81,836 of those also carrying an FK and rows; 106 of the 124 flags fall inside (Appendix F). WikiDBs is entirely inside it under C0 and 36.7% under C1, and RelBench's single-column keys are inside it throughout.

Vendors do declare ownership. Two open-source ERPs ship copies in machine-readable form: ERPNext's `fetch_from` (555 copies over 838 tables) and Odoo's stored `related` fields (169 over 655); a third, iDempiere, confirms it on a mined lower bound; all three are ORM-backed web ERPs (Appendix P). Algorithm 1 certifies no target in either under C0, ERPNext's dump carrying 0 FKs, and 313 and 55 under the vendors' own declarations (Appendices Q, P). In both systems we can inspect at both layers, it is the export that declares too little, not the source, so the census above measures what public exports expose rather than what source systems declare (Appendix Q). Relational foundation models are ranked on these corpora (Kumo, 2026; Wang et al., 2025; Yuan et al., 2025); across their 135 in-domain targets the closure certifies none, and a 39-check value audit of 4DBInfer adds none at its own value bar (Appendix R).

Exact-name C1 flags 40,524 SchemaPile targets whose leaks hold no more often than matched controls (4.5% of 1,201 against 5.2%). A trigram relaxation, C1f, calibrated on 130 vendor-declared renames, does reach copies C1 cannot express: 6.7% of 2,713 against 4.6% ($p = 0.005$). It also declares eight times the (S3) violations, taking a 300-database WikiDBs sample from 37.7% inside Φ_0 to 6.7%: adding declarations can cost the guarantee (Appendix W). C2, the strongest guess, gives 22.2% ($n = 144$) against 9.5% ($n = 179$), separating on the rerun sample too ($+14.8$pp, Appendix W), yet still leaving three proposed copies in four not holding. Better proposal is not authority; sample verification is.

A positive control, and determinations in the wild. We calibrate the evidence bar with planted copies: exact, recoded and 90%-noisy, 20 CTU tasks each, run through the identical pipeline. A Clopper–Pearson bar recovers 8 and 6 and rejects all noise; the strict bar (100 repeated groups) recovers *none*: a copy of a k-valued target yields at most k repeated groups, so the floor is unreachable whenever $k < 100$, whatever the data. Among 553 WikiDBs pairs, 103 are structural copies and 79 pass the Clopper–Pearson bar. An independent value-first search of the CTU pool returns 10 findings, none expressible in the declared grammar and none meant to be: same-table equalities, aggregates and confounds, which §2 places outside the class (Appendices T, X). Nobody had flagged these columns and the declarations come from that corpus's own metadata, so the WikiDBs findings are unprompted. Removing a confirmed feature costs more than ten accuracy points on 13 of the 67 tasks, and on 16 of the 67 the loss has an interval excluding zero, to a maximum of 69 points (Appendix U). Appendix Y tabulates every evidence bar.

5.3 The missing declaration is ownership

Corollary 2 says the missing determinations must come from a declared copy, i.e. ownership. WikiDBs, which declares FKs and a Wikidata property for 96.2% of columns but no owner, tests this. We test all 9.8M (FK column, target) units rooted at a single table. A unit is positive when the referenced table carries the target's value through the join (base rate 3.84%). Copying along an FK whenever the two share a property (C1) gives precision 5.3% [5.2, 5.3] at recall 85.6% (Figure 9). A learned ownership list applied schema-only to held-out databases raises it to 10.9% (C3), and to 61.2% [59.5, 63.0] at recall 19.0% with copies kept off FK columns (C3d, inside Φ_0). A wrong copy on an FK column propagates through every later hop. Applied blind to held-out databases it selects no better than removing any foreign-key source column, or than a sham list of the same size (Appendix U.1).

Declarations act locally (Corollary 4): an injected declaration changes only its own task's verdicts, and deleting SchemaPile's FK and `UNIQUE` declarations never creates a determination (Appendices F, G).

5.4 Does the detector compute the theorem's relation, on synthetic and on real schemas?

RelLeak-Synth draws 320 random schemas with planted leaks and hard negatives, scored against *data-true* labels; C0 flags nothing and the C1 errors fall where Corollary 3 permits. Among methods

that read no row only the closure computes the determination relation, exact on the checker's $66{,}589$ pairs against precision 0.158 and 0.281 for two schema heuristics. Against row-reading detectors Algorithm 1 scores a lower F1 (0.636 against 0.978), losing recall to forced abstention and precision to untrue guessed declarations, though a low-cardinality true copy defeats row-based confirmation at any data volume. RelLeak-Real plants 20 kinds of column into six real databases: every declared exact copy gets LEAK and every undeclared twin ABSTAIN, 13/13 each, at precision 1.00 and recall 0.18 (Appendices E, D, T).

6 Related work

Leakage detection. Leakage's prevalence and cost are documented (Kaufman et al., 2012; Kapoor & Narayanan, 2023; Rosenblatt et al., 2024); training on the test task is a different issue (Dominguez-Olmedo et al., 2025). Table 5 (Appendix B) positions the closure against the value-based and code-based families.

Relational learning. Relational deep learning controls temporal leakage with cutoff times (Alteryx, 2024; Fey et al., 2024; Robinson et al., 2024; Kolesnichenko et al., 2026). Relational foundation models build on the same declared keys (Ranjan et al., 2026; Dwivedi et al., 2026; Choi et al., 2025; Truong et al., 2025), so a schema-only screen can vet their task pools before features are assembled. Target-conditioned exclusion is done by hand (Wang et al., 2024; Gu et al., 2026), and Poole (2026) traces benchmark leaks to query construction. Corollary 2's lookup rule is the key–FK dependency Kumar et al. (2016) and Shah et al. (2017) use to decide when a join is *redundant*. The copy tying T to the referenced table turns that redundancy test into a verdict that a feature is *inadmissible*. The two are not interreducible: theirs concerns predictive sufficiency of a feature set, ours a single position's forced determination. Join inference (Cong et al., 2026) and schema corpora (Döhmen et al., 2024) are not target-conditioned. Inclusion-dependency discovery and instance-based schema matching propose cross-table correspondences directly; we evaluate both as candidate generators in Appendix W.1.

Dependency theory. FD propagation to select–project–join views is decidable (Klug & Price, 1982; Abiteboul et al., 1995; Fan et al., 2008) and undecidable for relational-algebra views (Klug, 1980). Implication of FDs and inclusion dependencies is undecidable (Chandra & Vardi, 1985; Mitchell, 1983), already for keys and FKs into several keys (Fan & Libkin, 2002; Fan & Siméon, 2003), which Theorem 2 re-encodes in SQL-declarable metadata. Non-interaction under (S1)–(S3) belongs to the separability line for key-based inclusion dependencies (Johnson & Klug, 1984; Calì et al., 2003); new are the copy declaration as an ownership primitive and the decidable island it opens between key-based and arbitrary inclusion dependencies, with Theorem 2 as its sharp edge. The closure is a counter-based Horn closure (Beeri & Bernstein, 1979), deciding from declarations as description logics decide from a TBox (Calvanese et al., 2005).

Inference control. Multilevel databases ask whether a cleared user can infer a sensitive attribute through FDs, MVDs and other constraints (Su & Özsoyoğlu, 1991; Qian et al., 1993), and disclosure monitors track the channels they open (Brodsky et al., 2000; Pappachan et al., 2024). Reading that attribute as the target gives our question (Appendix B).

7 Limitations and conclusion

Limitations and conclusion. A certified determination is a leak only if the feature is not available at prediction time, which we do not decide, and a stale declaration yields a false LEAK. One of our two positive real-benchmark declarations is hand-written, a hindsight sampling bounds but does not remove (Theorem 3); temporal, aggregate and procedural determinations lie outside Φ_0 (Appendix J); and completeness can fail on a small finite domain. Given an ownership declaration, a whole-key closure decides target leakage from the schema alone, soundly and completely, with a counterexample for every abstention. That declaration is almost never published: keys and foreign keys certify no benchmark leak and only 124 key cycles in 1.1M targets, while two ERPs record ownership internally and strip it on export. A few declared lines would decide a hand-curated removal list before a row is read.

Reproducibility statement

All proofs, with their assumptions, are in Appendix A. The executable checker that tests every theorem on random and exhaustively enumerated instances is described in Appendix D. Every public number in the paper maps to one script and one output file in Appendix Z. All corpora are public (SchemaPile, CTU, RelBench v2, WikiDBs, Chinook and Northwind), and the adapters that read their declared constraints are described in Section 4. The detector, the RelLeak-Synth and RelLeak-Real generators and scorers, the model-impact scripts, and the checker will be provided as anonymous supplementary material.

Ethics statement

We use only public corpora under their published terms. SchemaPile's shipped sample values contain personal-data placeholders that the corpus authors inserted. We compare values only for equality and report aggregates. No human subjects were involved beyond the authors' own annotation of schema columns. A schema-only detector reduces the need to read sensitive columns in order to screen them, but its abstentions are not guarantees of safety, and we say so wherever abstention is reported.

AI use statement

We used generative AI tools (large-language-model assistants operating as coding and writing agents) for the following tasks: running and orchestrating experiment code (corpus download and parsing, the detector, the synthetic benchmark, the baselines, and drafting and generating the tables and figures of this paper). We did not use generative AI to fabricate data or results: every number reported was produced by code that was run on public data and traces to an output file (Appendix Z). We reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims and artifacts produced with the aid of generative AI.

References

Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases*. Addison-Wesley, 1995.

Alteryx. Featuretools documentation: Handling time (cutoff times). `https://featuretools.alteryx.com/en/stable/getting_started/handling_time.html`, 2024. Accessed 2026-09-10.

Amazon Web Services. Amazon sagemaker data wrangler: Analyze and visualize (target leakage). `https://docs.aws.amazon.com/sagemaker/latest/dg/data-wrangler-analyses.html`, 2024. Accessed 2026-09-10.

Francisco Galuppo Azevedo, Clarissa Lima Loures, and Denis Oliveira Correa. Task scarcity and label leakage in relational transfer learning. 3rd DATA-FM Workshop at ICLR 2026, 2026.

Pablo Barceló, Egor V. Kostylev, Mikael Monet, Jorge Pérez, Juan Reutter, and Juan Pablo Silva. The logical expressiveness of graph neural networks. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=r1lZ7AEKvB`.

Catriel Beeri and Philip A. Bernstein. Computational problems related to the design of normal form relational schemas. *ACM Trans. Database Syst.*, 4(1):30–59, 1979. doi: 10.1145/320064.320066.

Samuele Bortolotti, Emanuele Marconato, Tommaso Carraro, Paolo Morettin, Emile van Krieken, Antonio Vergari, Stefano Teso, and Andrea Passerini. A neuro-symbolic benchmark suite for concept quality and reasoning shortcuts. In *Advances in Neural Information Processing Systems 37 (Datasets and Benchmarks Track)*, 2024. URL `https://papers.nips.cc/paper_files/paper/2024/hash/d1d11bf8299334d354949ba8738e8301-Abstract-Datasets_and_Benchmarks_Track.html`.

Ahmed Bouajjani, Javier Esparza, and Oded Maler. Reachability analysis of pushdown automata: Application to model-checking. In *CONCUR '97: Concurrency Theory, 8th International Conference*, volume 1243 of *Lecture Notes in Computer Science*, pp. 135–150. Springer, 1997. doi: 10.1007/3-540-63141-0_10.

Alexander Brodsky, Csilla Farkas, and Sushil Jajodia. Secure databases: Constraints, inference channels, and monitoring disclosures. *IEEE Trans. Knowl. Data Eng.*, 12(6):900–919, 2000. doi: 10.1109/69.895801.

Andrea Calì, Domenico Lembo, and Riccardo Rosati. On the decidability and complexity of query answering over inconsistent and incomplete databases. In *Proc. ACM PODS*, pp. 260–271, 2003. doi: 10.1145/773153.773179.

Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. Dl-lite: Tractable description logics for ontologies. In *Proceedings of the Twentieth National Conference on Artificial Intelligence (AAAI-05)*, pp. 602–607, 2005.

Clément L. Canonne. A survey on distribution testing: Your data is big. but is it blue? *Theory of Computing Library, Graduate Surveys*, (9):1–100, 2020.

M. A. Casanova, R. Fagin, and C. H. Papadimitriou. Inclusion dependencies and their interaction with functional dependencies. *Journal of Computer and System Sciences*, 28(1):29–59, 1984. doi: 10.1016/0022-0000(84)90075-8.

Ashok K. Chandra and Moshe Y. Vardi. The implication problem for functional and inclusion dependencies is undecidable. *SIAM Journal on Computing*, 14(3):671–677, 1985. doi: 10.1137/0214049.

Yizuo Chen and Adnan Darwiche. Identifying causal effects under functional dependencies. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Dongwon Choi, Sunwoo Kim, Juyeon Kim, Kyungho Kim, Geon Lee, Shinhwan Kang, Myunghwan Kim, and Kijung Shin. RDB2G-Bench: A comprehensive benchmark for automatic graph modeling of relational databases. In *Advances in Neural Information Processing Systems (Datasets and Benchmarks Track)*, volume 38, 2025. URL https://openreview.net/forum?id=ZbcUIxACQE.

Shir Chorev, Philip Tannor, Dan Ben Israel, Noam Bressler, Itay Gabbay, Nir Hutnik, Jonatan Liberman, Matan Perlmutter, Yurii Romanyshyn, and Lior Rokach. Deepchecks: A library for testing and validating machine learning models and data, 2022.

Tianji Cong, Yuanyuan Tian, Andreas Mueller, Rathijit Sen, Yeye He, Fotis Psallidas, Shaleen Deep, and H. V. Jagadish. Nexus: Inferring join graphs from metadata alone via iterative low-rank matrix completion. arXiv:2602.08186, 2026.

Graham Cormode, Lukasz Golab, Flip Korn, Andrew McGregor, Divesh Srivastava, and Xi Zhang. Estimating the confidence of conditional functional dependencies. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 2009.

Stavros S. Cosmadakis, Paris C. Kanellakis, and Moshe Y. Vardi. Polynomial-time implication problems for unary inclusion dependencies. *Journal of the ACM*, 37(1):15–46, 1990. doi: 10.1145/78935.78937.

Till Döhmen, Radu Geacu, Madelon Hulsebos, and Sebastian Schelter. Schemapile: A large collection of relational database schemas. *Proc. ACM on Management of Data (SIGMOD)*, 2(3):Article 172, 2024. doi: 10.1145/3654975.

Ricardo Dominguez-Olmedo, Florian E. Dorner, and Moritz Hardt. Training on the test task confounds evaluation and emergence. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=jOmk0uS1hl.

F. Drobnjaković, P. Subotić, and C. Urban. An abstract interpretation-based data leakage static analysis. In *Theoretical Aspects of Software Engineering (TASE), LNCS 14777*, volume 14777, pp. 109–126, 2024. doi: 10.1007/978-3-031-64626-3_7.

Devdatt Dubhashi and Desh Ranjan. Balls and bins: A study in negative dependence. *Random Structures & Algorithms*, 13(2):99–124, 1998.

Vijay Prakash Dwivedi, Sri Jaladi, Yangyi Shen, Federico Lopez, Charilaos I. Kanatsoulis, Rishi Puri, Matthias Fey, and Jure Leskovec. Relational graph transformer. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=2d3j6bt21A.

Nick Erickson, Lennart Purucker, Andrej Tschalzev, David Holzmüller, Prateek Mutalik Desai, David Salinas, and Frank Hutter. TabArena: A living benchmark for machine learning on tabular data. In *Advances in Neural Information Processing Systems (Datasets and Benchmarks Track)*, volume 38, 2025.

Kawin Ethayarajh, Yejin Choi, and Swabha Swayamdipta. Understanding dataset difficulty with $\mathcal{V}$-usable information. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 5988–6008. PMLR, 2022. URL https://proceedings.mlr.press/v162/ethayarajh22a.html. ICML 2022 Outstanding Paper.

Wenfei Fan and Leonid Libkin. On XML integrity constraints in the presence of DTDs. *Journal of the ACM*, 49(3):368–406, 2002. doi: 10.1145/567112.567117.

Wenfei Fan and Jérôme Siméon. Integrity constraints for XML. *Journal of Computer and System Sciences*, 66(1):254–291, 2003. doi: 10.1016/S0022-0000(02)00032-6. Preliminary version in PODS 2000, pp.23-34, DOI 10.1145/335168.335172.

Wenfei Fan, Shuai Ma, Yanli Hu, Jie Liu, and Yinghui Wu. Propagating functional dependencies with conditions. *Proc. VLDB Endow.*, 1(1):391–407, 2008. doi: 10.14778/1453856.1453901.

Matthias Fey, Weihua Hu, Kexin Huang, Jan Eric Lenssen, Rishabh Ranjan, Joshua Robinson, Rex Ying, Jiaxuan You, and Jure Leskovec. Position: Relational deep learning—graph representation learning on relational databases. In *International Conference on Machine Learning (ICML)*, volume 235, pp. 13592–13607, 2024. URL https://proceedings.mlr.press/v235/fey24a.html.

Floris Geerts and Juan L Reutter. Expressiveness and approximation properties of graph neural networks. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=wIzUeM3TAU.

Stefan Grafberger, Shubha Guha, Julia Stoyanovich, and Sebastian Schelter. MLINSPECT: A data distribution debugger for machine learning pipelines. In *Proc. 2021 Int. Conf. on Management of Data (SIGMOD, demo track)*, pp. 2736–2739, 2021. doi: 10.1145/3448016.3452759.

Léo Grinsztajn, Edouard Oyallon, and Gaël Varoquaux. Why do tree-based models still outperform deep learning on typical tabular data? In *Advances in Neural Information Processing Systems (Datasets and Benchmarks Track)*, volume 35, 2022.

Justin Gu, Rishabh Ranjan, Charilaos Kanatsoulis, Haiming Tang, Martin Jurkovic, Valter Hudovernik, Mark Znidar, Pranshu Chaturvedi, Parth Shroff, Fengyu Li, and Jure Leskovec. RelBench v2: A large-scale benchmark and repository for relational data. In *The Fourteenth International Conference on Learning Representations*, 2026.

H2O.ai. Data leakage and shift detection in Driverless AI. https://docs.h2o.ai/driverless-ai/1-10-lts/docs/userguide/leakage-shift-detection.html, 2024. Accessed 2026-09-10.

Marton Havasi, Sonali Parbhoo, and Finale Doshi-Velez. Addressing leakage in concept bottleneck models. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Xingyue Huang, Miguel Romero, Ismail Ceylan, and Pablo Barceló. A theory of link prediction via relational weisfeiler-leman on knowledge graphs. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://papers.nips.cc/paper_files/paper/2023/hash/3eceb70f47690051d6769739fbf6294b-Abstract-Conference.html.

Y. Huhtala, J. Kärkkäinen, P. Porkka, and H. Toivonen. Tane: An efficient algorithm for discovering functional and approximate dependencies. *The Computer Journal*, 42(2):100–111, 1999. doi: 10.1093/comjnl/42.2.100.

International Business Machines Corp. Preventing data leakage in automated machine learning. U.S. Patent 11,847,544 B2, 2023.

David S. Johnson and Anthony C. Klug. Testing containment of conjunctive queries under functional and inclusion dependencies. *Journal of Computer and System Sciences*, 28(1):167–189, 1984. doi: 10.1016/0022-0000(84)90081-3.

Sayash Kapoor and Arvind Narayanan. Leakage and the reproducibility crisis in machine-learning-based science. *Patterns*, 4(9):100804, 2023. doi: 10.1016/j.patter.2023.100804.

Shachar Kaufman, Saharon Rosset, Claudia Perlich, and Ori Stitelman. Leakage in data mining: Formulation, detection, and avoidance. *ACM Transactions on Knowledge Discovery from Data*, 6(4):Article 15, 2012. doi: 10.1145/2382577.2382579. Preliminary version in Proc. 17th ACM SIGKDD (KDD 2011), pp. 556–563.

Jyrki Kivinen and Heikki Mannila. Approximate inference of functional dependencies from relations. *Theoretical Computer Science*, 149(1):129–149, 1995. doi: 10.1016/0304-3975(95)00028-U.

Anthony Klug. Calculating constraints on relational expression. *ACM Trans. Database Syst.*, 5(3): 260–290, 1980. doi: 10.1145/320613.320615.

Anthony Klug and Rod Price. Determining view dependencies using tableaux. *ACM Trans. Database Syst.*, 7(3):361–380, 1982. doi: 10.1145/319732.319738.

Oleksii Kolesnichenko, Jakub Peleška, and Gustav Šír. Relational task generation language: A declarative specification framework for relational deep learning, 2026.

Arun Kumar, Jeffrey Naughton, Jignesh M. Patel, and Xiaojin Zhu. To join or not to join? Thinking twice about joins before feature selection. In *Proc. 2016 Int. Conf. on Management of Data (SIGMOD)*, pp. 19–34, 2016. doi: 10.1145/2882903.2882952.

Kumo. KumoRFM-2: Scaling foundation models for relational learning, 2026. URL https://arxiv.org/abs/2604.12596.

John C. Mitchell. The implication problem for functional and inclusion dependencies. *Information and Control*, 56(3):154–173, 1983. doi: 10.1016/S0019-9958(83)80002-3.

Christopher Morris, Martin Ritzert, Matthias Fey, William L. Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and leman go neural: Higher-order graph neural networks. In *AAAI*, volume 33, pp. 4602–4609, 2019. doi: 10.1609/aaai.v33i01.33014602. URL http://dx.doi.org/10.1609/aaai.v33i01.33014602.

Jan Motl and Oliver Schulte. The ctu prague relational learning repository. arXiv:1511.03086, 2015.

Yonatan Oren, Nicole Meister, Niladri S. Chatterji, Faisal Ladhak, and Tatsunori Hashimoto. Proving test set contamination in black-box language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=KS8mIvetg2.

Liam Paninski. A coincidence-based test for uniformity given very sparsely sampled discrete data. *IEEE Transactions on Information Theory*, 54(10):4750–4755, 2008.

T. Papenbrock and F. Naumann. A hybrid approach to functional dependency discovery. In *ACM SIGMOD*, pp. 821–833, 2016. doi: 10.1145/2882903.2915203.

Primal Pappachan, Shufan Zhang, Xi He, and Sharad Mehrotra. Preventing inferences through data dependencies on sensitive data. *IEEE Trans. Knowl. Data Eng.*, 36(10):5308–5327, 2024. doi: 10.1109/TKDE.2023.3336630.

David Poole. Why isn't relational learning taking over the world? *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(46):39763–39770, Mar 2026. ISSN 2159-5399. doi: 10.1609/aaai.v40i46.41332. URL http://dx.doi.org/10.1609/aaai.v40i46.41332.

Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. CatBoost: Unbiased boosting with categorical features. In *Advances in Neural Information Processing Systems*, volume 31, 2018. Precursor: defines target leakage / prediction shift in target statistics.

Xiaolei Qian, Mark E. Stickel, Peter D. Karp, Teresa F. Lunt, and Thomas D. Garvey. Detection and elimination of inference channels in multilevel relational database systems. In *Proc. IEEE Symp. on Security and Privacy*, pp. 196–205, 1993. doi: 10.1109/RISP.1993.287632.

Rishabh Ranjan, Valter Hudovernik, Mark Znidar, Charilaos I. Kanatsoulis, Roshan Reddy Upendra, Mahmoud Mohammadi, Joe Meyer, Tom Palczewski, Carlos Guestrin, and Jure Leskovec. Relational transformer: Toward zero-shot foundation models for relational data. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=rpPtgMC5s9.

Joshua Robinson, Rishabh Ranjan, Weihua Hu, Kexin Huang, et al. Relbench: A benchmark for deep learning on relational databases. In *NeurIPS Datasets and Benchmarks*, volume 37, 2024.

Matthew Rosenblatt et al. Data leakage inflates prediction performance in connectome-based machine learning models. *Nature Communications*, 15, 2024. doi: 10.1038/s41467-024-46150-w.

Benjamin Scheidt and Nicole Schweikardt. Color refinement for relational structures. In *Mathematical Foundations of Computer Science (MFCS), LIPIcs 345*, volume 345, pp. 88:1–88:19, 2025. doi: 10.4230/LIPIcs.MFCS.2025.88.

Vraj Shah, Arun Kumar, and Xiaojin Zhu. Are key-foreign key joins safe to avoid when learning high-capacity classifiers? *Proc. VLDB Endow.*, 11(3):366–379, 2017. doi: 10.14778/3157794.3157804.

Tzong-An Su and Gültekin Özsoyoğlu. Controlling FD and MVD inferences in multilevel relational database systems. *IEEE Trans. Knowl. Data Eng.*, 3(4):474–485, 1991. doi: 10.1109/69.109108.

Quang Truong, Zhikai Chen, Mingxuan Ju, Tong Zhao, Neil Shah, and Jiliang Tang. A pretraining framework for relational data with information-theoretic principles. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://openreview.net/forum?id=xNUNxRj2vJ.

Liane Vogel, Jan-Micha Bodensohn, and Carsten Binnig. Wikidbs: A large-scale corpus of relational databases from wikidata. In *NeurIPS Datasets and Benchmarks*, volume 37, 2024.

Minjie Wang, Quan Gan, David Wipf, et al. 4dbinfer: A 4d benchmarking toolbox for graph-centric predictive modeling on relational databases. In *NeurIPS Datasets and Benchmarks*, volume 37, 2024.

Yanbo Wang, Xiyuan Wang, Quan Gan, Minjie Wang, Qibin Yang, David Wipf, and Muhan Zhang. Griffin: Towards a graph-centric relational database foundation model. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, volume 267 of *Proceedings of Machine Learning Research*, 2025.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *ICLR*, 2019. URL https://openreview.net/forum?id=ryGs6iA5Km.

Yilun Xu, Shengjia Zhao, Jiaming Song, Russell Stewart, and Stefano Ermon. A theory of usable information under computational constraints. In *International Conference on Learning Representations (ICLR)*, 2020.

Chenyang Yang, Rachel A. Brower-Sinning, Grace A. Lewis, and Christian Kästner. Data leakage in notebooks: Static detection and better processes. In *Proc. 37th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, 2022. doi: 10.1145/3551349.3556918.

Yiwen Yuan, Zecheng Zhang, Xinwei He, Akihiro Nitta, Weihua Hu, Dong Wang, Manan Shah, Shenyang Huang, Blaž Stojanovič, Alan Krumholz, Jan Eric Lenssen, Jure Leskovec, and Matthias Fey. ContextGNN: Beyond two-tower recommendation systems. In *International Conference on Learning Representations (ICLR)*, 2025.

Ligeng Zhu, Zhijian Liu, and Song Han. Deep leakage from gradients. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

A PROOFS

Lemma 1 (Lossless lookups)**.** *For admissible I, restriction is a bijection $V_{\bar{J}}(I) \to V_J(I)$. Hence $S \models_J X \to T$ iff $S \models_{\bar{J}} X \to T$ for positions of J.*

Proof. Every edge of $\bar{J} \setminus J$ is an FK lookup. The referenced tuple exists (the FK is satisfied and non-null) and is unique (pk is a key). □

The closure on Figure 1, step by step. The copy $(f, \mathrm{BillingCountry}, \mathrm{Country})$ gives the root an f-lookup child of relation Customer in $\bar{J}$, alongside the Customer node already reached by the view's own FK. Both are f-children of the root, so (E1) equates each of their CustomerId positions with the root's CustomerId, and (E3) then merges the two nodes, CustomerId being Customer's whole declared key. Now take $X = \{\text{Invoice.CustomerId}\}$. Initially $\mathrm{cl}(X) = \{[\text{Invoice.CustomerId}]\}$ together with the constant classes. By (E1) that class is [Customer.CustomerId], so the key rule fires at the Customer node on the whole of $\mathrm{pk}(\text{Customer})$ and adds [Customer.a] for every attribute a, Country included. By (E2) the copy gives Invoice.BillingCountry ≡ Customer.Country, so $[T] \in \mathrm{cl}(X)$ and Algorithm 1 returns LEAK with that firing sequence as its derivation. Without the copy declaration (E2) is unavailable, no rule reaches $[T]$, and the two frozen rows of Figure 1(c) witness it: they satisfy every declared key and the FK, agree on every position of $\mathrm{cl}(X)$, and differ on T.

Proof of Theorem 1, (a)⇒(b). Fix admissible I and rows ρ_1, ρ_2 of $V_{\bar{J}}(I)$. *Claim 1:* $p \equiv q$ implies $\rho(p) = \rho(q)$ for every row ρ, by induction on the derivation of ≡: (E1) holds by the bindings. (E2) holds since $\rho(u)$ satisfies the copy declaration and its f-lookup is $\rho(v)$. (E3) holds since $\rho(u), \rho(v)$ are tuples of R that agree on a declared key, hence are equal. *Claim 2:* if $[p] \in \mathrm{cl}(X)$ and ρ_1, ρ_2 agree on X, then $\rho_1(p) = \rho_2(p)$, by induction on the closure. Constants agree. For a key-rule step, $\rho_1(v), \rho_2(v)$ agree on K and are therefore equal. By Claim 1, agreement on a class is well defined. (b)⇒(c) is trivial. □

Proof of Theorem 1, (c)⇒(a). Assume $[T] \notin \Gamma := \mathrm{cl}(X)$. *Step 1 (two frozen copies).* For each non-constant class γ and $k \in \{1, 2\}$ let $\nu_k(\gamma) = \langle \gamma, 1 \rangle$ if $\gamma \in \Gamma$ or $k = 1$, and $\langle \gamma, 2 \rangle$ otherwise. Constant classes take their constant. Let $t_k(v) = (\nu_k([v.a]))_a$ and $I_0(R) = \{t_k(v) : \mathrm{rel}(v) = R\}$. *Claim 3 (keys hold in I_0).* If $t_k(u), t_l(v)$ agree on a declared key K of R then, since values encode classes, $[u.\kappa] = [v.\kappa]$ for $\kappa \in K$, and if $k \neq l$ each such class is in Γ. If $u \neq v$, (E3) gives $u.a \equiv v.a$ for all a. If $k \neq l$ (whether or not $u = v$), every class of K at u lies in Γ, so the key rule fires at u and puts every $[u.a]$ in Γ, where $\nu_1 = \nu_2$. Hence $t_k(u) = t_l(v)$. *Claim 4 (copies of I_0-tuples hold).* For a copy (f, a, b) of $\mathrm{rel}(u)$, u has an f-child v in $\bar{J}$. By (E1) $t_k(v)$ carries the key $t_k(u)[X_f]$ and is the unique such tuple (Claim 3). By (E2) $t_k(u)[a] = t_k(v)[b]$. *Step 2 (FK completion).* While a tuple r and an FK $f = R[X] \subseteq R'[\mathrm{pk}(R')]$ lack a referenced tuple, add s with $s[\mathrm{pk}(R')] := r[X]$, a new value $\star$ in the FK source attributes of R' that are not in $\mathrm{pk}(R')$, fresh values in free attributes, and copy targets left open (joins, bindings and copies relate attributes of one domain. $\star$ and fresh values are chosen per domain). All key and FK values lie in $V_0 \cup \{\star\}$ (V_0 the values of I_0) and each (relation, key value) is added at most once, so the loop terminates. As I_0 has two tuples per node of $\bar{J}$ and $|V_0| \leq m := 2|\mathrm{Pos}(\bar{J})|$, and a created tuple is fixed by its relation together with its primary-key value in $(V_0 \cup \{\star\})^{|\mathrm{pk}|}$, I has at most $2\|\bar{J}\| + \sum_R (m+1)^{|\mathrm{pk}(R)|}$ tuples. *Step 3.* Fill each open copy target $s.a$ of a copy (g, a, b) with $g(s)[b]$, processing relations from the sinks of the copy graph upward, which is well founded by (S3). *Claim 5 (I is admissible).* Primary keys: Step 2 adds a tuple only when no tuple with that primary-key value is already present, so an added tuple carries a primary-key value absent from I_0 and from the tuples added before it, and I_0 satisfies

them by Claim 3. Secondary keys: each contains a free attribute in which added tuples carry distinct fresh values (S2). FKs are closed by Step 2. Copies hold for I_0-tuples by Claim 4 (their lookups are unchanged because no added tuple repeats a key) and for added tuples by Step 3. *Claim 6.* $\rho_k(v) := t_k(v)$ are rows of $V_{\bar{J}}(I)$ that agree on X (classes in Γ) and differ on T. Restrict to J by Lemma 1. *Satisfiability.* If an $\equiv$-class contains two distinct constants, Claim 1 gives $V_J(I) = \emptyset$ for every admissible I. Otherwise, Steps 1–3 with $\Gamma = \mathrm{cl}(\emptyset)$ (constant classes take their constant) yield an admissible I containing the row ρ_1. So satisfiability is necessary and sufficient for a non-empty view. □

Proof of the complexity bound. $\bar{J}$ is built by following the acyclic copy graph. Let β be the largest number of distinct foreign keys along which one relation declares copies, and let h be the height of the copy graph. Each node receives at most β closure children and every closure path follows the copy graph. Hence at most $\sum_{i=1}^{h} \beta^i$ nodes are attached below each node of J, and for $\beta = 1$ they form a path through distinct relations, so $\|\bar{J}\| \leq \|J\|(1 + \|S\|)$. The bound is attained by the chain $R_i(k, p_a, p_b, a, b)$, $i < h$, with primary key k, FKs $f_a = R_i[p_a] \subseteq R_{i+1}[k]$ and $f_b = R_i[p_b] \subseteq R_{i+1}[k]$ and copies (f_a, a, a), (f_b, b, a): for the one-node view $\{R_0\}$, $\bar{J}$ has $2^{h+1} - 1$ nodes. (E1)–(E2) are union–find operations. (E3) is congruence closure (each node is a term whose arguments are its key classes), in $O(n \log n)$. The key rule is Horn closure with counters (Beeri & Bernstein, 1979), linear in $\sum_v \sum_K |K|$, at most $\|\bar{J}\|$ times the total size of the declared keys of a single relation, a per-declared-key schema factor that the bound stated in the theorem absorbs. All single-feature verdicts need $|F|$ closures. The essential members of F need $|F| + 1$. □

Proof of Corollary 2. By Theorem 1, $S \models_J L \to T$ iff $[T] \in \mathrm{cl}(\{L\})$. The class $[T]$ enters the closure either initially ($[T] = [L]$ or $[T]$ a constant class), or through a key-rule step at a node w with a member of $[T]$ among its positions. That key-rule step's premises are all classes of a whole declared key of $\mathrm{rel}(w)$. Unfolding the premises yields the DAG. The hypothesis that L is not an identifier is a presentational restriction: for an identifier the key rule fires at the root, so the lookup case applies with w the root and the biconditional holds there too. □

Proof of Proposition 1. We build a finite admissible instance with two R-tuples that agree outside $\{t\} \cup \mathrm{Free}(R) \cup \mathrm{Cp}(R)$ and differ on t. Let r_1 take pairwise distinct values on all attributes of R that are not copy targets, and let r_2 agree with r_1 except that $r_2.t$ and $r_2.w$ for *every* $w \in \mathrm{Free}(R)$ take new values. Perturbing all of $\mathrm{Free}(R)$, not one witness, is what the next paragraph needs and is permitted because the antecedent excludes $\mathrm{Free}(R)$. If $\mathrm{Free}(R) = \emptyset$ then R declares no secondary key, since by (S2) every secondary key contains a free attribute, so no secondary-key obligation arises and perturbing $r_2.t$ alone suffices. The copy targets of r_1, r_2 are left open. Starting from $\{r_1, r_2\}$, run Step 2 of the proof of Theorem 1. While a tuple r and an FK $f = R_1[X] \subseteq R'[\mathrm{pk}(R')]$ lack a referenced tuple, add s with $s[\mathrm{pk}(R')] := r[X]$, $\star$ in the other FK attributes of R', new values in its free attributes, and open copy targets. Every primary-key and FK value lies in the finite set of values of r_1, r_2 plus $\star$ (copy targets are neither key nor FK attributes by (S3)). A tuple is added only for a key value not yet present, so this terminates. Then fill every open copy target $s.a$ of a copy (g, a, b) with $g(s)[b]$, relations in reverse topological order of the copy graph (S3). The instance is admissible: r_1, r_2 differ on $t \in \mathrm{pk}(R)$ and every added tuple carries a primary-key value that was absent, so primary keys hold. Every secondary key contains a free attribute (S2), on which all tuples carry pairwise distinct values. FKs hold by construction. Copies hold because filling changes no attribute used by a lookup. Since $t \in \mathrm{pk}(R)$, t is not a copy target, so r_1, r_2 agree on $\mathrm{pk}(R) \setminus \{t\}$ and differ on t. The joint statement is Corollary 2 applied to F. For the last sentence, take $E(e, p, q, t)$, $A(p, q, x)$, $B(b, y)$ with primary keys e, (p, q), b, FKs $f = E[p, q] \subseteq A[p, q]$ and $h = A[q] \subseteq B[b]$, and copies (f, t, x), (h, x, y). Then $\{p, q\}$ jointly leaks t, q alone determines t through B, so p is inessential, yet q does not determine p. □

Proof of Proposition 2. Hypothesis (one-to-many child). C is a one-to-many child of E in the strict sense: C is the target of no foreign key of S and labels no node of $\bar{J}$. This is what *one-to-many child* is read to mean throughout: a child table is reached from E only in the reverse direction of its own foreign key, and the nodes of $\bar{J}$ are reached by lookups and keyed joins. Take the two rows ρ_1, ρ_2 of the counterexample of Theorem 1 for $X = \emptyset$. They differ on T, so their root tuples differ on some attribute of $\mathrm{pk}(E)$. Add one child r_i per entity with $r_i[Y] := \rho_i(root)[\mathrm{pk}(E)]$, the same new value γ in c, and new pairwise distinct values in all other attributes except copy targets. Complete FKs

and fill copies as in Steps 2–3. The new values and the placeholder $\star$ of Step 2 are taken outside the frozen root key values. By the hypothesis, $\bar{J}$ has no node of relation C, so I_0 contains no C-tuple, and no foreign key targets C, so Step 2 creates none either: the C-tuples of the instance are exactly r_1, r_2. Their Y-values are the two root key values, which differ, so each entity has exactly one child. The children have distinct primary keys (they differ on Y). Every secondary key of C contains a free attribute other than c (by (S2) and the hypothesis), on which the children carry distinct new values. No tuple references a child, again by the hypothesis, and existing lookups are unchanged. Every aggregate of c sees $\{\gamma\}$ for both entities while T differs. Under $\texttt{UNIQUE}(c)$ a value of c identifies the child and, through Y, its entity. A child copy of T determines T by the copy declaration. □

Proposition 4 (Description tables)**.** *Let $TXT(code, lang, txt, \dots)$ have primary key $(code, lang)$, no FKs and no copy declarations, with txt free, and let J consist of the root and one node v of TXT reached by a keyed join binding $code$ to a root attribute and $lang$ to a constant c. Then $S \models_J v.txt \to v.code$ iff TXT declares a secondary key contained in $\{lang, txt\}$, i.e. $\texttt{UNIQUE}(lang, txt)$ or $\texttt{UNIQUE}(txt)$.*

Proof. The classes of $v.txt$ and $v.lang$ are $\{v.txt\}$ and $\{v.lang, c\}$. This is because txt is bound by no edge and occurs in no copy, and v has no closure children. No other node has relation TXT unless the root's copy closure reaches it through an FK into TXT, whose node has a key class of root attributes, not $\equiv c$. So (E3) does not merge it with v. Hence $\mathrm{cl}(\{v.txt\})$ fires a key rule only at a node all of whose key classes lie in $\{[v.txt], [v.lang]\}$, i.e. at v for a declared key $K \subseteq \{lang, txt\}$. The primary key contains $code$. If such K exists, the key rule adds $[v.code]$. Otherwise, the closure is $\{[v.txt]\}$ plus constant classes. Apply Theorem 1. □

Proof of Corollary 3. (i) is Theorem 1(a⇒b) and (iii) is Theorem 1(c⇒a). (ii): let P be sound and flag (L, T). Then L is not an identifier and $S \models_J L \to T$, so $[T] \in \mathrm{cl}(\{L\})$ by Theorem 1(c⇒a), i.e. D^* flags (L, T). Since D^* is itself sound and schema-only by (i), it is the greatest such procedure. For the last sentence: a violating instance shows $S \not\models_J L \to T$, implication is invariant under isomorphism, and a sound procedure flags only implied pairs. The domination in (ii) appeals to soundness alone, so it holds of any sound procedure; the schema-only restriction is used only in the last sentence, where dependence on (S, J, F, T) alone is what carries the verdict across the isomorphism. □

Proof of Corollary 4. Every rule defining $\equiv$ and cl is monotone in the declarations, and by Theorem 1 implication depends only on the keys, lookups and copies of relations in $\bar{J}$. The copy-closed view meant in the statement is the one taken after the declaration is added, a copy declaration being able to enlarge $\bar{J}$ itself. □

Proof of Theorem 2. Fan & Libkin (2002, Lemma 3.2) prove that implication of keys by keys and foreign keys is undecidable over finite instances (for both finite and unrestricted implication see Fan & Siméon (2003, Thm. 3.5)). The proof is by reduction from implication of an FD $\theta = R_\theta : X \to y$ by FDs and inclusion dependencies (Chandra & Vardi, 1985; Mitchell, 1983). Take every original relation's key to be $\mathrm{att}(R)$. Their encoding of an FD $R : X \to Y$ ($X \cap Y = \emptyset$) introduces a fresh R_{new} with keys X and XY, the FK $R[XY] \subseteq R_{new}[XY]$, and an FK from R_{new} back to R. We keep $R_{new}(X, Y)$ with $\mathrm{pk}(R_{new}) = X$ and $\texttt{UNIQUE}(XY)$ and drop the back-FK. Dropping it rests on a standing assumption about the cited construction, which is taken on trust here and not reproved: in the proof of their Lemma 3.2 the back-FK from R_{new} to R is used only for the goal gadget (direction $\Sigma' \models \neg\varphi_1 \Rightarrow \Sigma \models \neg\theta$), so an occurrence of that FK in any other gadget of their argument falsifies the assumption and with it this reduction; the remaining steps below are self-contained. For the other FD gadgets $R_{new} := \pi_{XY}(I)$ satisfies the key X whenever $I \models R : X \to Y$, while φ_1, φ_2 alone give $R : X \to Y$. Now Y is free in R_{new}, so (S2) holds, and $R[XY] \subseteq R_{new}[XY]$ is the only declaration outside Φ_0. The inclusion-dependency gadget $R_1[X] \subseteq R_{new}[Y]$, $R_{new}[YZ] \subseteq R_2[YZ]$ ($\mathrm{pk}(R_{new}) = Y$, $Z = \mathrm{att}(R_2)$) and the goal gadget ($\mathrm{pk}(R_{\theta,new}) = Xy$, FKs $R_\theta[Xy] \subseteq R_{\theta,new}[Xy]$ and $R_{\theta,new}[\mathrm{att}] \subseteq R_\theta[\mathrm{att}]$) use only FKs into primary keys and declare no secondary key. Both directions of their instance translation preserve finiteness, and implication of FDs by FDs and INDs is undecidable for finite and for unrestricted instances (Chandra & Vardi, 1985; Mitchell, 1983), so the reduction yields both. They prove $\Sigma \models \theta$ iff $\Sigma' \models$ "X is a key of $R_{\theta,new}$", i.e., since Xy is its key, iff $\Sigma' \models R_{\theta,new} : X \to y$. Add

$E(e, X', y')$ with $\mathrm{pk}(E) = \{e\}$ and the FK $E[X'y'] \subseteq R_{\theta,new}[Xy]$, and let $J = \{E\}$, $F = X'$, $T = (E, y')$. If $\Sigma' \models X \to y$, two E-rows that agree on X' reference tuples that agree on X, hence on y. Conversely, given a counterexample for $X \to y$ with tuples u_1, u_2, add $e_i \mapsto u_i$. No FK references E, so the instance stays admissible and violates $X' \to y'$. T is not a key attribute and F excludes the root key. Corollary 5 follows because the encoding uses exactly one construct outside Φ_0. □

Remark 1 (Copy form)**.** An FK $R[XY] \subseteq R'[XY]$ with $\mathrm{pk}(R') = X$ holds iff $R[X] \subseteq R'[X]$ holds and every $r \in R$ satisfies $r.y = f(r).y$ for $y \in Y$, so the construct is a set of copies into (key) attributes of R. To keep one copy source per attribute, route the i-th FD on R through a twin $D_i(\mathrm{att}(R))$ with $\mathrm{pk}(D_i) = \mathrm{att}(R)$ and FKs $D_i[\mathrm{att}] \subseteq R[\mathrm{att}]$, $R[\mathrm{att}] \subseteq D_i[\mathrm{att}]$. $D_i = R$ in every admissible instance.

Proof of Proposition 3. Let $C_i(k, d)$ have primary key k and FK $C_i[k] \subseteq C_{i+1}[k]$ (identifying FKs). S_A is one 6-cycle, S_B two 3-cycles. All vertices of both labelled schema graphs have the same degree profile, so colour refinement stabilizes with identical colourings. In S_B the depth-3 lookup from C_0 returns to a node of relation C_0 whose key is $\equiv$ the root key, so (E3) merges it with the root and the reached $d \equiv$ the root's d. In S_A the reached node has relation C_3, nothing merges, and Theorem 1 gives a counterexample. The two tasks are given by the same walk (root colour, three FK edges of one colour, attribute d), so every colour-refinement-invariant procedure returns the same output on both, and it is wrong on one of them. The checker confirms identical colour histograms and the two verdicts, with a certificate for the negative one. □

Proposition 5 (Bridge to $\mathcal{V}$-usable information)**.** *Let T be discrete and $\mathcal{V}$ a predictive family in the sense of Xu et al. (2020) (so it satisfies optional ignorance) that contains the lookup predictors $f[l] = \delta_{g(l)}$ for all maps g. If $S \models_J L \to T$ then, for every admissible I and every distribution on $V_J(I)$, $H_{\mathcal{V}}(T \mid L) = 0$, so $I_{\mathcal{V}}(L \to T) = H_{\mathcal{V}}(T)$ is maximal (Ethayarajh et al., 2022). The converse fails.*

Proof. On every row of $V_J(I)$, $T = g(L)$ for the function g given by the FD (no finiteness is needed), so the lookup predictor has log-loss 0, and $H_{\mathcal{V}} \geq 0$. Converse: in $E(e, l, t)$ with no declaration linking l and t, the instance $\{(1, a, x), (2, b, y)\}$ has $I_{\mathcal{V}}(l \to t) = H_{\mathcal{V}}(t)$ although $S \not\models l \to t$. □

A family containing the lookup predictors of all maps makes $H_{\mathcal{V}}$ the Shannon conditional entropy, so the proposition is a consistency check on the semantics rather than a characterization of usable leakage; the informative regime is a restricted $\mathcal{V}$, under which a declared determination need not be usable.

Remark 2 (Finite domains)**.** Soundness needs no assumption on domains. Suppose no secondary keys are declared and every attribute ranges over one domain D (it suffices that attributes linked by an FK or a copy share a domain, counting per domain). The construction then uses at most $2m$ frozen values, $\star$ (also in the free attributes of added tuples, which is safe without secondary keys), and the k constants of the view. Here m is the number of non-constant row classes of J. Completeness holds whenever $|D| \geq 2m + 1 + k$. A per-attribute bound does not suffice: with $E(e, x, t)$, $P(p)$, FK $E[x] \subseteq P[p]$ and $\mathrm{dom}(P.p) = \{0\}$, every instance has $x = 0$, so $\emptyset \to x$ holds, but $[x] \notin \mathrm{cl}(\emptyset)$. Below that bound it fails even without constant selections: for $R(b, c, fk)$ with key b and a self-referencing FK $fk \to b$ over a two-element domain, the lookup's fk determines the root's fk on every instance, but not in the closure. Fan et al. (2008) show coNP-completeness for SC views with general FD sources. Whether hardness persists for key-only sources and constant-free FK views is open.

Finite domains off keys and FKs (binary targets). Suppose every finite domain is used only by attributes that lie in no declared key (primary or `UNIQUE`), in no FK and in no join binding, and that attributes linked by a copy share a domain. Then Theorem 1 holds unchanged. In particular, a binary target that is a label column in no key or FK is decided completely. Take the instance of the proof of Theorem 1 and, on each finite domain D fix $d_0 \in D$ (and $d_1 \in \mathrm{dom}(T) \setminus \{d_0\}$) and replace every value by d_0, except that $\langle [T], 2 \rangle$ becomes d_1. Keys, FKs and bindings read no such attribute, so Claims 3–5 are unaffected. Copies relate attributes of one domain and the replacement is a function of the value, so they still hold. The two rows still agree on $\mathrm{cl}(X)$ and differ on T. We

re-verified this value map on 25,548 single-feature non-implications (two runs of 3,000 random Φ_0 tasks whose target's copy-connected attributes avoid keys and FKs; one binary domain; 0 failures). When a finite domain meets a key or an FK, completeness can fail, as the two examples above show.

Remark 3 (FKs into minimal secondary keys)**.** If FKs may reference any declared key and (S2$^+$) holds, then the closure, unrestricted implication and finite implication all coincide (Theorem 5, proved in Appendix A.1). Algorithm 1 ignores such FKs, which is sound by Corollary 4, though Theorem 5 shows it need not.

A.1 FKs into secondary keys: (S2$^+$) and finite implication

The class. Φ_0^+ has the declarations of Φ_0 except that an FK $f = R[X] \subseteq R'[K_f]$ may reference any declared key $K_f \in \mathcal{K}(R') = \{\mathrm{pk}(R')\} \cup \mathcal{U}(R')$. Write $\mathrm{Ref}(R')$ for the declared keys of R' that some FK references. (S3) is unchanged, and (S2) is replaced by

> **(S2$^+$)** for every $K' \in \mathrm{Ref}(R')$ and every $K \in \mathcal{K}(R')$ with $K \neq K'$ there is an attribute $w(K, K') \in K \setminus K'$ that is neither an FK source attribute of R' nor a copy target of R'.

When every FK references a primary key, $\mathrm{Ref}(R') = \{\mathrm{pk}(R')\}$ and (S2$^+$) says exactly that every secondary key contains a free attribute, which is (S2). Hence $\Phi_0 \subseteq \Phi_0^+$.

Lemma 2 (Containment)**.** *If S satisfies (S2$^+$) then no declared key of a relation is contained in a referenced key of that relation. In particular every referenced key is minimal, and the construct of Theorem 2 (`PRIMARY KEY`(X), `UNIQUE`(X, Y), an FK into (X, Y)) is excluded.*

Proof. If $K \subseteq K'$ with $K \neq K'$ and $K' \in \mathrm{Ref}(R')$, then $K \setminus K' = \emptyset$, so no witness exists. For the construct, $\mathrm{pk} = X \subsetneq (X, Y) = K'$. □

Theorem 5 (A sufficient condition on declared keys)**.** *Let $S \in \Phi_0^+$, J a satisfiable feature view, $X \subseteq \mathrm{Pos}(J)$ and $T \in \mathrm{Pos}(J)$. The following are equivalent: (a) $[T] \in \mathrm{cl}(X)$; (b) $S \models_J X \to T$ over all instances; (c) over finite instances. If (a) fails, Steps 1–3 of the proof of Theorem 1, unchanged, return a finite admissible counterexample with at most $2\|\bar{J}\| + \sum_R \sum_{K \in \mathrm{Ref}(R)} (m+1)^{|K|}$ tuples, where $m = 2|\mathrm{Pos}(\bar{J})|$.*

Proof. (a)⇒(b). The proof of Theorem 1 uses only that a lookup along f resolves to the unique tuple of R' whose K_f-value is $r[X]$, which holds because K_f is a declared key, and that (E3) and the key rule range over declared keys. Neither depends on $K_f = \mathrm{pk}(R')$. (b)⇒(c) is immediate.

(c)⇒(a). Assume $[T] \notin \Gamma := \mathrm{cl}(X)$ and run Steps 1–3 of the proof of Theorem 1: freeze two copies $t_1(v), t_2(v)$ of every node of $\bar{J}$ into I_0; complete every dangling FK $f = R[X] \subseteq R'[K_f]$ at a tuple r by one tuple s with $s[K_f] := r[X]$, the filler $\star$ in the other FK attributes of R', a globally fresh value in every other non-copy-target attribute, and copy targets left open; then fill copy targets from the sinks of the copy graph upward. Let V_0 be the finite set of values of I_0, with $\star \notin V_0$.

Claim T (termination). Every FK source position of every tuple of I carries a value in $V_0 \cup \{\star\}$. For I_0 tuples this holds by definition. For a created tuple s, an FK source position lies either in K_f, where it inherits a value of r at an FK source position (induction), or outside K_f, where it is $\star$, because a fresh value is written only at attributes that are not FK source attributes. Hence every key value an FK can demand lies in $(V_0 \cup \{\star\})^{|K|}$, and the loop creates at most one tuple per triple (relation, referenced key, key value), since it first tests whether one exists. The bound follows.

Claim K (every declared key holds in I). Let R' be a relation and $K \in \mathcal{K}(R')$. (i) Two I_0 tuples: Claim 3 of the proof of Theorem 1, which is stated for declared keys and so covers secondary keys. (ii) An I_0 tuple and a created tuple s, created for $K' \in \mathrm{Ref}(R')$: if $K = K'$, no tuple carried the value $s[K']$ when s was created, and I_0 is never extended; if $K \neq K'$, then by (S2$^+$) s carries a globally fresh value at $w(K, K') \in K$, since the witness is not an FK source attribute and hence not $\star$, and not a copy target and hence not overwritten in Step 3, and fresh values do not occur in I_0. (iii) Two created tuples $s \neq s'$, created for K' and K'': if $K = K' = K''$ they differ on K because they were created for different key values; if $K \neq K'$ then s carries a fresh value at $w(K, K')$

while s' carries either a different fresh value there or, when $w(K, K') \in K''$, a value of $V_0 \cup \{\star\}$; symmetrically for $K \neq K''$. In every case they differ on K.

Because every declared key holds, every lookup in I is unique, so Claims 4–6 of the proof of Theorem 1 apply verbatim: the copy declarations hold for I_0 tuples, whose lookups are unchanged as no created tuple repeats a key value, and for created tuples by Step 3; $\rho_k(v) := t_k(v)$ are rows of $V_{\bar{J}}(I)$ that agree on $\Gamma \supseteq X$ and differ at T. Restrict to J by Lemma 1. □

What (S2$^+$) is doing. The two claims divide the difficulty. Filling FK attributes with $\star$ is safe because the other keys of that relation are separated by their witnesses rather than by their FK attributes, and it is the alternative repair, fresh values in FK attributes, that destroys termination. Outside (S2$^+$) the construction fails in a specific way: two completion tuples collide on a primary key, exactly as Claim K predicts. On the sharpest families that satisfy (S2) and key minimality but not (S2$^+$), namely the self-cycle $R[a_0] \subseteq R[a_1]$ with $\mathrm{pk} = (a_0)$ and $\texttt{UNIQUE}(a_1)$ and the corresponding two-cycle, an exhaustive search over all admissible instances within the enumeration bound found a finite counterexample for every one of the 42 non-implications and no gap, but the witnesses complete the injective cycle into a permutation by matching dangling FK sources to unmatched key values. That is a Hall-type argument which we do not prove, so we state the theorem under (S2$^+$) and leave the strictly larger class open.

Measurement. Over 950 random Φ_0^+ tasks across three seeds, all $9{,}060$ non-implications produced an admissible counterexample violating the dependency, with 0 failures and 0 unsoundness, judged by the unmodified verifier of Theorem 1. Soundness was checked exhaustively on 108 implied pairs with 0 violations; a further 39 implied pairs exceeded the enumeration bound and are reported as undecided rather than clean. Relative to Rosati's class of single key dependencies with foreign keys, which is known to be finitely controllable, Φ_0^+ allows several declared keys per relation and so lies outside it.

B FULL RESULTS MAP

Table 3 extends Table 4, and Table 5 positions the closure against each method family (§6).

constraint class	status of $S \models_J F \to T$	source
1 keys only (PK, `UNIQUE`); RelBench	PTIME (inherited)	Klug & Price (1982)
2 Φ_0: + FKs to PKs (cyclic, self-ref.), copies	**complete**; near-linear in $\|\bar{J}\|$ (exp. in $\|S\|$); memoised DAG poly. in $\|S\|+\|J\|$ only for bounded arity, fixed ℓ, (E3) among explicit nodes	Thm. 1, Cor. 1
3 Φ_0, finite domains (e.g. binary T)	complete if off keys, FKs and bindings	Rem. 2
4 Φ_0^+: + FKs into any declared key, (S2$^+$)	**complete**; finite = unrestricted	Thm. 5
5 Φ^+: + FK into a `UNIQUE` superkey	**undecidable**	Thm. 2

Table 4: Results map (13 rows in Appendix B); classes are SQL-declarable unless noted. Row 2 is proved here, with certificates and finite = unrestricted implication; row 5 re-encodes Fan & Libkin (2002) with one SQL construct. RelBench sits in row 1; 59.8% [58.9, 60.6] of FK-bearing SchemaPile schemas meet (S1), and 52.4% are in Φ_0 after the adapters.

	constraint class	view / domains	status	source
1	keys only (PK, `UNIQUE`)	SPJ, infinite	PTIME	Klug & Price (1982); Fan et al. (2008)
2	Φ_0	feature view, infinite	near-linear in $\|\bar{J}\|$ (can be exponential in $\|S\|$); sound and complete; finite = unrestricted; certificates	Thm. 1
3	Φ_0	small finite	sound; complete if the finite domains avoid keys, FKs and bindings (e.g. binary T), or if $\|D\| \geq 2m+1+k$ (one domain, no `UNIQUE`); otherwise completeness can fail; complexity open	Rem. 2
4	Φ_0^+: keys + FKs into any declared key, (S2$^+$)	feature view, infinite and finite	sound and complete; finite = unrestricted = closure; same construction and certificates as row 2	Thm. 5
5	Φ^+: + FK into a `UNIQUE` superkey	root only	undecidable (finite and unrestricted)	Thm. 2; Fan & Libkin (2002)
6	FDs + inclusion deps. (not declarable)	single relation	undecidable	Chandra & Vardi (1985); Mitchell (1983)
7	FDs + unary inclusion deps.	single relation	PTIME (finite $\neq$ unrestricted)	Cosmadakis et al. (1990)
8	inclusion deps. only	–	PSPACE-complete	Casanova et al. (1984)
9	FDs, SC/SPC views	finite (general)	coNP-complete	Fan et al. (2008)
10	FDs, full relational-algebra views	–	undecidable	Klug (1980)
11	Φ_0 + multi-source copies	infinite	closure sound; completeness not established	checker
12	reachability in a typed column graph	–	$O(V+E)$; unsound and incomplete for rows 1–2	App. D
13	one-to-many aggregates of free child attributes	–	not implied under the hypotheses of Prop. 2; implied under `UNIQUE`(c) by member-returning aggregates	Prop. 2

Table 3: Decidability and complexity of "$S \models_J F \to T$".

method family	reads data	target-cond.	multi-table	guarantee	abstention
probes; feature-as-target classifiers	yes	yes	no	none	–
dataset-level determinism filters	yes	yes	no	none	–
FD discovery (TANE, HyFD)	yes	no	no	exact on instance	–
static / provenance analysis of pipelines	code	no	–	code errors	–
manual curation (4DBInfer, RelBench v2)	yes	yes	yes	none	–
temporal masking (cutoffs, RTGL)	timestamps	no	yes	temporal	–
key–FK join avoidance (Hamlet)	counts	yes	yes	redundancy	–
message passing on schema graphs	no	yes	yes	colour-refinement bound	–
FD propagation; inference control	no	given FD	yes	decidable (SPC)	–
this work	**no**	**yes**	**yes**	**complete on Φ_0, maximal**	**certified**

Table 5: Positioning; citations in Section 6.

Leakage detection. Table 5 (Appendix B) positions the closure against each method family. Leakage is information about the target that is not legitimately available (Kaufman et al., 2012). Its prevalence and cost are documented (Kapoor & Narayanan, 2023; Rosenblatt et al., 2024), and training on the test task is a different issue (Dominguez-Olmedo et al., 2025). Value-based detectors read data: probes in AutoML platforms (H2O.ai, 2024; Amazon Web Services, 2024), predictive-power scores (Chorev et al., 2022), feature-as-target classifiers (International Business Machines Corp., 2023), determinism filters (Grinsztajn et al., 2022; Erickson et al., 2025) and FD discovery (Huhtala et al., 1999; Papenbrock & Naumann, 2016). Leakage also arises in feature construction (Prokhorenkova et al., 2018), and code-based detectors analyse pipelines for contamination (Yang et al., 2022; Drobnjaković et al., 2024; Grafberger et al., 2021) or prove it (Oren et al., 2024).

Inference control. Multilevel databases ask whether a user cleared for some attributes can infer a sensitive one through FDs, MVDs and other constraints (Su & Özsoyoğlu, 1991; Qian et al., 1993), and disclosure monitors track the channels such constraints open (Brodsky et al., 2000; Pappachan et al., 2024). Reading the sensitive attribute as the target gives our question, and the closure is an instance of this reasoning. What differs is the object (a target-conditioned view of many-to-one joins, identifiers and one-to-many children excluded), copies and the merge rule, certificates for every abstention, and a boundary inside SQL-declarable constraints.

Learned schema readers. Message passing is bounded by colour refinement (Xu et al., 2019; Morris et al., 2019; Barceló et al., 2020; Geerts & Reutter, 2022), a guarded counting logic on relational structures (Scheidt & Schweikardt, 2025), or refinements of it (Huang et al., 2023). Proposition 3 shows declared determination on the schema graph escapes it (readers with unique relation labels, and GNNs over the data graph, are out of scope). $\mathcal{V}$-usable information (Xu et al., 2020; Ethayarajh et al., 2022) is the dataset-level counterpart of our notion. Qualitative FDs (Chen & Darwiche, 2024) and rsbench (Bortolotti et al., 2024) also reason from declared structure.

C Adapters and fragment membership

Adapters map each corpus to Definition 1. SchemaPile provides per-column `UNIQUE` and `NULLABLE` flags, primary keys, FKs and indexes. CTU provides `information_schema` key and constraint tables. RelBench provides a primary key, a time column and single-column FKs per table. FKs into non-key columns are not lookups and are dropped. Relations without a primary key take all attributes as key (sound). Nullable FKs follow inner-join semantics and are flagged. Under inner-join semantics an entity whose nullable FK is null, or whose keyed join finds no row, has no view row, so V_J has at most (not exactly) one row per entity. On SchemaPile the adapters keep 98,516 of the 103,575 declared FKs, of which 1,311 reference a `UNIQUE` key and are kept as flagged lookups. They drop 5,059 that reference a non-key or a missing table (in 1,734 schemas) and flag 31,626 as nullable. 8,046 schemas contain a relation without a primary key. With these adapters 60.9% of SchemaPile schemas and 63.9% of CTU databases are in strict Φ_0. Outside it the closure is sound but not guaranteed complete.

D Verification of the theory and of the code

An executable checker generates random schemas with composite keys, identifying, self-referencing and cyclic FKs, copies, same-type distractors, description tables with and without uniqueness, and secondary keys. It computes the closure. It constructs the counterexample of Theorem 1 for every negative verdict, and it verifies it by brute force. On 10,000 tasks (66,589 feature–target pairs) it certified all 58,975 non-implications (5,132 leaks: 3,895 copies, 807 primary-key lookups, 430 secondary-key lookups; 2,482 identifiers). It found no soundness violation on 88,059 random instances or on 2,462,911 exhaustively enumerated admissible databases (951,298 and 1,511,613 at domain sizes 2 and 3). Ablations locate what completeness needs: dropping the merge rule loses 519 determinations, dropping copy closure 290. Per-member key edges are unsound in 14,380 cases. With an exact two-element domain the closure was sound and missed 2 of 449 determinations, as Remark 2 predicts. All 600 co-key queries were certified non-implied. Rule-based schema heuristics do not compute the relation. On the same pairs:

(a) RelLeak-Synth: 320 random schemas, 2–4 targets each

detector	P	R	F1	ms	rows
Algorithm 1 (C1 copies)	0.653	0.620	0.636	0.80	0
FK into target's table	0.333	0.190	0.242	0.013	0
single-feature probe	0.801	1.000	0.890	335	20,097
multivariate probe (GBM)	1.000	0.659	0.794	1,189	20,097
FD discovery ($\lvert\text{LHS}\rvert \leq 2$)	0.957	1.000	0.978	151	20,097

(b) RelLeak-Real: 6 real databases, 119 cells × 10 seeds

detector	P	R	F1	ms	rows
Algorithm 1	1.00	0.18	0.31	<10	0
single-feature probe (GBM)	0.71	1.00	0.83	530	all
predictive power score	0.76	1.00	0.87	8	all
FD discovery	1.00	1.00	1.00	150	all
permutation importance (GBM)	0.69	0.13	0.21	1,740	all

Table 6: Detectors against data-true labels (held-out lookup in the rows); ms: time per task (a) or per detector call (b); rows: rows read. (a) Means over the 59 tasks on which the value methods ran. FD discovery is in-sample. Agreement with the declared labels of Definition 3, which checks the implementation rather than the method, is in Table 7 (Appendix E).

- Reachability over a typed column graph reaches precision 0.158 and recall 0.826.
- A single-hop rule set reaches 0.281 and 0.713.
- A rule that flags FK columns of other tables reaches 0.037 and 0.039.

Proposition 6 (Algorithm 1 computes the closure)**.** *For every $S \in \Phi_0$, feature view J, feature L and target T, Algorithm 1 outputs the verdict of Definition 3, that is, exactly the detector D^* of Corollary 3: union–find realises (E1) and (E2), congruence closure realises (E3), and counter-based Horn closure realises the key rule, each exactly. An executable checker reports 0 disagreements on 161,687 random (L, T) pairs (Appendix D).*

Proof. Algorithm 1 keeps one term per pair (node of $\bar{J}$, declared key signature): for a node v and a declared key K of $\mathrm{rel}(v)$, the term $F_{\mathrm{rel}(v),K}$ applied to the classes $[v.k]$, $k \in K$, with every $a \in \mathrm{att}(\mathrm{rel}(v))$ a projection of that term. Under this encoding (E1) and (E2) are unions of position classes, (E3) is the congruence rule for these terms, and the key rule is a Horn rule whose premises are the classes of one key. Each rule only coarsens the partition of $\mathrm{Pos}(\bar{J})$ or enlarges the set of closed classes, so all of them are monotone on the finite lattice of partitions of $\mathrm{Pos}(\bar{J})$ ordered by refinement, paired with the subset lattice of classes. Algorithm 1 applies the rules to quiescence rather than in one pass, which matters when a relation has several declared keys: a merge on one key can newly satisfy the premise of another key's congruence or Horn rule at a different pair of nodes, and that consequence is taken in a later round. Quiescence is reached after finitely many rounds because each round that changes the state strictly climbs the finite lattice, the quiescent state is a fixpoint of all rules, and, since the iteration starts at the least element and fires every enabled rule, it is the least fixpoint, namely $\equiv$ and $\mathrm{cl}(X)$. The verdict of Definition 3 is then read off $\mathrm{cl}(\{L\})$ as in Corollary 3(i). □

E RelLeak-Synth details

Where Algorithm 1's data-true errors come from. Under C1 on RelLeak-Synth, Algorithm 1's 1,386 false positives are declarations that are true of the schema but untrue in the rows (799 stale copies, 587 noisy composite-FK pairs). Its 1,598 misses are undeclared joint determinations (799) and copies of T on one-to-many children (799), exactly the two families Corollary 3 forces any sound schema-only method to abstain on. The single-feature probe, by contrast, flags 99% of planted 45%-flipped copies of T, which no declaration implies.

Baseline coverage: 59 of 320 tasks. The value methods in Table 6a are averaged over the 59 tasks on which all of them completed within the per-task budget. FD discovery and the multivariate probe time out on the largest generated schemas. The restriction is a compute cap: Algorithm 1's own scores on those same 59 tasks are within 0.01 F1 of its scores on all 320, so the comparison is not biased toward the schema-only method.

Declared-label agreement. Table 7 scores the detectors of Table 6 against the declared labels of Definition 3. These columns measure agreement with the definition, not accuracy: for Algorithm 1 they are close to 1 by construction (Theorem 1), so they test the implementation, and the data-true labels of Table 6 test the method.

(a) RelLeak-Synth			(b) RelLeak-Real		
detector	P	R	detector	P	R
Algorithm 1 (C1 copies)	0.960[a]	1.000	Algorithm 1	1.00	1.00
FK into target's table	n/a[c]	0.000	single-feature probe (GBM)	0.13	1.00
single-feature probe	0.751	0.750	predictive power score	0.14	1.00
multivariate probe (GBM)	1.000	0.659	FD discovery	0.18	1.00
FD discovery ($\lvert\text{LHS}\rvert \leq 2$)	0.467	0.700	permutation importance (GBM)	0.00[b]	0.00[b]

Table 7: [c]This baseline emits positive flags (Table 6a) but recovers no declared leak, so precision is $0/0$ and is reported as undefined rather than as 1.000. Agreement with declared labels (Definition 3) on the tasks of Table 6: a check of the implementation, not a measure of accuracy. [a]The 167 extra flags are depth-1 shared-type chains that the C1 copy source declares and the benchmark's labels do not. [b]Permutation importance splits credit among redundant copies.

Each schema draws its table count, root-key arity, FK depth (1–5), a two-table FK cycle, a self-referencing FK and a noise level at random. Every target receives one instance of each of the 18 mechanisms, so that per-mechanism recall is estimable. The data-true label is held-out, entity-grouped, chance-normalized lookup accuracy ≥ 0.98. Candidates that are near-bijective with the root entity are excluded from data-true scoring, because a lookup cannot be evaluated on them. Value probes use fixed a-priori thresholds. FD discovery uses HyFD and TANE with $\lvert\text{LHS}\rvert \leq 2$. Table 8 gives recall per mechanism for the reference closure. Figure 2 gives the cost and the FK-deletion ablation. The benchmark's reference closure counts a type shared along a multi-hop chain as a copy. Under Definition 1 that chain needs a copy declaration per hop, which is the only disagreement with the checker's closure (799 rows). Scored as in Table 6a, the reference closure reaches data-true P 0.711, R 0.810, F1 0.757 and declared P 0.833, R 1.000, at 1.4 ms per task. Rescored with Algorithm 1 (Table 6a), C0 flags nothing because the generator declares no copies. Under C1, precision and recall against the benchmark's own declared labels are 1.000 and 0.868 on in-view rows. The multi-hop chains are flagged only at depth 1 (167 of 799), and no hard negative is flagged. Treating nullable FKs as lookups or dropping them gives identical scores. Algorithm 1 takes a median 0.94 ms per schema. Its data-true error cells are those of Section 5.4.

	mechanism	closure (ref.)	probe: single	probe: multivar.	FD discovery
leaks (recall)	multi-hop chain, shared type only	1.00	0.00	0.00	1.00
	child copy of T	1.00	1.00	0.00	1.00
	same-row copy	1.00	1.00	1.00	1.00
	copy via shared type	1.00	1.00	0.71	1.00
	description-table lookup	1.00	1.00	0.93	1.00
	stale declaration (not data-true)	1.00	0.00	0.00	0.00
	composite-FK joint pair	0.33	0.00	0.00	0.17
hard negatives (FPR)	co-key member alone	0.00	0.00	0.00	0.00
	45%-flipped copy of T	0.00	**0.99**	0.00	0.00
	FK chain broken at a partial key	0.00	0.00	0.00	**1.00**
	value-level partial cause	0.00	0.00	0.00	0.00
	inactive routing column	0.00	0.00	0.00	0.00
	joint determination, undeclared (recall)	0.00	0.00	0.00	0.33
	determination through a nullable FK	0.00	0.00	0.00	**1.00**
	independent attribute of a 1:N child	0.00	0.00	0.00	0.00
	same domain, independent values	0.00	0.00	0.00	0.00
	1:1 sibling table	0.00	0.00	0.00	0.00
	plain weak predictor	0.00	0.00	0.00	0.00

Table 8: RelLeak-Synth per mechanism (320 schemas): recall for leaks, false-positive rate for hard negatives. The composite-FK joint pair is a determination only at noise 0. The undeclared joint determination is scored as recall.

RelLeak-Real. Known columns (20 kinds: exact, recoded, noisy and partial copies of the target; lookups at hops 0–2; declared and undeclared twins; non-leaking controls) are planted at recorded random positions. These positions span CTU financial, Hockey and Credit, and RelBench rel-event, rel-trial and rel-avito. Declared ground truth marks the columns whose copy or key the planted

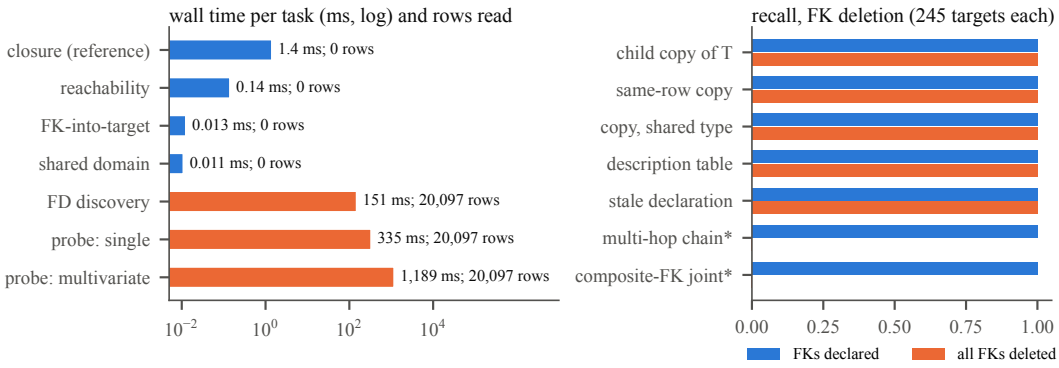


Figure 2: RelLeak-Synth. Left: mean wall time per task and rows read (schema-only methods in blue read 0 rows). Right: recall with all declared FKs deleted. The two FK-dependent mechanisms (*) drop to 0, the others are unchanged.

schema declares, data-true ground truth is measured on the rows. Schema-only verdicts are constant across the 10 seeds in all 119 cells. The value detectors used 872 s of compute for the whole run and Algorithm 1 less than 0.01 s per task. Two RelBench tasks are collapsed to first-snapshot labels, and rel-event and rel-avito are row samples. Lookup hosts at hops 1–2 are synthetic one-to-one tables on real roots except in financial. In the composite-FK construction each member alone is independent of the target: marginal value baselines never flag it (single-feature GBM chance-normalized accuracy ≤ 0.0007, PPS 0, no FD), while a joint model's permutation importance gives each member 0.563. Table 9 shows that a planted leak changes a model only when it is present at serving time.

Dataset	Kind	clean	leak (train+test)	leak (train only)
financial	exact copy	0.642	1.000	0.500
financial	noisy copy p90	0.642	0.900	0.601
financial	partial coverage50	0.642	0.919	0.623
hockey	exact copy	0.506	1.000	0.500
hockey	noisy copy p90	0.506	0.898	0.498
hockey	partial coverage50	0.506	0.874	0.497
reltrial	exact copy	0.589	1.000	0.500
reltrial	noisy copy p90	0.589	0.912	0.561
reltrial	partial coverage50	0.589	0.890	0.571

Table 9: RelLeak-Real: test AUC of a gradient-boosted model (10 seeds) without the planted column, with it at training and test time, and with it at training time only (imputed at test time).

F SchemaPile details

copy source	C0	C1	C2
SchemaPile: targets flagged (of 1,137,685)	124	40,524	8,249
FD holds in shipped rows: leaks vs. control (%)	–	4.5 vs. 5.2	22.2 vs. 9.5
testable pairs: leaks / control	–	1,201 / 1,054	144 / 179
CTU: columns flagged (of 12,856)	0	438	101
documented targets flagged (of 100)	0	2	1
designated targets abstained (of 4)	4	4	4
RelBench: official tasks flagged (of 23)	0	0	–

Table 10: Real corpora under Algorithm 1, by copy source (0 rows read). Data check: pairs with keys unique in the rows and non-trivial (a repeated feature value, at least two target values), against matched control columns over the same join. Counts are unchanged at view depth 5, 400 nodes and copy depth 8. "Documented targets" are the prediction targets the CTU repository documents

for 100 databases; "designated targets" are the four most commonly benchmarked CTU targets (financial `loan.status`, mutagenesis `molecule.mutagenic`, Hepatitis `dispat.Type`, imdb `actors.gender`).

SchemaPile-Perm v0.2 (md5-verified) has 22,989 schemas, 198,756 tables, 1,330,024 columns and 103,575 declared FKs. 10,763 of 22,559 schemas with targets declare no FK. Shipped sample rows exist for 218,946 columns in 7,098 schemas (median 4 rows). Each target is rooted at its own table (N:1 view, depth 3, at most 200 nodes, masked target). On samples of real schemas in Φ_0, each of 22,888 abstentions has a finite counterexample that passes a brute-force check, and 2,046 leaks held on all 7,329 random admissible instances tried. Under C0 all 124 flagged targets re-read the target through an identifying key cycle, and 12 of them also have a leaker by join equality. Under explicit copy assumptions, Algorithm 1 flags 40,524 targets under C1 (same name and type along an FK; unflagged 96.44% [96.22, 96.65]). Of these, 39,885 have a copy, 35,355 a primary-key lookup and 9,276 a `UNIQUE` lookup. It flags 8,249 targets under C2 (shared referenced key; unflagged 99.28% [99.20, 99.35]). By mechanism (non-trivial pairs, keys unique in the rows, FD holds, 95% CI over schemas), the rates are:

- C1 copies hold for 3.9% [1.9, 6.2] of 465 pairs, against 2.1% [0.9, 3.7] of 427 controls.
- C1 primary-key lookups hold for 4.8% [2.7, 7.4] of 726, against 7.1% [4.8, 9.7] of 620.
- C2 copies hold for 20.3% [8.4, 44.5] of 74, against 7.8% [3.2, 17.0] of 90.
- C2 primary-key lookups hold for 23.5% [10.1, 46.5] of 68, against 11.4% [5.3, 21.4] of 88.

Verdict counts are identical at view depth 5, 400 nodes and copy depth 8, and under detector versions 1.1 and 1.2.

Where the corpus counts apply. Of the 22,989 schemas and 1,137,685 targets, under C0:

condition on the schema	schemas	targets	% of targets
in Φ_0	14,003	381,560	33.5
at least one FK declared	11,809	757,382	66.6
at least one FK kept	11,201	717,208	63.0
ships a data sample	7,098	493,863	43.4
in Φ_0 and at least one FK kept	6,188	234,972	20.7
in Φ_0, at least one FK kept, ships data	1,537	81,836	7.2

Inside Φ_0 the closure is complete, so each of the 381,560 targets there receives either a certificate or a forced abstention with a counterexample; outside Φ_0 an abstention is sound but not proved necessary. A schema with no FK kept (11,788 schemas, 420,477 targets) cannot express a cross-table determination, and a schema shipping no rows is not a task a value-based method can screen, so the 81,836 targets meeting all three conditions are the narrowest population on which the closure and a value-based method are both applicable. 106 of the 124 flagged targets are in schemas inside Φ_0, all 124 are in FK-bearing schemas and 20 are in schemas that ship rows; of the 115,520 IDENTIFIER targets, 28,961 are in schemas inside Φ_0.

Declaration ablation. For each of 5 seeds, every declared FK and `UNIQUE` of every schema draws one uniform number and is deleted when that number is below p, so deletions are nested in $p \in \{0, 0.1, \ldots, 1\}$. Primary keys are kept, and each pass uses the view and copy settings of Table 10. At $p = 0$ the pass reproduces Table 10 (124 C0 and 40,524 C1 LEAK targets). The number of determined (feature, target) pairs, i.e. LEAK plus IDENTIFIER, falls monotonically in every seed, from 421,842 (C0) and 1,458,135 (C1) to 0 (Figure 3), as Corollary 4 requires. Under C1 the LEAK and JOINT target counts also fall monotonically (40,524 and 31,686 at $p = 0$; every JOINT target also has a single-feature LEAK). Under C0 no target has a JOINT leak at any p. The C0 LEAK count rose by one target in two steps (seed 0, $p = 0.1 \to 0.2$: 111 to 112; seed 1, $p = 0.6 \to 0.7$: 48 to 49). Both increases occurred in one schema whose IDENTIFIER pairs fell at the same step. A deleted key can turn an IDENTIFIER into a non-identifier that still determines the target, so the LEAK verdict, unlike the determination, need not be monotone.

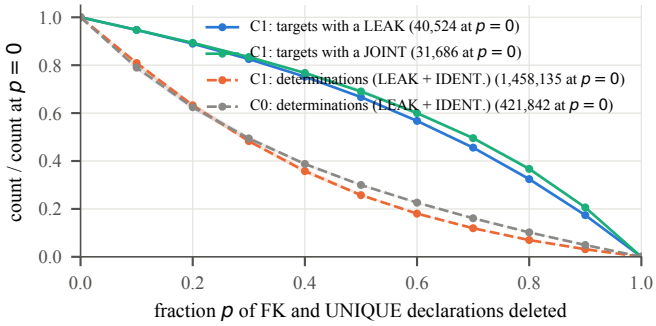


Figure 3: Declaration ablation on SchemaPile: counts relative to $p = 0$ after deleting a random fraction p of FK and `UNIQUE` declarations; mean over 5 seeds, band = range over seeds.

G RELBENCH DETAILS

RelBench v2 registers exactly 23 autocomplete tasks over 9 datasets (amazon 1, avito 2, event 3, f1 2, hm 1, ratebeer 1, salt 8, stack 1, trial 4). Its manifest declares single-column primary keys (71 of 83 tables) and 95 single-column FKs. Composite keys cannot be expressed. Value audit: for each task, the best visible single column, pair and greedy triple, scored by held-out lookup under RelBench's temporal split and under a random split. Random splits inflate scores through entity memorization (e.g., 0.633 vs 0.018 for qualifying-position). On rel-salt, (plant, shipping condition) $\rightarrow$ shipping point is 0.989 at item grain ($n = 2{,}319{,}540$; 0.989 when grouped by document; 0.961 temporal). Two other header dependencies are 0.001 and 0.059. Restoring rel-salt's native composite item key makes a co-key rule flag the line number as a determinant of 12 header attributes, and the rows refute all 12 (normalized ≤ 0.038). All RelBench verdicts are from Algorithm 1 (median 0.11 ms per task, 0 rows): the 23 official tasks abstain under C0 and C1, and with rel-salt's composite item key restored it flags none of them. The injection sweep adds, per task, a value-verified check table keyed by the task's best visible determinant and an ownership declaration of the target. Composite determinants yield JOINT, or LEAK when one column determines them all. The value baselines of Table 1 run on unpruned views with the removed columns re-inserted and 5-fold entity-grouped splits. Their wall-clock sums over the 23 tasks are 29.5 s for PPS, 121–152 s for HyFD, 198–417 s for TANE and 636–2,015 s for Pyro, and about 1,750 s for the single-feature probe on an uncontended machine. HyFD and Pyro failed on the widest views (2 and 4 runs), and Pyro cannot run above 64 columns. Approximate FD discovery (Pyro, TANE) finds 34–39 removals at precision 0.15–0.19. Most of its false flags fall on header fields that are legitimately known at prediction time. All value methods first scan 93.2M rows to build the 23 views. The model-impact runs use scikit-learn's histogram gradient boosting ($\ell_2 = 1$) on the one-hop feature view, all training classes, and every test row. The rel-salt fits use a 250k-row training subsample (on all 1.62M rows item-incoterms gives $\Delta = +27.8$ instead of $+28.1$). The published RelBench GNN reaches 98.4% on item-shippoint without the removed columns, so a re-inserted lookup has little room there. In rel-trial, a column often read as determining enrollment is a join handle and a temporal leak: only 0.21% of its rows exist at the task's seed time, and none in the test period.

The 90 removals (Table 1). By reason, 62 are other tasks' labels, 19 same-row attributes and 9 free-text columns. 15 of the 62 labels live on one-to-many children and are not view positions. A removal determines T in the rows if its held-out lookup score under the temporal split is ≥ 0.95 or an exact FD holds on the 100k-row sample (HyFD). Seven do: positionOrder (1.000), header Incoterms for item-incoterms (0.998), shipping point for item-plant (0.970), badge name for badge class (0.991) and three age bounds for the adult and child labels (exact FDs). The two copies are not exact: position = positionOrder on 99.91% of non-null rows and item and header Incoterms agree on 99.93% of items. The same-row equality that certifies positionOrder is, moreover, outside Φ_0, whose copies run along FKs. With the two copy declarations Algorithm 1 flags positionOrder and header Incoterms, plus the item's FK to its header and the header key, which determine T wherever the copy holds. A preliminary structural detector that scores declared copies directly reports 3 of 90 at precision 1.0 for the same declarations. It also counts the items' Incoterms for sales-incoterms, a

reverse copy through a one-to-many child that is not a view position (Proposition 2), and it raised no flag on the item's key path. The injection sweep fires on exactly the injected task: gated, 1 injected and 1 fired; ungated, 23 injected and 22 fired (12 LEAK, 10 JOINT), with no flag outside the injected targets.

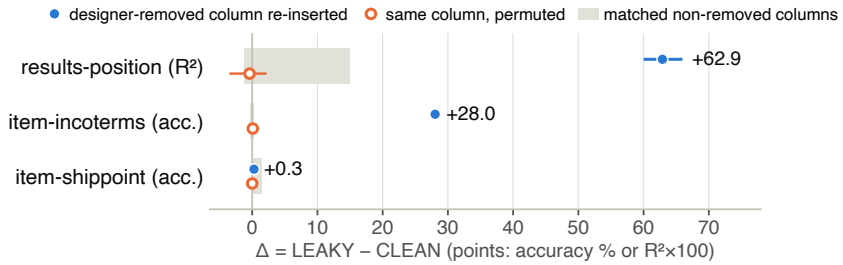


Figure 4: Re-inserting RelBench's hand-removed columns (Table 2): Δ with 95% CI for the three named removals (blue), the same column permuted (orange), and the range for matched non-removed columns of the same table (grey band). 5 seeds.

Blind declarations and a language-model detector. Each run was a fresh headless language-model session with no tools, no saved notes or project settings and the bundle as its only input. Three runs used each bundle (two Opus, one Sonnet). The *declaration* bundle holds the nine RelBench schemas (tables, columns, types, keys, foreign keys, up to three example values per column; no tasks, targets, removal lists or results). The *detector* bundle adds the 23 task cards (entity, target, time column). Renamed bundles replace table and column names and categorical values by consistent tokens and shift dates. With the blind declarations, Algorithm 1 abstains on all 23 tasks: 0 of 90 removals, 0 of 7 determining columns, no other flags. The declarations are mostly true: copies hold on $\geq$ 99% of joined rows for 14/19 named and 13/14 renamed, and `UNIQUE`s have no duplicates for 27/40 and 16/27. But almost every true copy repeats a parent's date, and none has a target as its child. Asked directly, the model does better than Algorithm 1 on these schemas (Table 11): the majority of three runs finds 24 removals at precision 0.80, 16 of them beyond the other tasks' targets that the task list itself reveals. It also flags none of the header fields value baselines over-flag. It reads no rows but needs knowledge: 10 of the 12 runs named RelBench in a recognition probe, and RelBench's removal lists are public, so its hits cannot be separated from recall. Tokenized names cut the determining columns found from 6 to 2. It is a knowledge-dependent baseline beside the schema-only Algorithm 1, which certifies only what is declared.

method (RelBench, 23 tasks)	reads rows	needs knowl.	labels (62)	same-row (19)	text (9)	all (90)	det. (7)	other flags	header (4)
Algorithm 1 as released	no	no	0	0	0	0	0	0	0
Algorithm 1 + 2 author declarations	no	hindsight	1	1	0	2	2	2	0
Algorithm 1 + blind declarations, named	no	no[a]	0	0	0	0	0	0	0
Algorithm 1 + blind declarations, renamed	no	no[a]	0	0	0	0	0	0	0
task-list rule (other tasks' targets)	no	no	60	0	0	60	2	18	0
LM detector, named	no	yes	8	14	2	24	6	6	0
LM detector, renamed	no	yes	4	12	2	18	2	2	0
PPS $\geq$ 0.5 (100k rows)	yes	no	3	7	0	10	7	5	–

Table 11: RelBench's 90 hand-removed columns found by each method. Blind declarations and LM detector: union (declarations) or majority of three isolated runs at confidence $\geq$ 0.5 (detector). header: the 4 header fields value baselines flag although they are known at prediction time. [a]The declarers may recall the datasets from pre-training.

Impact with RelBench's GNN. To check that the impact of re-inserted copies is not specific to gradient boosting, we ran RelBench's heterogeneous GraphSAGE baseline with temporal neighbour sampling on the two copy tasks. This reused the CLEAN/LEAKY construction, temporal split and training rows of the gradient-boosting runs, and scored every official test row. rel-f1 uses the example defaults (5 seeds); rel-salt uses a reduced sampling configuration (batch 256, neighbours $[64, 32]$, 3 epochs; 3 seeds), identical across conditions, because the default exceeded our 10 GB

memory cap. The best validation epoch was often the last, so the GNN gains are likely lower bounds. Shuffled-column placebos are null on both tasks. Adding a column of pure noise raises the GNN's clean rel-f1 score by $+6.0$ [$+3.9$, $+7.8$] points, an effect of the extra input rather than of target information. Holding the number of inputs fixed (LEAKY minus shuffled) gives $+60.9$ and $+30.8$ for the GNN, and $+63.3$ and $+27.9$ for gradient boosting (Table 12, Figure 5).

Task (metric)	Model	Seeds	CLEAN	LEAKY	Δ [95% CI]	Permuted Δ [95% CI]
item-incoterms (acc. %)	GBM	5	71.8	99.9	$+28.0$ [$+27.8$, $+28.3$]	$+0.1$ [-0.1, $+0.2$]
item-incoterms (acc. %)	GNN	3	67.8	99.9	$+32.1$ [$+30.6$, $+33.2$]	$+1.3$ [-0.1, $+3.7$]
results-position ($R^2\times100$)	GBM	5	36.7	99.6	$+62.9$ [$+60.0$, $+66.0$]	-0.4 [-3.5, $+2.2$]
results-position ($R^2\times100$)	GNN	5	34.2	95.8	$+61.6$ [$+58.0$, $+65.2$]	$+0.7$ [-3.8, $+5.3$]

Table 12: Impact of re-inserting the certified copies, by model family (95% CIs over seeds and test rows).

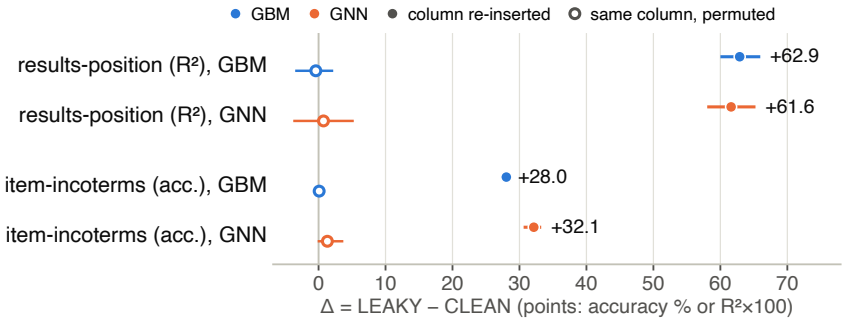


Figure 5: Δ (LEAKY−CLEAN) with 95% CIs per model family and task; filled: re-inserted copy, hollow: shuffled column.

H A FEATURE PIPELINE, AND SCREENING WITHOUT ROWS

Deep Feature Synthesis. We replace the hand-built one-hop views of Table 2 by featuretools' Deep Feature Synthesis (DFS, version 1.31.0, default primitives, depth 2) (Alteryx, 2024), which stacks aggregation and transformation primitives along the declared relationships, on the two copy tasks. RelBench's official temporal split supplies the cutoff times (bucketed to the day). rel-f1 results-position uses the full task table; rel-salt item-incoterms uses 80,000 training and 32,000 test rows drawn from the official split, with parent tables restricted to the rows they join. The model is histogram gradient boosting with the hyperparameters of Table 2, 5 seeds on 90% subsamples, and 95% paired bootstrap intervals ($B = 2{,}000$). Three conditions differ only in their input tables: CLEAN (as released), LEAKY (the one designer-removed column re-inserted before synthesis) and DETECTOR_CLEANED (LEAKY without the columns Algorithm 1 flags, reading no row, under the declarations of Table 2(b)).

Results. DFS multiplies the leak (Table 13, Figure 6): on results-position its primitives turn positionOrder into 16 features (the column and its mean, maximum, minimum, standard deviation and skew by race, driver and constructor). R^2 rises from 0.149 to 0.997. On item-incoterms the header Incoterms enters as one feature and accuracy rises from 72.0% to 99.9%. Before synthesis, Algorithm 1 flags positionOrder on rel-f1 (a certificate outside Φ_0, as in Table 2). On rel-salt it flags the header Incoterms plus the item's FK to its header and the header key, which DFS follows as a relationship rather than using as features. Dropping the flagged columns and re-running DFS reproduces the CLEAN feature set and score in both tasks. DFS's clean view matches the hand-built one on item-incoterms (72.0% vs. 71.8%) and is weaker on results-position (R^2 0.149 vs. 0.367).

Screening without rows. Algorithm 1 (v1.2; copy sources C0, C1 and C2) screens the whole SchemaPile catalogue, 22,989 schemas and 1,137,685 candidate targets, in 117.5 s of wall-clock time (445.2 CPU-seconds on 4 workers) without reading a row. A repeat of the same sweep took

task (metric)	condition	#feat.	#feat. from leak	score	95% CI	detector verdict
results-position (rel-f1)	CLEAN	84	0	0.149	[0.047, 0.218]	–
	LEAKY	100	16	0.997	[0.996, 0.998]	LEAK
	DETECTOR_CLEANED	84	0	0.149	[0.047, 0.218]	–
item-incoterms (rel-salt)	CLEAN	81	0	72.0	[71.2, 72.6]	–
	LEAKY	82	1	99.9	[99.9, 99.9]	LEAK
	DETECTOR_CLEANED	81	0	72.0	[71.2, 72.6]	–

Table 13: Deep Feature Synthesis on the two copy tasks (score: R^2 for results-position, accuracy in % for item-incoterms; 95% paired bootstrap CI). #feat. from leak: features DFS derives from the re-inserted column. Verdict of Algorithm 1 on the re-inserted column under the declarations of Table 2(b).

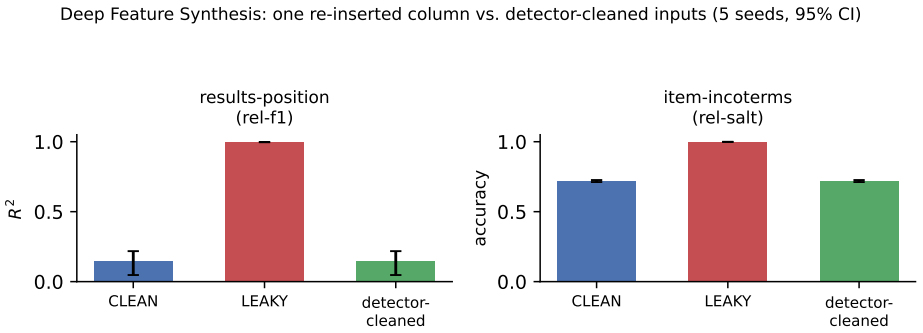


Figure 6: Deep Feature Synthesis under CLEAN, LEAKY and DETECTOR_CLEANED inputs on results-position and item-incoterms (95% CI).

106.8 s (403.6 CPU-seconds), the run quoted in §4; wall-clock varies by about 10% between runs on the same machine. Only 7,098 of the 22,989 schemas ship any data sample. For the other 15,891 no value probe or FD discovery can run without first obtaining the data from its owners, so the comparison is between screening and not screening, not between two speeds. For the 7,098 with samples, scaling the measured per-unit costs of the value methods on CTU and RelBench gives about 46 days for a single-feature probe. FD discovery would take 114 to 238 days on the whole catalogue (projections, not runs). The screen is sound for what is declared, and every ABSTAIN carries a counterexample database (§5.2). It does not see undeclared determinations such as the one-to-many sales-incoterms leak (Table 2).

I WikiDBs, GitTables and small FK corpora

Columns carry a Wikidata property in 96.2% of cases. The WikiDBs sample is the first 5,094 databases of the first archive split in archive order, which is independent of database index (Spearman $\rho = -0.0021$). Data checks cover entity-valued columns. Join-graph reachability leaves 2.12% [2.00, 2.24] of non-key columns unflagged and a property-bound copy 9.36% [9.08, 9.67]. The ownership list is learned on half the databases in each of 20 cross-fit folds (threshold 0.5, about 39 owned properties) and applied schema-only to the other half. GitTables declares no keys. Copies keyed on Schema.org types leave 91.8% of columns unflagged at macro precision 51.5% and recall 14.6%. On Chinook and Northwind, all 12 real cross-table copies are renamed (billing and shipping columns). A same-name gate recovers 1 and a Schema.org gate 0.

J CTU details

Independent schemas and hosting duplicates. The CTU server lists 147 schemas, of which two are cached MySQL system catalogs, giving 145 content databases. *119 declare at least one foreign*

key and a primary key (12,856 non-key candidate targets). Of the 26 that declare none, 15 are single-table, where a foreign key is trivially absent, and 11 are multi-table, 8.5% of the 130 multi-table databases. The repository also hosts the classic PKDD'99 Financial dataset four times, at four visibility levels:

- `financial` (8 tables, 8 declared FKs).
- `Financial_ijs` (9 tables, 8 FKs).
- `Financial_std` (7 tables, 6 FKs).
- `ctu_financial` (8 tables, 0 declared keys or FKs, with the same table set and the same `account`-table columns as `financial`).

A closure sweep counts all four as independent rows, so the "declares nothing" bucket contains at least one database, `ctu_financial`, whose silence is a hosting artefact (every key and foreign key stripped from a copy of `financial`) rather than an absence of structure. Independent real-world schemas with no declared structure therefore number at most 25 of 144, pending a full duplicate sweep (Appendix Q gives the four-way table and the asymmetry with the ERP case). The per-database experiments below and in §5.2 reference the database by the name `financial`, not the stripped copy `ctu_financial`, and are unaffected.

On Basketball_men a coach's win percentage per team-year tracks team rank (Spearman -0.84, $N = 1{,}433$), an aggregate over one-to-many children that no declared key or FK implies. A rule that treats any FK column of another table as a determinant flags 4,497 targets. Every one of its 14,666 raw flags is a join handle, and with join handles excluded it flags none. Algorithm 1's verdicts read 0 data rows and take a median 0.71 s per database, almost all of it the metadata load from the remote server (0.07 ms for the closure). The relational GNN has one node type per table and one edge type per declared FK and its reverse, two heterogeneous SAGE layers, early stopping on a train-internal split, and 5 seeds. Removing the leaking tables' node types gives drops of 0.559 (Accidents, macro-F1), 0.269 (FNHK, R^2), 0.239 (northwind, R^2), 0.070 (PremierLeague) and 0.027 (financial). Final model-impact runs (gradient boosting, 5 seeds, paired 95% CIs over seeds) cover 28 databases. Under C0 and C1 Algorithm 1's leaker set is empty in all of them, so CLEAN equals LEAKY ($\Delta = 0$). C2 flags only Credit. Removing the tables that the FK-column rule reaches through join handles leaves gaps, all through one-to-many children:

- Accidents: $+0.660$ [$+0.648$, $+0.672$].
- PremierLeague: $+0.554$ [$+0.522$, $+0.586$].
- FNHK: $+0.523$ [$+0.455$, $+0.590$].
- northwind: $+0.392$ [$+0.176$, $+0.609$].
- stats: $+0.155$ [$+0.029$, $+0.282$].

Removing leakers named by hand gives $+0.811$ (SFScores), $+0.449$ (financial), $+0.356$ (Seznam) and $+0.208$ (ErgastF1), which no declared key or FK reaches. A block-level placebo (leave-one-block-out Δ of every other block, 20 databases) exceeds 0.05 in two databases, and the join-handle removal exceeds the placebo maximum in 7 of the 20. Under entity-grouped splits, 10 of the 11 databases with $\Delta \geq 0.05$ keep $\Delta \geq 0.05$. Against block-level truth (leave-one-block-out $\Delta \geq 0.05$; 28 databases, 252 blocks), value detectors reach the following database-level F1 scores, against 0.55 for the join-handle rule:

- 0.24 (PPS ≥ 0.8).
- 0.17 (single-feature probe ≥ 0.95).
- 0.44–0.53 (FD discovery, $|\text{LHS}| \leq 2$).
- 0.69 (entity-grouped PPS ≥ 0.3).
- 0.75 ($|\rho| > 0.3$).

They first assemble the view (median 165 s, up to 903 s per database) and read a median 10,000, mean 101,027 and up to 1,625,554 rows. Model-side compute after assembly is a median 9.6 s for the single-feature probe and 264 s for FD discovery, whose runs are capped at 300 s.

Algorithm 1 Closure detector for declared leakage (sound and complete on Φ_0; reads no data row)

Require: schema S, feature view J rooted at E, target T, features F, copy declarations (a switchable set)
1: $\bar{J} \leftarrow$ copy-close(J); union–find on $\mathrm{Pos}(\bar{J})$ for (E1), (E2); congruence closure for (E3)
2: index each declared key (primary or `UNIQUE`) by its premise classes, with one counter per key
3: **for all** $L \in F$ **do**
4: $\quad G \leftarrow \mathrm{cl}(\{L\})$, keeping a witness pointer for every key firing
5: $\quad$ **if** every root-key class is in G **then** emit IDENTIFIER (L)
6: $\quad$ **else if** $[T] \in G$ **then** emit LEAK (L) as equality ($[T] = [L]$), constant, or lookup(w, K, depth)
7: **end for**
8: **if** $[T] \in \mathrm{cl}(F)$ and F does not fix the root key **then**
9: $\quad$ emit JOINT (F) with essential members $\{L \in F : [T] \notin \mathrm{cl}(F \setminus \{L\})\}$
10: **end if**
11: emit ABSTAIN for all other features, each with the counterexample of Theorem 1 as certificate

K THE CLOSURE DETECTOR, IN FULL

Algorithm 1 is the detector of §4, stated in full. It reads no data row.

L COMPUTING THE CLOSURE WITHOUT THE COPY-CLOSED VIEW

This is the machinery behind Corollary 1, still a *corollary* in our terms[2] despite the machinery this appendix takes to state it precisely.

Throughout $S \in \Phi_0$, J is given explicitly, h is the height of the relation-level copy graph, β the largest number of distinct foreign keys along which one relation declares copies, and $w = \max_R |\mathrm{att}(R)|$. A position of $\bar{J}$ is written (u, π, a): u a node of J, π a path of copy edges below it, $a \in \mathrm{att}(\mathrm{rel}(u, \pi))$.

Definition 4 (The step map)**.** For a position $p = (u, \pi, a)$, $\mathrm{step}(p)$ is defined by exactly one of two cases. **(down-copy)** If a is the target of a copy declaration (f, a, b) of $\mathrm{rel}(p)$, then $\mathrm{step}(p) = (u, \pi f, b)$. **(up-key)** If a is the i-th attribute of the key bound by the parent edge of p's node, then $\mathrm{step}(p)$ is the i-th binding attribute at the parent (or the bound constant). step is undefined otherwise. $\mathrm{nf}(p)$ is its normal form.

Lemma 3 (Normal form)**.** step *is a well-defined partial function, terminates in at most* $h + \mathrm{depth}(\bar{J})$ *steps, and for all positions* p, q *of* $\bar{J}$*:* $p \equiv_0 q$ *iff* $\mathrm{nf}(p) = \mathrm{nf}(q)$*, where* $\equiv_0$ *is the equivalence generated by* (E1) *and* (E2) *alone. Consequently class identity is decided in* $O(h + \mathrm{depth})$ *per position, although* $\equiv_0$ *may have exponentially many classes.*

Proof. Well defined. By (S3) a copy target is descriptive, hence in no primary key, so the down-copy and up-key clauses are mutually exclusive. By (S3) a copy target carries at most one copy declaration, so down-copy is deterministic. Up-key is deterministic because an edge binds each key attribute once. *Termination.* A down-copy step descends one edge of the relation-level copy graph, which is acyclic of height h by (S3), so at most h down-copy steps occur consecutively. After a down-copy from (u, π, a) to $(u, \pi f, b)$, either b is again a copy target, or up-key applies and returns (u, π, x) with x an attribute of the foreign key f. By (S3) an FK attribute is never a copy target, so no down-copy applies at x, and every subsequent step is an up-key, each strictly decreasing depth. Hence step terminates. *Characterisation.* (E1) is exactly up-key read as an equality and (E2) exactly down-copy, so $p \equiv_0 \mathrm{step}(p)$ and therefore $\mathrm{nf}(p) = \mathrm{nf}(q)$ implies $p \equiv_0 q$. Conversely "having the same normal form" is an equivalence relation that contains every generator of $\equiv_0$, hence contains $\equiv_0$. □

Lemma 4 (Saturation is downward closed)**.** *If the key rule fires at a node* v *of* $\bar{J}$ *then it fires at every node of the copy subtree rooted at* v*, and every position of that subtree lies in* $\mathrm{cl}(X)$.

[2]Throughout the paper, a result is labelled a corollary when its proof proceeds by direct appeal to Theorem 1's closure characterization, whatever that appeal takes to state precisely (as here). It is labelled a theorem when it establishes a decision procedure or a decidability boundary in its own right.

Proof. Firing at v puts every class $[v.a]$, $a \in \mathrm{att}(\mathrm{rel}(v))$, into $\mathrm{cl}(X)$, in particular every foreign-key attribute. For a copy child v_f, (E1) gives $[v_f.k] = [v.x_k^f] \in \mathrm{cl}(X)$ for every $k \in \mathrm{pk}(\mathrm{rel}(v_f))$, so the key rule fires at v_f with $K = \mathrm{pk}$. Induction over the subtree, which is finite by (S3). □

Proposition 7 (Shape of the (E1)–(E2) closure)**.** *Let* $\mathrm{cl}_0(X)$ *be the closure taken over the* $\equiv_0$*-classes of Lemma 3, that is, with the merge rule* (E3) *omitted. Then* $\mathrm{cl}_0(X) = \{\mathrm{nf}(x) : x \in X\} \cup \{$*constant classes*$\} \cup \{\mathrm{nf}(p) : p$ *a position of a node in a saturated subtree*$\}$*, and membership of a single anchor is decided by a prefix test along its copy path, never by enumeration. The identity is stated for* cl_0 *because* nf *decides* $\equiv_0$ *only: under an* (E3) *merge a single* $\equiv$*-class has several normal forms, so the right-hand side does not enumerate the classes of* cl*.* (E3) *is layered on top separately, by the algorithm below, which applies it among the nodes kept explicit.*

The algorithm. Keep the first ℓ copy levels explicit (a view of $\|J\| \sum_{i \le \ell} \beta^i$ nodes, on which (E1)–(E3) and the key rule are run by Algorithm 1 unchanged). Below that frontier, replace the subtree by the memoised fixpoint $\mathrm{solve}(R, A)$: the least $A' \supseteq A$ closed under (i) the key rule at R, and (ii) for each copy foreign key f of R, $a \in A'$ whenever $b \in \mathrm{solve}(\mathrm{rel}(f), \{k_i : x_i^f \in A'\} \cup \{b' : (f, a', b') \in \mathrm{Cp}, a' \in A'\})$ for the copy (f, a, b). Memoising on (R, A) shares every identical subtree. The copy graph is acyclic, so the recursion terminates. A target whose anchor lies below the frontier is answered by Lemma 4 (does some node on its copy path fire?), never by expanding the subtree.

Proposition 8 (Cost)**.** *The memo has at most* $\|S\|2^w$ *entries. With the explicit layer the total is* $O\big((\|J\|\beta^\ell + \|S\|2^w) \cdot \mathrm{poly}(\|S\|)\big)$*, independent of* $\|\bar{J}\|$*. For bounded arity and fixed* ℓ *this is polynomial in* $\|S\| + \|J\|$*.*

Exactness and what is residual. Lemmas 3 and 4 are exact. The one approximation is (E3): merges are applied among the nodes kept explicit. On the blow-up family B_k of Theorem 1 ($\|\bar{J}\| = 2^{k+1} - 1$), the memo has 2 entries at every k. The verdict is computed in 1.1 ms at $k = 200$, where $\bar{J}$ would have $3.2 \cdot 10^{60}$ nodes. The reference needs 0.56 s already at $k = 16$. Against the unchanged detector, $\ell = 2$ reproduced every verdict on 40,786 random and adversarial pairs (and on 8,001 SchemaPile pairs), while $\ell = 0$ disagreed on 0.10%–2.8% of them. Each disagreement is an (E3) merge involving a copy-closure node. Whether the 2^w factor is necessary is open. We note only that the whole-key rule conjoins classes that all sit at *one* node and flow *downward* from its parent. This follows because (S2) and (S3) forbid a declared key made of copy targets, so no conjunction flows upward. That is why the closure is a *stack-synchronous* pushdown saturation rather than an alternating one, escaping the DExpTime of alternating pushdown reachability (Bouajjani et al., 1997).

Why the blow-up is rare in practice. On SchemaPile-Perm under the same-name-and-type copy source, $\beta \ge 2$ (the only source of the exponential) holds for 8.4% of relations (1,010 of 12,054; $\max \beta = 12$). But the copy graph is shallow, so $\|\bar{J}\|$ never exceeded 18 nodes on 941 sampled tasks. The exponential is a worst-case phenomenon that the corpora do not exhibit. Corollary 1 removes it from the bound regardless.

Remark 4 (FKs into secondary keys: the completion terminates)**.** Under (S2$^+$) the completion in Step 2 of the proof of Theorem 1 needs no repair: it terminates and leaves every declared key distinct (Theorem 5, Appendix A.1). The measurement agrees. Over 950 randomly generated Φ_0^+ tasks, every one of the 9,060 non-implications yielded a finite admissible instance violating the dependency, with no failure and no unsoundness. A separate three-seed comparison isolates what the choice of filler does: across 1,393 non-implications the completion of Theorem 1 succeeded on all of them, while substituting a globally fresh value for $\star$ at FK source attributes succeeded on 38, the remaining 1,355 exhausting the tuple budget without closing. A fresh value at an FK source position demands a further tuple whose own FK source is again fresh, which is what walks the infinite 1:1 chain. The chain is a property of that substitution, not of the fragment.

M Sample-verified ownership: proof and details

Setup. Fix a task's root table E and K candidate copy declarations $d_1, \ldots, d_K = (f, a, b)$ (an FK f, child attribute a, parent attribute b; or a same-row pair with no FK). Root rows are sampled

uniformly, with replacement. A root row is *covered* by d_i if every FK hop on d_i's declared lookup path resolves to a non-null row (a dangling hop is scored as uncovered, not violated, matching Definition 1's admissibility and SQL null semantics). A covered row is *violated* if the two dereferenced values disagree. A parent row referenced by many root rows is tested once per referencing root row, so it carries weight proportional to its fan-in, matching the unit the downstream task is scored on.

Per-row argument. If a certified LEAK's derivation (Corollary 2) uses declarations $d_1, \ldots, d_m$, then on every root row where the (trusted) declared keys/FKs hold and every d_i is covered and unviolated, the derivation's equality chain holds literally, so $T(r) = g(F(r))$. Hence the dependency holds on every covered root row except those violating some d_i; a row some d_i leaves uncovered satisfies neither the premise nor its negation, so it lies outside the guarantee. If each has true violation rate $\leq \varepsilon$, a union bound over the m components gives $\geq 1 - m\varepsilon$ of the covered rows.

Proof of Theorem 3. For one candidate with true violation rate ρ_i, $P(0 \text{ violations in } n \text{ draws}) = (1-\rho_i)^n$. If $\rho_i > \varepsilon$ this is $< (1-\varepsilon)^n \leq e^{-n\varepsilon}$. Union bound over the (at most) K candidates with $\rho_i > \varepsilon$: $P(\exists \text{ such an accepted candidate}) \leq Ke^{-n\varepsilon}$, which is $\leq \delta$ at $n = \lceil \ln(K/\delta)/\varepsilon \rceil$. The exact (non-relaxed) bound $K(1-\varepsilon)^n \leq \delta$ is used directly wherever tighter. Combined with the per-row argument, every certified LEAK using m accepted declarations holds on $\geq 1 - m\varepsilon$ of the root rows all m cover, simultaneously, with probability $\geq 1 - \delta$. ■

Clopper–Pearson variant. The zero-violation rule rejects a true declaration with any nonzero natural noise rate ρ_0 once n is large enough ($(1-\rho_0)^n \to 0$). Accepting instead iff the exact Clopper–Pearson upper $(1-\alpha)$-bound on the sampled violation rate ($\alpha = \delta/K$, Bonferroni) is $\leq \varepsilon$ tolerates a few violations and degrades gracefully. It is what recovers rel-f1's 99.91%-true same-row equality once n is large enough to bound its 0.09% true rate tightly.

Keys are trusted, not tested. Declared primary/unique keys and FKs are assumed to hold (Definition 1's admissibility). Only copy/ownership declarations, our own "switchable" input, are sample-tested. Testing keys too would need a different (collision-style, two-row) statistic and is left open.

Headline property and what is new. n^* depends on K only through $\log K$, and the per-leak bound degrades only through derivation length m (bounded by declared FK-chain depth). Neither depends on the number of features, tasks, or (feature, target) pairs the accepted set later certifies. The concentration inequality itself is a standard Chernoff/union bound. Kivinen & Mannila (1995) introduced the g_1/g_3 error measures for approximate FDs, and HyFD (Papenbrock & Naumann, 2016) samples to propose FD candidates and then validates them exactly over the full data. What is new is testing at the level of the schema's K primitive copy candidates (one layer below the closure). This lets the guarantee propagate for free through Algorithm 1's own derivations into a space of certified determinations that can be exponentially larger than K (Theorem 1). It does this rather than needing its own sample budget per derivation or per (feature, target) pair.

Abstentions. An ABSTAIN here means "not implied by the *accepted* declaration set," not by the schema author's full intended set. A true declaration can be missing either because it was never proposed (a candidate-proposer coverage gap) or because it was sample-rejected (bounded by δ under the acceptance rule). By monotonicity (Corollary 4), every acceptance is safe to compose into the full schema's certified verdicts. Abstentions are not evidence of absence, exactly as for the schema-only detector alone (§3), now with the extra abstention source (sampling risk, quantified) made explicit.

Empirical validation. A Monte Carlo check (200 simulated candidates, 35% true at background noise rate 5×10^{-4}, 65% false at rate in $[0.02, 0.9]$, ε=0.01, δ=0.05, 300 trials/cell) confirms the bound: at $n \geq n^*(200, 0.01, 0.05) = 830$, the empirical false-accept rate is $0/300$, matching the theorem. Below n^*, where the bound is vacuous, the empirical false-accept rate correctly rises toward 1.

RelLeak-Synth, zero-violation rule. We sample-verify C1/C-all on 60 of RelLeak-Synth's 320 schemas ($n \in \{10, \ldots, 1000\}$, 3 seeds, δ=0.05) against our data-true labels (Table 14). Precision reaches 1.00 at $n \geq 30$ (against blind C1's 0.653), but recall is flat at 0.255 (against 0.620) and F1

plateaus at 0.406. Two of the benchmark's "declared" mechanisms (a bijective label permutation, a code→text description table) share a declared semantic type with the target but are not literal copies. The verifier (testing literal equality, Definition 1) correctly rejects them, at the cost of the recall blind trust gets for free. Rows read scale as $2Kn$: 216/task at n=10 for C1 (93× fewer than FD discovery's 20,097), reaching parity by n=1000 (Figure 7).

method	n	P	R	F1	rows/task
sample-verified C1 (ours)	10	1.000	0.255	0.406	216
sample-verified C1 (ours)	1000	1.000	0.255	0.406	21,640
sample-verified C-all (ours)	10	0.987	0.255	0.405	7,304
Algorithm 1 (C1, blind)†	–	0.653	0.620	0.636	0
FD discovery ($\lvert\text{LHS}\rvert \leq 2$)†	–	0.957	1.000	0.978	20,097

Table 14: RelLeak-Synth (60 of 320 schemas), 3 seeds, δ=0.05, zero-violation rule, against the our data-true labels. †Cited from Table 6, not re-run.

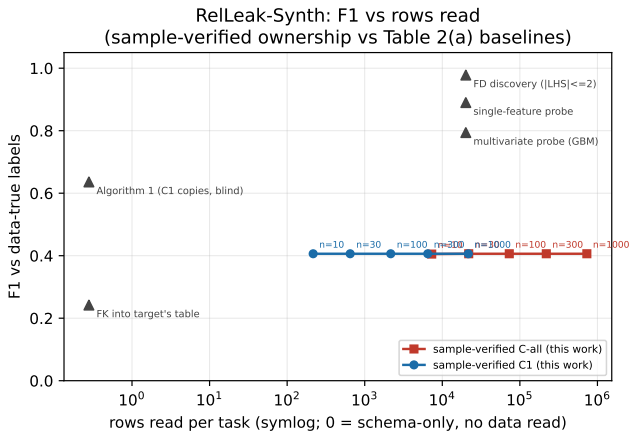


Figure 7: RelLeak-Synth: F1 against rows read per task (symlog axis), sample-verified C1/C-all at $n \in \{10, \ldots, 1000\}$ against Table 6's baselines.

Shared sampling and value maps. Shared sampling (one n-row sample tested by all K candidates) and a directed *value-map* candidate (accept iff no two sampled rows share $f(r).b$ with a different $r.a$) close both gaps. On all 320 RelLeak-Synth schemas, rows read fall from $2Kn$ to n plus distinct parent fetches, and value maps recover the bijective recodings with zero false positives. Recall rises from 0.252 to 0.750, and F1 from 0.402 to 0.857, at n=1000 (9,389 rows/task, under half of FD discovery's 20,097).

The certificates behind Table 2. On RelBench's rel-salt, the item's Incoterms copy of its header's is *not* exact (99.93% agreement), so the zero-violation rule's small-n acceptance is the wrong headline. It disappears once n sees the 0.07% defect. The headline is the Clopper–Pearson tolerant rule, its 95% coverage verified over 200 seeds (Table 15). One declaration, verified on 10,000 root rows (20,000 read), 7 violations, certifies the determination with defect at most 0.1314% at 95% confidence against a measured true defect of 0.0733%, name-blind. The same, coverage-verified treatment certifies item-plant ($\leq$ 0.07%, true 0.059%) and badges-class ($\leq$ 0.065%, true 0.060%, both under the split-sample value-map statistic defined below); eligibilities-adult stays a negative ($>$ 0.7% at 100,000 rows).

Proposing item-incoterms without hindsight. Literal-name matching (C1) cannot even propose this declaration: item and header Incoterms are named differently (`ITEMINCOTERMSCLASSIFICATION` vs. `HEADERINCOTERMSCLASSIFICATION`), so C1 yields one schema-wide candidate for this task and the pair is absent from it. A name-blind, type-compatible generator (C-all, 53 candidates on this task, no use of column names) does propose

it. Under the zero-violation rule of Theorem 3 it is accepted with zero sampled violations at every $n \in \{10, 30, 100, 300, 1000\}$ over 3 seeds, already 1,060 of item-incoterms's 2,319,540 rows at n=10. The closure then returns LEAK through the header column, reproducing this section's certified determination with no hand-written declaration. Results-position has no such analogue: same-row equality is outside the FK-based candidate grammar entirely, so a human still has to write it down; only the 95%-confidence bound then comes for free, tightening from $> 1\%$ at n=1000 to 0.46–0.61% by n=3000.

candidate (tolerant rule, δ=0.05)	n	rows read	certified ε (95%)	true defect	coverage (200 seeds)
item-incoterms (header → item, literal; 7 violations)	10,000	20,000	0.1314%	0.0733%	0.985 [0.957, 0.997]
item-plant (split-sample value map)	30,000	30,000	0.070%	0.059%	0.970 [0.936, 0.989]
badges-class (split-sample value map)	100,000	100,000	0.065%	0.060%	0.935 [0.891, 0.965]
eligibilities-adult (negative)	100,000	100,000	0.727%	0.674%	–

Table 15: Sample-verified certificates, tolerant rule as the headline (not the zero-violation rule's small-n acceptance, which correctly abstains at these n once it can see the defect). Coverage column: fraction of 200 independent seeds at this exact n whose certified ε was at least the measured true defect, with a two-sided 95% Clopper–Pearson CI on that fraction; all three CIs include or exceed the nominal 0.95.

The split-sample value-map statistic and its coverage. The g_3-style "excess" count compares each sampled row's a to the *majority* a observed for its b value. We estimate that majority on half the sample and score it on the other half, so the statistic never refers to itself. The naive alternative, estimating the majority from the very rows it scores, is self-referential: a lone or rare b-value's "sample majority" is unreliable and cannot disagree with itself, so the statistic under-covers, and the gap grows with n. For a high-cardinality b (389 distinct values) its nominal 95% coverage measures at 65% by n=100,000, over 20 seeds. Coverage of the split-sample statistic is measured over 200 seeds, since 20 seeds cannot separate 0.90 from 0.95 coverage, given $P(X \le 18 \mid X \sim \mathrm{Bin}(20, 0.95)) = 0.26$. It is 0.985/0.970/0.935 at the three quoted cells (Table 15). Each has a 95% CI that includes or exceeds the nominal 0.95. Point coverage declines mildly with the map's cardinality D (0.985 at D=0 (literal) to 0.970 at D=65 to 0.935 at D=126). This is not statistically established from three cells, but directionally consistent with the D-dependence of Appendix N, and worth re-checking before extending value maps to substantially higher D than tested here. That D-dependence is a property of the tolerant rule's statistic, not of Theorem 3's literal-copy bound, which estimates no majority and so has no self-reference at all (its own coverage, 0.985 at n=10,000, is the control case above).

N SAMPLE COST OF VERIFIED DECLARATIONS

Root rows are drawn i.i.d. uniformly (Remark 5 covers sampling without replacement). One sample is shared by all candidates. The *defect* $\rho(d)$ of an equality declaration $d = (f, a, b)$ is the mass of root rows where its lookup path resolves and $r.a \neq f(r).b$. The defect of a *value map* $d = (f, a, b, h)$ with h unknown is $g_3(b \to a)$ over root rows and its cardinality $D(d)$ is the number of distinct b-values. For F-groups write p_i for the mass of group i, q_i for its minority mass, $g_3 = \sum_i q_i$, g_1 for the mass of pairs agreeing on F and differing on T, and $D^* = \min(D, \varepsilon N/2)$.

Proof of Theorem 4(i), upper bound. A candidate with $\rho > \varepsilon$ survives n i.i.d. draws with probability $(1-\rho)^n < (1-\varepsilon)^n$. A union bound over the at most K such candidates gives $K(1-\varepsilon)^n \le \delta$. No independence between candidates is used, so one shared sample suffices. □

Proof of Theorem 4(i), lower bound. Let $M = 1/\varepsilon \in \mathbb{N}$. Take $A_j(k, b)$, $j \le K$, $E(e, f_1..f_K, a_1..a_K)$ with FKs $f_j : E[f_j] \subseteq A_j[k]$ and copies (f_j, a_j, b) (in Φ_0), and index root rows by the grid $[M]^K$, so a uniform row has i.i.d. uniform coordinates. Let $V_j = \{x : x_j = 0\}$, of mass exactly ε. In the far instance for j, a_j disagrees with its owner exactly on V_j, so $\rho_j = \varepsilon$. All free value labels are a uniformly random injection, independent over j. Put the uniform prior on the 2^K truth vectors and give the tester the sampled indices. Conditioned on them the observation splits into K independent parts. Part j determines b_j if some sampled row lies in V_j and otherwise has the same law under both hypotheses, so the posterior is the prior. The joint MAP rule is optimal for

"all K correct" and factorises, giving $\Pr[\text{all correct}] = (1-q/2)^K \le e^{-Kq/2}$ with $q = (1-\varepsilon)^n$. A worst-case-correct algorithm attains $\ge 1-\delta$ under this prior, so $q \le 2\ln\frac{1}{1-\delta}/K \le 4\delta/K$ for $\delta \le \frac{1}{2}$. For $K = 1$ and two-sided error, couple the two instances by the sampled index: they differ only when a sampled row lies in V_1, so $\mathrm{TV} \le 1-(1-\varepsilon)^n$ and $(1-\varepsilon)^n \le 2\delta$. □

Proof of Theorem 4(ii), lower bound. Hard family: $c = D-1$ conflicted F-groups of two rows each plus one clean group, $N = c/\varepsilon$. In the YES instance both rows of a group carry the authority's value. In the NO instance the second row of each conflicted group carries a fresh value, so $g_3 = c/N = \varepsilon$, the copy defect is ε, and $g_1 = 2c/N^2 = 2\varepsilon^2/(D-1)$. Randomise all labels by a uniform injection and couple the two instance distributions by the sampled indices: the observation (an equality pattern) differs only if two sampled rows are the two *distinct* rows of one conflicted group, whence $\mathrm{TV} \le \binom{n}{2} 2c/N^2 = n(n-1)\varepsilon^2/(D-1)$. Two-sided error δ requires $\mathrm{TV} \ge 1-2\delta$. The bound holds even for a tester that knows the whole F column, the group structure and A without its b column, all of which agree in the two instances. □

Proof of Theorem 4(ii), upper bound. Split the sample into independent halves S_1, S_2 of size k. If S_1 meets group i in a row labelled t', every group-i row labelled $\ne t'$ collides with it, a mass $\ge p_i - \max_t p_{it} = q_i$; so with $H_i = \mathbf{1}[S_1 \text{ hits } i]$ and $W \ge \sum_i q_i H_i$ we get $\Pr[\text{miss} \mid S_1] \le (1-W)^k \le e^{-kW}$. Multinomial occupancy indicators are negatively associated (Dubhashi & Ranjan, 1998) and $u \mapsto e^{-kq_i u}$ is non-increasing and non-negative, so $\Pr[\text{miss}] \le \prod_i \big(1 - h_i(1-e^{-kq_i})\big) \le \prod_i \big(1-\varphi(kq_i)^2\big)$ with $\varphi(x) = 1-e^{-x}$ and $h_i = 1-(1-p_i)^k \ge \varphi(kq_i)$. As $-\ln(1-\varphi(x)^2) \ge (1-\ln 2)\min(x, x^2)$, and as either the groups with $kq_i \ge 1$ carry minority mass $\ge \varepsilon/2$ (giving $\sum kq_i \ge k\varepsilon/2$) or the rest do (giving, by Cauchy–Schwarz and $q_i \ge 1/N$, $\sum (kq_i)^2 \ge k^2\varepsilon^2/4D^*$), $\Pr[\text{miss}] \le \exp\big(-(1-\ln 2)\min(k\varepsilon/2,\, k^2\varepsilon^2/4D^*)\big)$, which is $\le \delta$ for $k \ge \max\{6.52\ln(1/\delta),\, 3.61\sqrt{D^*\ln(1/\delta)}\}/\varepsilon$. Completeness is immediate: a YES instance contains no colliding pair. A δ-refined lower bound matching this form follows by Poissonising the same family: with $\Lambda \sim \mathrm{Poi}(2n)$ draws the $2c$ half-groups receive independent $\mathrm{Poi}(x)$ counts, $x = 2n\varepsilon/c$, so the no-collision probability is $(1-(1-e^{-x})^2)^c \ge e^{-cx^2/(1-x^2)}$, and de-Poissonising costs $\Pr[\mathrm{Poi}(2n) < n] \le e^{-0.3n}$. □

Remark 5 (Sampling without replacement)**.** The upper bound is unchanged (a without-replacement sample dominates the distinct rows of a with-replacement sample and detection is monotone). The lower bound becomes $n(n-1) \ge (1-2\delta)(D-1)(N-1)/(\varepsilon^2 N)$.

Remark 6 (Derivation length)**.** With the per-declaration rule, a certified LEAK of derivation length m holds on $\ge 1-m\varepsilon$ of root rows. The factor m moves into the logarithm: the conjunction of a derivation's declarations is itself a unary predicate and "all components accepted" is "the conjunction showed no violation", so $n \ge \ln(|\mathcal{D}|/\delta)/\ln\frac{1}{1-\varepsilon}$, with $\mathcal{D}$ the sample-independent family of derivations over subsets of the candidates ($\ln|\mathcal{D}| \le m\ln(eK/m)$), gives $1-\varepsilon$ for every certified derivation. A fresh second sample of $\ln(P/\delta)/\ln\frac{1}{1-\varepsilon}$ rows testing each of the P certified LEAK s end-to-end removes m outright.

Corollary 6 (Cost of a derivation)**.** *With one shared sample, K_E equality and K_M value-map candidates (cardinalities D_j^*), taking $n \ge \max\{\ln(2K_E/\delta)/\ln\frac{1}{1-\varepsilon},\ \max_j 2\lceil 3.61\sqrt{D_j^*\ln(2K_M/\delta)}/\varepsilon\rceil\}$ (with the $6.52\ln(2K_M/\delta)/\varepsilon$ term when it dominates) gives, w.p. $\ge 1-\delta$, defect $\le \varepsilon$ for every accepted candidate and $1-m\varepsilon$ for every certified LEAK. The cost charged to a LEAK is therefore the maximum over the steps of its derivation:* $\log$ *for equalities and low-cardinality maps, $\sqrt{D^*}$ for a high-cardinality map.*

Corollary 7 (Amortisation)**.** *The budget $\ln(K/\delta)/\varepsilon$ is independent of the number P of (feature, target) pairs, tasks and joins certified from the accepted set, provided they share the root table and hence the sampling distribution. It does* not *transfer across root tables, where a parent row's fan-in can inflate the defect. A schema-blind tester deciding P pairs from one shared sample needs $\Omega(\sqrt{D\ln(P/\delta)}/\varepsilon)$ and $O((\sqrt{D\ln(P/\delta)} + \ln(P/\delta))/\varepsilon)$ suffices.*

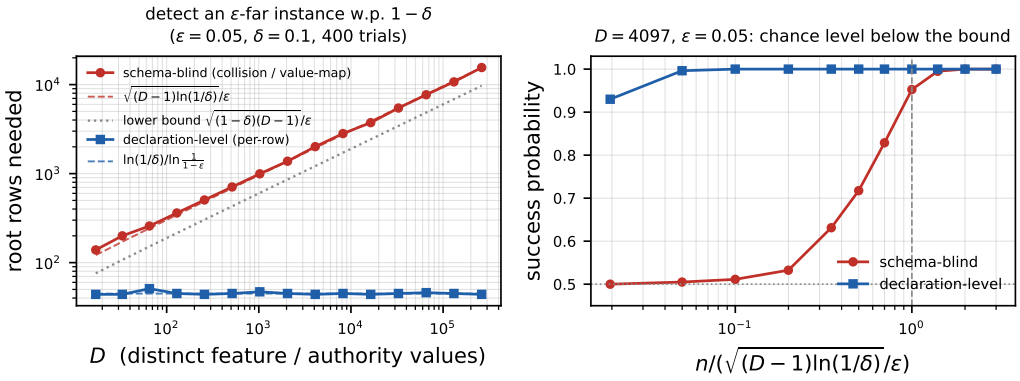


Figure 8: Root rows needed to detect an ε-far instance with probability $1-\delta$ ($\varepsilon = 0.05$, $\delta = 0.1$, 400 trials/point; hard family of Theorem 4). **Left:** the declaration-level budget is flat in D (44–51 rows over four orders of magnitude, against $\ln(1/\delta)/\ln\frac{1}{1-\varepsilon} = 44.9$), while the schema-blind budget follows $\sqrt{(D-1)\ln(1/\delta)}/\varepsilon$ within 3% and exceeds the proven lower bound $\sqrt{(1-\delta)(D-1)}/\varepsilon$. The ratio grows from 3.2 to 353. **Right:** at $D = 4097$ the schema-blind tester is at chance until n approaches the bound. Both methods scale as $1/\varepsilon$, so the $\sqrt{D}$ gap is ε-free.

O A DECLARATION FORMAT AND ITS INTEGRATION INTO RELBENCH AND 4DBINFER

A declaration file is a small JSON/YAML sidecar, written once by the dataset or benchmark author beside the manifest they already ship (RelBench's `manifest.yaml`, 4DBInfer's `metadata.yaml`). An adapter merges it into the closure detector's schema with *no change to the detector itself* (Algorithm 1, unmodified). Each entry declares one target attribute a of a table R and one of four forms:

- **copy**: Definition 1's (f, a, b), holding on every row.
- **default**: an *overridable* copy (ERPNext's `fetch_if_empty`: the value is copied unless already set), holding up to a certified defect rate rather than universally.
- **value_map**: $a = g(b)$ for a declared, possibly non-identity function g (Odoo's non-stored-through `related`, a bijective recoding, a threshold). It is fed to the detector through the sample-verified pipeline's post-closure fixpoint rather than as an equality, since unioning a non-injective map's endpoints as one closure class is unsound.
- **same_row**: $a = b$ within one table, no FK (RelBench's own `results.position` = `results.positionOrder`), outside Φ_0's FK vocabulary entirely but already supported by the detector's `row_eq` mechanism.

Every declaration also carries `provenance` (who declared it and how: domain knowledge, source documentation, mined from code or data) and an optional `verification` block (sample size, violations, and a Clopper–Pearson defect bound at a stated confidence). Both are filled in by a validator that checks the file against a JSON Schema, reconstructs Definition 1's S1–S3 fragment-membership conditions, and PAC-verifies each declaration on a data sample (Theorem 3) using the same, unmodified detector. On RelBench's own rel-salt and rel-f1 manifests, one such declaration each turns the closure's verdict from ABSTAIN to a certified LEAK. For item Incoterms this is a separate, larger-sample verification of the same declaration as §5's headline, run independently by this format's own validator: 20,000 sampled root rows, 13 violations, defect bound $\leq 0.1033\%$ at 95% confidence. This cell's coverage was not separately checked, which is why §5 leads with the 10,000-row, coverage-verified cell of Table 15. `position`, defect bound $\leq 0.20\%$. Of RelBench v2's own 90 audited hand-removed columns, exactly 7 need any declaration at all, only one of them

FK-based. The authoring cost is 0.30 declarations per task, each 250–700 characters: one short, human-readable file per benchmark, not a per-audit re-derivation.

```
{ "id": "salesdocumentitem.ITEMINCOTERMS_default",
  "kind": "default", "overridable": true,
  "table": "salesdocumentitem",
  "attribute": "ITEMINCOTERMSCLASSIFICATION",
  "on_fk": {"column": "SALESDOCUMENT"},
  "references": {"table": "salesdocument",
                 "attribute": "HEADERINCOTERMSCLASSIFICATION"},
  "provenance": {"declared_by": "rel-salt maintainers",
                 "method": "source_documentation",
                 "notes": "item Incoterms defaults from the
                           header; overridable per item"},
  "verification": {"verified": true, "sample_size": 20000,
                   "violations": 13, "defect_bound": 0.001033,
                   "confidence": 0.95, "method": "clopper_pearson"} }
```

P VENDOR-DECLARED OWNERSHIP

We read the schemas of ERPNext, Frappe and HRMS (branch version-15; DocType JSON) and Odoo (branch 18.0; model source parsed with Python's `ast`), with no server installed and no row read. Adapters map a DocType or a regular Odoo model to a relation keyed by `name` or `id`. They map a Link field or a stored Many2one to an FK. They map `unique`, naming fields and `UNIQUE` constraints to secondary keys. They map a child table to an FK to its parent when a single DocType embeds it (43 shared children are polymorphic and dropped). They map `fetch_from:` $l.b$ on field a to the copy (f_l, a, b), and `related:` $f_1.\cdots.f_k.b$ to a chain of k copies through bridge attributes. Hops through non-stored related fields and delegation inheritance (`_inherits`) are expanded, hops through one-to-many fields (240) are not functional and are skipped, and non-stored related fields (641) are declared equalities without a column. Of 580 `fetch_from` declarations, 555 resolve; 76 carry `fetch_if_empty` (a default the user may override) and 188 copy into a link. Of Odoo's 169 stored copies, 113 copy into a Many2one. Frappe re-applies a fetch on every save until a document is submitted but not when the source record changes later. Odoo recomputes stored related fields when any field on the path changes. Both declarations therefore hold at the time a record is written, not necessarily in a dump after later master-data edits. Outside Φ_0 lie copies into links and one-relation copy cycles along self-references (3 in ERPNext, 1 in Odoo). Restricted to copies that cannot be overridden and target descriptive attributes, only two violations remain. One is a location copying its coordinates from its parent location (ERPNext). The other is a company name that is both unique and copied from the company's partner, which leaves that key without a free attribute (Odoo). Views, candidates and verdicts follow Appendix F. Under C0 no target is certified. Under the declarations, every certified target is a declared descriptive copy, with JOINT never the only verdict. C1 guesses 2,151 and 2,676 copies. Its misses are renamed copies (83 and 45), same-name copies into links, which the descriptive-target rule excludes (160 and 75), and copies whose declared types differ, such as Data and Small Text (91 in ERPNext). Dropping the type gate raises recall to 84% and 74%, still at 10% and 3% agreement. C2 reaches 33% and 66%, all copies into links, at 2% and 1%. Agreement is a lower bound on precision, since both ERPs also copy values in code without declaring them. The shipped demo fixtures leave every declared copy target empty (ERPNext) or write it through the ORM (Odoo), so this check is schema-only. For rel-salt, the dataset's own paper declares keys and FKs and notes that Incoterms "can be defined independently on item and header level". RelBench renames the two columns, and the public OData metadata of the standard sales-order API declares keys, navigation properties and 16 same-named header and item properties, but no referential constraint, value list or derivation. Algorithm 1 certifies nothing on the item under these declarations, while C1 on the vendor's names guesses 15 item-from-header copies, the Incoterms among them.

A third, independent ERP. iDempiere (760 tables, 16,627 columns, GPL-2.0) shows the same split. Unlike ERPNext, its schema-synchronization code emits real primary-key, foreign-key, unique, and check constraints, confirmed by reading `MColumn.java` and against shipped migration DDL. Its Application Dictionary also declares field-level copies. An `IsTranslated` flag drives a real, stored, foreign-key-linked duplicate column in a companion `_Trl` table, with no database-level trace of the equality. A separate `ColumnSQL` construct declares further copies and value maps that are never stored at all. On the subset minable from public migration history

(11 declarations, 6 tables; the live Application Dictionary is not distributed with the source), the dump-level closure certifies 0 of 11,957 targets and the declaration-level closure certifies 11. This reproduces the same zero-versus-nonzero contrast as both other systems. This subset is a stated lower bound, not a full-schema count, and its guessing figures (Table 16) are not comparable in absolute scale to ERPNext's and Odoo's, which cover their entire schemas.

		declared copies			C1 vs. declared		LEAK targets	
vendor schema	tables	copies	renamed	into FK	P	R	C1	declared
ERPNext v15 (`fetch_from`)	838	555	83	188	10%	40%	1,351	313
Odoo 18.0 (`related`, stored)	655	169	47	113	1%	21%	1,285	55
sales-order OData (public EDMX)	10	0	–	–	–	–	15	0
iDempiere 13 (`ColumnSQL`/`_Trl`)[‡]	760	11	–	–	23.4%	100%	–	11

Table 16: Ownership declared by software vendors (schema only, 0 rows). Copies: resolved declarations; into FK: copies into a link attribute, outside Φ_0. C1 vs. declared: precision and recall of same-name, same-type guesses against the vendor declarations. LEAK targets over all descriptive targets of all tables as roots (6,530 and 5,443; 0 under C0); declared: copies that cannot be overridden and target descriptive attributes (ERPNext: 367 when overridable copies are included). The sales-order row infers FKs from navigation properties; C1's 15 guesses include the header-to-item Incoterms. [‡]iDempiere's 11 copies and 760 tables are exact, but its P/R/declared columns are measured only on the 6-table, 11-declaration subset recoverable from public migration history (its live Application Dictionary is not in the source repository), so its guessing figures are not comparable in absolute scale to ERPNext's and Odoo's, which are measured over their entire schemas.

Q WHAT A SCHEMA EXPORT CARRIES

Our audit (§5.2) may measure what a public export exposes, not what the source system declares. If ownership is declared in an application or ORM layer and simply never reaches a dump, the negative result is conditional on a corpus's provenance, not a fact about "real schemas." We test this on the two cases where an independent application layer can be inspected: the two vendor ERPs of Appendix P.

The visibility ladder. Table 17 lists, for each corpus, the highest declaration class its audited artifact actually exposes. A *dump* (SQL DDL export, or an `information_schema` read against a live server) carries primary keys, unique constraints, indexes and defaults, and carries a foreign key only if the source system emits one at the database level. *Committed DDL text* (SchemaPile) carries whatever of that a maintainer chose to check into version control as literal SQL. This is a strict subset of dump level for any system whose schema is generated by ORM code at install or migration time. A *maintainer manifest* (RelBench, 4DBInfer) carries whatever the benchmark's curators typed into a loader file, authored independently of any source system. For several 4DBInfer tables, this describes tables the curators invented from flat logs that never had a relational schema to begin with. Only the *application or ORM layer* carries ownership copies (`fetch_from`, `related`) at all, and none of the four export levels above it does.

The two ERPs, two levels apart. Frappe's MariaDB DDL generator (`frappe/database/mariadb/schema.py`, `MariaDBTable.create()`) builds every table's `CREATE TABLE` statement from a declared column list, a primary key on `name`, and inline `UNIQUE` constraints. A search for the string `FOREIGN KEY` across the whole Frappe and ERPNext checkout returns zero matches. No table, of 838, ever receives a database-level foreign key or check constraint. The 2,911 Link-field edges used as foreign keys in Appendix P's C0 baseline are themselves application metadata (DocType JSON), read by an adapter that never touches SQL. Rerunning Algorithm 1 at the true dump level (`fks`=$\emptyset$, everything else unchanged) certifies 0 of 9,169 targets. The target count rises from 6,530 because, with no foreign keys, the 2,639 Link-field columns stop being excluded as FK sources and become plain descriptive targets ($6{,}530 + 2{,}639 = 9{,}169$). Odoo's ORM does issue genuine PostgreSQL `FOREIGN KEY` constraints, verified against `pg_constraint` at every schema sync (`odoo/modules/registry.py`, `check_foreign_keys`), so its dump level and Appendix P's C0 baseline coincide. Table 18 restates both ERPs' numbers across the levels this makes visible.

Table 17: Declaration classes actually carried by each audited artifact, not by the source system in general. PK/UNIQUE/FK are database-level constraints. "Copies" are row-aligned ownership declarations, as defined in §2. The FK column for CTU reads databases declaring $\geq$1 FK / content databases (App. J).

Declaration class	Corpus / artifact	PK	FK	Copies
Dump, live server read	CTU (§J)	yes	119/145	never
Committed DDL text	SchemaPile (Döhmen et al., 2024)	yes	mixed	never
Maintainer manifest	RelBench v2 (Gu et al., 2026), 4DBInfer (Wang et al., 2024)	yes	authored	never
Dump, no FK ever emitted	ERPNext/Frappe DDL generator	yes	**0/838 tbl**	not in dump
Dump, real DB FKs	Odoo PostgreSQL DDL	yes	yes (1,985)	not in dump
Application / ORM layer	ERPNext `fetch_from`, Odoo `related`	yes	yes	**555, 169**

Table 18: ERP declaration visibility, from a database dump up to the vendor's own application-layer declarations. Certified: targets Algorithm 1 certifies at that level. ERPNext's dump level is strictly poorer than its own C0 baseline (Appendix P), because that baseline was already application metadata, not a database export. Odoo's dump and C0 coincide because its FKs are real.

Level	FK edges	Copies	Certified	Targets
ERPNext, SQL dump	0	0	**0**	9,169
ERPNext, app-inferred FKs (our C0, App. P)	2,911	0	0	6,530
ERPNext, vendor's `fetch_from` declarations	2,911	555	**313 / 367**	6,530
Odoo, SQL dump = app-inferred FKs (C0)	1,985	0	0	5,443
Odoo, vendor's `related` declarations	1,985	169	**55**	5,443

The Financial quartet. CTU's own corpus statistic (Appendix J) is subject to the same effect, in miniature. The repository hosts the classic PKDD'99 Financial dataset four times, under four names, at four declared-visibility levels (Table 19). `ctu_financial` has the same table set and, where checked, the same `account`-table columns as `financial`, with every key and foreign key stripped. Our closure sweep (`closure/out/ctu/dbs.csv`) counts all four as independent database rows, so one real-world schema is counted once as fully declared and, separately, once more as an instance of a schema declaring nothing.

What the two corpora can and cannot settle. For the two ERPs there is code-level proof that ownership exists in the source system and is stripped on export: the vendor's own repository, read independently of any dump, shows the declarations a dump omits. For the ten value-first findings in the CTU relational pool (Appendix X), there is no application layer to check against. Each is a standalone research or administrative database with no known vendor codebase behind it, so it cannot be determined whether the missing declaration was lost on export or never written by anyone. Those ten findings therefore measure what a schema-only reader necessarily misses in this corpus; the vendor corpus, not this one, is what establishes that ownership is stripped on export.

Reframing. "Real schemas certify almost nothing from keys and foreign keys alone" is accurate as a statement about SQL dumps, committed DDL text and maintainer-authored benchmark manifests, the artifacts we read. It is false, however, as an unqualified claim about the two source systems whose application layer we can inspect directly. The defensible statement, used throughout §5 after this appendix: public exports of real systems carry too little to certify ownership from keys and foreign keys alone. Source systems with an application or ORM layer declare substantially more, and none of it reaches a dump.

Table 19: The Financial database, hosted four times by CTU. Table set and, for the account table, column names are identical across all four. Only declared structure differs.

Database	Tables	Declared PK	Declared FK
`financial`	8	8	8
`Financial_ijs`	9	9	8
`Financial_std`	7	7	6
`ctu_financial`	8	**0**	**0**

R PUBLIC BENCHMARK ECOSYSTEM AUDIT

This appendix extends our public-corpus audit (§5) to the broader ecosystem of benchmarks used to train and rank relational deep learning (RDL), GNN, and relational foundation models (RFMs) since 2023.

RelBench v1 vs. v2. RelBench v1 (Robinson et al., 2024) ships 30 official tasks over 7 databases (103,466,370 rows, 51 tables, 489 columns), all of type entity-classification, entity-regression, or recommendation, each defined by a seed-time cutoff over *future* rows (e.g. `user-churn`). RelBench's own temporal neighbour sampling guards against exactly this task family's risk, and its authors state it "systematically avoids time leakage." Because v1's targets are computed aggregates rather than values stored anywhere in the schema, most of v1's 30 official tasks fall outside the whole-key closure's applicability domain (structural, declared-schema determination). Our audit is therefore scoped to targets that are stored values, which is a property of the question, not a limitation of the run. RelBench v2 (Gu et al., 2026) instead ships 23 *autocomplete* tasks over 9 datasets ("predict an existing column," with 90 columns manually removed by the designers as known leakers). This is squarely in-domain and is the version audited in §5 (23/23 abstain under C0 and C1). Only 2 of the 90 hand-removed columns are recoverable from declared metadata, and only once an explicit ownership declaration absent from the official schema is added. Eleven columns drawn directly from v1's raw per-dataset schemas that could plausibly be posed as autocomplete-style targets (e.g. `rel-f1` `results.position`, `rel-trial` `studies.enrollment`) were tested against the same closure and also abstain in every case. An earlier, cruder detector implementation had over-flagged 11 such columns, and the false fires trace to join-handle foreign keys (an FK into the target's own table) rather than genuine value copies.

CTU. The CTU Prague Relational Learning Repository underlies a large fraction of pre-2023 relational learning and several RDL papers' baselines. Restricting to the repository's 100 documented prediction targets across databases with declared primary and foreign keys, the closure flags 0 under C0 and 2 under C1. Both are traced to name/type coincidences refuted by a direct value check, not genuine functional determinations. The four targets most commonly designated in the RDL literature (`financial`, `mutagenesis`, `Hepatitis`, `imdb`) all abstain under every copy policy tested.

4DBInfer. 4DBInfer (Wang et al., 2024) spans 8 datasets from 2.7M to 2B rows with click-through, conversion, churn, rating, and citation-style tasks. We downloaded the six datasets under our compute budget (amazon, mag, diginetica, retailrocket, seznam, stackexchange; outbrain and avs, at 2B and 350M rows, were skipped) and audited every task's officially declared metadata directly. Of 12 tasks, 8 have a target that is an actual stored column elsewhere in the schema (in-domain for a copy detector). The other 4 (e.g. diginetica's click-through label) are materialised only inside the task's own split files, like RelBench v1. All 8 in-domain tasks **abstain under C0 and C1**, on both the officially shipped feature view and an unrestricted 3-hop view. The sample-verified verifier of Appendix M independently certifies the same 8 targets using **0** sampled rows: 5 of 6 schemas have no same-name/type FK-copy candidate to even propose (their FK targets are near-bare key tables). On a non-official schema-level check (stackexchange's `Comments` table, 2 candidates) the verifier correctly rejects both after reading 7,968 real rows.

A value-level audit went further than declared keys, covering:

- Root-row columns (12 checks).

- Every column reachable by 1–2 hops of many-to-one lookup, and co-key sibling tables sharing a business key without a declared FK (e.g. RetailRocket's item-availability/category tables, asof-joined on the view's own timestamp).
- One-to-many aggregates outside the closure's class (Proposition 2): leave-one-out sibling means/modes and, for MAG, the mode topic/author and the mode venue-label of a paper's cited and citing neighbours.

39 checks in total. A feature counts as determining only if (i) its held-out margin over the majority/mean baseline has a 95% bootstrap CI excluding zero and is $\geq$ 2 points. It must also satisfy (ii): an exact-FD check on repeated keys shows $\geq 95\%$ constant groups over ≥ 100 groups. **Zero of 39 pass both.** Two candidates cross a raw 0.95 accuracy threshold on their own (RetailRocket `itemid→added_to_cart` in the root-column pass, and an asof item-category lookup in the joined pass). But their margins over baseline are indistinguishable from zero once tested. Several joined/aggregate features carry a large, statistically real, but far-from-deterministic association. A client's own past service choices predict a new one by up to 35 points over baseline (exact-FD 0–17%), and a paper's citation neighbourhood's venue predicts its own by up to 36 points (exact-FD 0–4%). This is legitimate relational signal, not a copy. No feature triggers the impact step. This 100-group floor is the same bar that Appendix T's positive control later shows to be unreachable on low-cardinality targets, so these zeros are reported as what the bar found, not as evidence of absence. Separately, and independently of our own audit, 4DBInfer's authors self-report hand-removing two same-row leaks with no formal ownership mechanism to justify the fix generally. These are `Review.review_text`/`review_time`/`summary` for the Amazon rating task, and the raw `Paper.label` column (which *is* the target) for the MAG venue task. Both are independent, third-party corroboration of our central claim, one from the published text and one from our own reproduction.

Relational foundation model evaluation pools. The public corpora audited here are not a narrow slice of the RDL/RFM literature: KumoRFM-2 (Kumo, 2026) states its 41-task evaluation draws on "RelBenchV1, RelBenchV2, SALT and 4DBInfer". Griffin (Wang et al., 2025) pretrains and evaluates on 4DBInfer and RelBench over 150M+ nodes. ContextGNN (Yuan et al., 2025) is evaluated within RelBench's recommendation tasks. A schema-only audit of RelBench and 4DBInfer is therefore, to first order, an audit of the substrate the current relational foundation-model leaderboard is ranked on.

Benchmark	Targets	C0/C1 flagged	Det.	Basis
RelBench v2 (23 autocomplete tasks)	23	0/0 (2/90 w/ decl.)	0	reproduced
CTU designated targets	4	0/0	0	reproduced
CTU documented targets	100	0/2 (both false positives)	0	reproduced
RelBench v1 (raw-schema candidate cols.)	11	0/0	0	reproduced
RelBench v1 (30 official temporal tasks)	30	n/a, out of domain	n/a	analysis
4DBInfer (8 in-domain tasks, 6 datasets)	8	0/0 (0 rows)	0/39 (+2 lit.)	reproduced

Table 20: Public RDL/RFM benchmark corpora: closure verdicts on officially shipped task inputs. C0 = keys/FKs only, C1 = + same name/type along one FK. "Det." = data determinations found under the 100-group bar of this appendix (see Appendix T for that bar's known blind spot).

S CROSSING THE CERTIFIED BOUNDARY ON THE MODEL SIDE

This appendix extends the model comparison to tasks whose leak is a one-to-many aggregate, outside Φ_0, and measures what a model gains when the certified boundary is crossed on the model side rather than the detector side. We add two cases to rel-salt sales-incoterms (already in Table 2): 4DBInfer's seznam/prepay and mag/venue tasks (Wang et al., 2024).

seznam/prepay. The official task view ships only {client_id, date, sluzba, kc_proklikano}. We construct (not shipped) an as-of, backward-only join of the sibling event table Dobito's sluzba column by client_id. Dobito and Probehnuto share the business key client_id via Client but have no declared FK between them. A HistGradientBoosting model with CLEAN features {Client.kraj, Client.obor (a declared 1-hop lookup), kc_proklikano, date-derived features} scores 62.7% test accuracy (official temporal split, majority baseline 33.1%). Adding the as-of feature raises this to 74.9%, +12.2

[+12.0, +12.4] points, 5 seeds. The standalone feature's own association (a 50/50 held-out lookup, no other features) is +35.1pp [+34.7, +35.5], with an exact-FD fraction of 0.00% over 8 repeated groups, a real, large, non-deterministic association, not a copy. Algorithm 1 ABSTAINs (no declared FK connects the two tables), and the verifier proposes zero C1 candidates for the whole schema (no same-name/type pair exists along any declared FK), certifying ABSTAIN with 0 rows read.

mag/venue. Paper declares no outgoing FK at all, so the only relational signal beyond a paper's own 128-dim embedding is the reverse structure of the Cites table. We construct (not shipped) the mode label of a paper's cited and citing neighbors, using only train-split-known labels (no leakage of a validation/test row's own split-mates). Standalone associations are +35.6pp [+35.2, +35.9] (citing) and +34.2pp [+33.9, +34.6] (cited), with exact-FD fractions of 0.29% and 0.00%, citation homophily, not a copy. Native multiclass gradient boosting did not converge reliably on the official 349-way task within an affordable round budget. A GBM using the mode-label feature alone reaches 32.7% at 50 rounds, matching a direct-lookup ceiling of 34.8%. Combined with the embedding under 150–500 rounds, the CLEAN-vs-LEAKY margin ranged from +2.9pp to −1.5pp across configurations. This is because per-class-tree multiclass boosting spreads gain across 349 one-vs-rest sub-problems, and a feature informative for a subset of classes competes poorly, at affordable round counts, against an embedding with small gains spread across many classes. We therefore report the model-impact number on a stated restriction of the evaluated population to the 20 most frequent train-split classes (40–45% of rows in every split). The restriction applies to which rows are scored, not to the mechanism, the official columns or the metric. It was imposed post hoc, after the full 349-way model failed to converge in budget, and was not pre-registered. CLEAN (embedding only) scores 44.6% test accuracy (majority baseline 3.1%). LEAKY (+ mode label features) scores 54.1%, +9.5 [+8.9, +10.1] points, 5 seeds. Algorithm 1 ABSTAINs (no FK to walk), and the verifier proposes zero C1 candidates for the same reason.

CTU Accidents: a second official instance. A person's injury code `oseba.poskodba` is a declared child of the accident record `nesreca` via `oseba.id_nesreca`. The plain mode of a family's injury codes reaches 76.5% accuracy on the accident's severity class `klas_nesreca`, +8.4 points over a 68.1% majority baseline. The best of 120 severity orderings, *fitted to this data, not declared anywhere in the schema or found documented externally after a bounded search*, reaches 99.3% on the 7,157 multi-person accidents. Isolated from every other feature, a gradient-boosted classifier moves from 45.3% to 99.8% macro-F1 (5 seeds, +54.5 [+51.8, +57.3] points). The closure ABSTAINs either way: candidate views follow foreign keys only child-to-parent, and `oseba`'s key points the other way, so no aggregate of its children, fitted or declared, is in the candidate language. Unlike sales-incoterms, CTU declares no temporal split, so availability here is argued, not verified.

An exhaustive negative. Beyond sales-incoterms, a scan of RelBench's 14 remaining off-view removed columns (most with strongly negative margins) and, citing Appendix R's 39-check audit, all in-domain 4DBInfer tasks found no further official instance. About 257 candidate pairs were checked in total across this appendix's own scans and the cited exhaustive work of Appendices R, G and J; only sales-incoterms and CTU Accidents survive as official mechanisms outside Φ_0.

The verifier cannot form the hypothesis, let alone reject it. For sales-incoterms we additionally check what the verifier's own candidate generators can *propose* for the header target salesdocument.HEADERINCOTERMSCLASSIFICATION. Every candidate of the C1 and C-all generators has its copy-target attribute restricted to the FK's source (child) table. Since only salesdocumentitem carries a declared FK, no candidate can ever name salesdocument as its source. Result: 0 candidates from C1, C-all (132 schema-wide) and same-row (242 schema-wide) alike, against 12 real C-all candidates for the in-class sibling target salesdocumentitem.ITEMINCOTERMSCLASSIFICATION. Nothing is sampled and rejected here: there is no hypothesis to sample.

Boundary statement. Of these four out-of-class mechanisms, two are official. Sales-incoterms is a RelBench v2 hand-removed column, fully declared as a mechanism and temporally verified. CTU Accidents requires a fitted aggregate (the best of 120 orderings, not declared or found documented externally) and an availability argument rather than a proof, since CTU declares no temporal split to check against. The other two (seznam/prepay, mag/venue) are features we construct from public tables, not official task inputs. All four carry standalone associations (+8.4 to +35.6 points, or macro-F1 for Accidents) of the same order as our certified in-class model-impact numbers (+28.0, +62.9). They also carry marginal model impacts that remain large (+12.2, +9.5, +54.5 points)

even once ordinary in-class relational features are already in the model. On every case the closure ABSTAINs and, where checked, the verifier cannot even represent the hypothesis, not because no declaration was supplied, but for one of two structural reasons. For sales-incoterms and CTU Accidents, both one-to-many child aggregates, Proposition 2 proves no declaration in Φ_0's vocabulary could ever force it. For seznam/prepay, an as-of join between sibling tables with no declared FK between them, and mag/venue, where Paper has no outgoing FK at all, the barrier is Definition 2: a feature view is a tree of many-to-one FK lookups, and no such path exists. Mechanisms outside the class can be as damaging as the certified ones, and the method is silent on them by proof, not by accident.

The aggregate on a graph neural network. The measurements above use gradient boosting. Because the mechanism here is a one-to-many aggregate, which message passing computes natively, we repeat sales-incoterms with RelBench's own heterogeneous GNN at its published defaults, 5 seeds, early stopping on a validation split, evaluated on all 88,942 official test rows. Accuracy is 67.01 [66.55, 67.50] without the child Incoterms column, 99.81 [99.78, 99.84] with it, and 66.81 [66.51, 67.12] when that column is present but its values are permuted within the split. The aggregate is therefore worth $+32.81$ [$+32.31$, $+33.28$] accuracy points to the GNN, against -0.20 [-0.76, $+0.26$] for the permutation, so the gain is the child-to-parent alignment rather than the presence of a column. The majority-class baseline is 56.62. The effect is larger than the $+22.4$ measured for gradient boosting, though partly because the GNN's clean baseline is about ten points weaker (67.0 against 77.3); both models reach the same 99.8 ceiling once the aggregate is available. The closure abstains in every arm.

mechanism	class (provenance)	measurement	evid.	effect (acc. pts)
rel-salt item-incoterms (copy along FK)	in-class	GBM, 5 seeds	C	$+28.0$ [$+27.8$, $+28.3$]
rel-salt sales-incoterms (mode of children)	outside, official column	GBM, 5 seeds	M	$+22.4$ [$+22.2$, $+22.7$]
seznam/prepay (as-of sibling lookup)	outside, constructed	GBM, 5 seeds	M	$+12.2$ [$+12.0$, $+12.4$] (marg.)
mag/venue (mode label of neighbours)	outside, constructed	GBM, 5 seeds*	P	$+9.5$ [$+8.9$, $+10.1$] (marg.)
CTU Accidents (fitted aggregate of children)	outside, fitted†	GBM, 5 seeds	P	$+54.5$ [$+51.8$, $+57.3$]†

Table 21: Model impact of certified vs. out-of-class mechanisms (points of the official metric except where noted; 95% CI over test rows and seeds). Of the four out-of-class mechanisms, two are official benchmark columns: sales-incoterms, fully declared as a mechanism and temporally verified, and CTU Accidents, whose determining aggregate is fitted to the data (the best of 120 severity orderings) rather than declared or externally documented, and which lacks a declared temporal rule to check availability against; seznam/prepay and mag/venue are features we construct from public tables, not official task inputs. Evid. (evidence status, added for auditability, no number changed): C = certified by the closure from a declared, in-class mechanism; M = measured on a single prespecified mechanism and population, not closure-certified; P = obtained by post-hoc, unregistered search or restriction (CTU Accidents' fitted ordering; mag/venue's class restriction), both already flagged in-line where they occur. *mag/venue evaluated on a 20-most-frequent-class restriction of the population (40–45% of rows), not the official 349-way task, chosen post hoc after the full 349-way model failed to converge in budget, and not pre-registered. †CTU Accidents' effect is macro-F1 points on the isolated 5-column feature (not accuracy points on the official metric, unlike the other rows); the whole-table, cited figure is $+66.0$ [$+64.8$, $+67.3$] macro-F1 points.

T CALIBRATING THE EVIDENCE BAR WITH A POSITIVE CONTROL

Sections 5.2 and 5.1 test Algorithm 1 on schemas as declared and on RelBench's hand-removed columns re-inserted. Neither tests an uncurated pool of ordinary columns, nor validates the data-confirmation rule itself against a known ground truth. This appendix does both.

Positive control. 20 tasks were sampled round-robin across CTU databases, with root tables restricted to those with a declared PK. Definition 1's (S3) excludes every column of a PK-less table from ever being a copy target, confirmed directly by instrumenting the C1 proposer. Each task got three synthetic 1:1 lookup tables planted: an exact copy, a hashed bijective recoding, and a 90%-noisy copy. Each was joined by one new declared FK, named identically to the target so the same-name/type adapter proposes them exactly as it would a real satellite table. Run through the

unmodified screen, verifier and confirmation step: the screen flags all three at 100%. The literal test accepts only the exact copy. The value-map test additionally accepts the bijection. The effect-size criterion passes 90% of all three (mean margins 46.3/46.3/37.0 points). The four evidence bars then separate sharply (Table 22): the **structural** bar (purity 1.0 on however many repeated groups exist) recovers **all 20** exact and **all 20** recoded plants and **rejects all 20** noisy ones. The Clopper–Pearson bars recover 8 and 6 of 20 and also reject all noise. The **strict bar recovers nothing at all, not even the known exact copy**. Purity is 1.0 on all 20 exact and all 20 recoded plants. On the noisy plant it is 0.0 on 15 tasks, 0.5 on two and 0.105 on one. The mechanism is exact: a planted copy inherits the target's own cardinality, and this pool's rule caps every target at 2–50 classes, so no copy of it can ever produce $\geq$100 repeated groups. **The strict bar does not recover a known true positive on this population.**

A zero count at the strict bar is uninformative. The control recovers no known planted exact copy at the strict bar under identical conditions, so a zero count there does not bound a pool's copy content. On the CTU-24 pool's 355 flagged pairs the strict bar likewise returns 0 confirmed (Table 22), a count the control shows this bar returns whether or not copies are present. Uncurated-pool counts are therefore reported at bars the control shows can recover a known plant.

Three bars the control shows can recover a plant. Uncurated-pool counts are reported at three bars. (i) **structural**: perfect purity on however many repeated groups exist, no floor. (ii) **Clopper–Pearson bar**: a one-sided 95% lower confidence bound on the true purity rate given the observed (pure, total) repeated groups, thresholded at 0.5 and 0.8, combined with the same margin/CI gate. This is statistically valid at any group count, unlike a raw $\geq$95% point estimate. (iii) **eligible-100**: the strict bar's purity threshold, with the confirmed count reported only among pairs whose fully-read table actually has $\geq$100 repeated groups (a population fact here, not a sampling shortfall). Ineligible pairs are excluded from the denominator rather than counted as failures. The residual non-recovery of a known exact copy under (ii) is a property of the evidence, not a mistuned threshold: several planted tasks have as few as 1–3 repeated groups, and no purity test can reach high confidence from that little evidence. The margin criterion (90% pass rate throughout) is doing the real detection work when group counts are low, and purity-based confirmed counts are a lower bound on true prevalence, not a census of it.

Purity here is a census, not a sampled estimate. Every purity number in this appendix (structural, Clopper–Pearson bar, and eligible-100 alike) is computed as a census over the fully read table. It groups rows by feature value and asks directly whether every row in a repeated group carries the same target value, rather than estimating a per-group majority from a sample and then scoring that same sample against it. That sample-reuse construction is exactly the self-referential majority Appendix M's split-sample statistic avoids. Since no sampling occurs anywhere in this pipeline, the self-referential path is never available here.

Self-pairs as a confound. Widening the CTU corpus to 74 databases and adding a high-cardinality-target stratum (CAT_HI, >50 distinct values) gives the first non-zero strict-bar count anywhere in this analysis. This count is 2 of 64 resolved CAT_HI pairs. The pair is `Countries.Data."Country Name"` against `"Metadata - Countries"."Country Code"` and `"...".."Country Name"`. The second of these is a **self-pair**: feature and target share a column name across tables, a near-tautological mirror, not independent evidence. It is excluded. The first (code$\rightarrow$name, a genuine if substantively trivial reference-table lookup, margin $\approx$99.7 points, 247 pure groups) survives. Self-pairs are a real confound in the widened pool (29 of 60 CTU-74 CAT_2–50 resolved pairs; excluding them drops that stratum's structural copies from 9 to 0) and are filtered everywhere in Table 22. WikiDBs has zero self-pairs in any stratum (checked directly), so its count is unaffected.

WikiDBs: real copies in the wild, at the Clopper–Pearson bar. 400 WikiDBs databases were scanned locally (81 skipped: pathologically wide shared-property tables producing >200 copy declarations, a stated compute cap, not a leakage judgement). The setup uses the shared-Wikidata-property C1 proxy, the same four strata, and zero self-pairs. Of 553 resolved pairs: **103 structural copies**, **79 under the Clopper–Pearson bar at a purity threshold of** 0.5 (41, 28 and 10 across three strata; 40 at a threshold of 0.8), the bar being a one-sided 95% lower bound throughout, and

21 under the strict bar. The strict bar under-counts here specifically because its eligible denominator excludes the SMALL_TABLE stratum entirely (0 of 243 pairs can ever reach 100 groups there, by construction). The Clopper–Pearson bar, by contrast, recovers real, high-confidence findings from exactly that stratum. A near-unique-key guard was applied directly: for all 21 strict-confirmed pairs, distinct feature values divided by rows is 0.46–0.78, well short of near-unique ($>$0.9), so these are genuine many-to-one determinations (many specific entities, few classes), not singleton-group artifacts. Composition: 7 of 21 are gene/protein-ontology attributes (biologically constant by definition, e.g. `encoded_protein` $\rightarrow$ `gene_subclass`, 103–696 groups, 91–100% pure). The rest are mostly geographic redundancy (`capital_city` $\rightarrow$ `time_zone`/`settlement_type` in Ukrainian and Veneto administrative tables) or category-hierarchy redundancy. These are real, confirmed determinations in uncurated data, reported with this domain characterization rather than as 21 independent "bugs."

Coverage and sensitivity (CTU-24 pool). 554/559 tasks (99.1%) have $\geq$1 schema-only closure candidate. 29/559 (5.2%) are rooted at a PK-less table, where a copy declaration can never be proposed for the target at all (Definition 1, (S3)). Of 355 flagged pairs, 45 are unresolved (41 via $\geq$2 FK hops, 4 via two direct FKs to the same parent, genuinely ambiguous). Sensitivity at the originally specified grid (repeated groups $\in \{100, 30, 10\}\times$ margin $\in \{2, 1, 0.5\}$): the strict-bar count is unchanged in every cell, turning non-zero only once the group floor drops to 6 or below. That robustness is itself a finding about the bar: it is internally consistent across the grid, and the positive control shows it is not calibrated for this population.

Outcome change. In the trained CTU-24 sample the confirmation step removed nothing under the strict bar, so FULL and CLEAN are identical for all 138 trained tasks by construction (gradient boosting and logistic regression, 5 seeds). Inflation is exactly 0.0 points [0.0, 0.0] for both families. The confirmation step found nothing removable in that sample under that bar, and inflation is 0 because nothing was removed. Where the confirmation step does fire, the effect is large: the positive control's margins run 26–46 points and WikiDBs supplies 79 scaled-bar findings. A model-impact rerun on the WikiDBs findings appears in Appendix U and confirms the mechanism at a modest but real scale, consistent with the positive control.

population (self-pairs excluded)	structural	CP $\geq$ 0.5	CP $\geq$ 0.8	strict
control: exact copy (n=20, known leak)	**20**	8	6	**0**
control: recoded copy (n=20, known leak)	**20**	8	6	**0**
control: 90%-noisy copy (n=20, known non-copy)	**0**	0	0	0
CTU-24 pool (355 flagged, 310 resolved)	15	1	–	0
CTU-74, high-cardinality stratum (64 resolved)	1	1	–	1
WikiDBs, all 4 strata (553 flagged/resolved)	103	**79**	40	21

Table 22: Four evidence bars, validated against known plants before being applied to uncurated pools. Structural = purity 1.0 on at least one repeated group; CP = one-sided 95% Clopper–Pearson lower bound on purity, thresholded at the stated purity level, with the same margin/CI gate; strict = $\geq$95% constant on $\geq$100 repeated groups. The structural bar recovers every exact and recoded plant and rejects every noisy one; the strict rule recovers none of them, so a zero count under it is uninformative about the data. Dashes: not computed for the CTU strata.

U WikiDBs: from confirmed determination to measured model impact

Appendix T confirms 21 structural copies at the strict bar and 79 at the Clopper–Pearson bar, the 21 a subset of the 79. Those determinations describe the data. This appendix measures what a model trained on that data actually loses, over a task set deduplicated at the level of root tables so that no finding is counted twice.

Task construction. For each confirmed (root table, target, feature) triple, the feature view is built exactly as §2 specifies: the root table's other columns, plus every column of every table reached by one declared foreign-key hop, keys excluded. All 100 confirmed rows are root-table-only determinations, with feature and target both raw columns of the same table. The confirmed feature

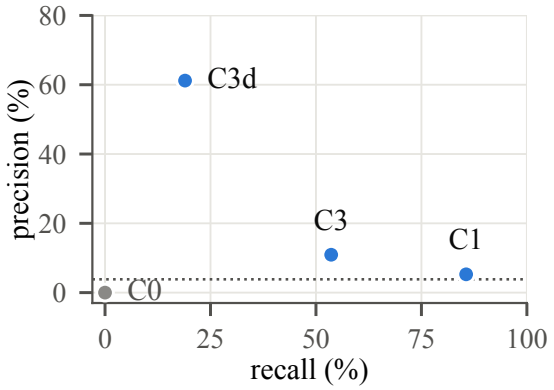


Figure 9: WikiDBs: precision and recall of Algorithm 1 over all 9.8M (FK column, target) units (%, 95% CI; dotted: base rate), by copy source: C0 keys and FKs; C1 shared property; C3 adds a learned ownership list; C3d is C3 kept inside Φ_0.

is very often itself an FK-source column of the root table. The confirmed target is a different root column whose value is duplicated at the FK's destination under a shared Wikidata property (for example, a gene table's `host_taxon` duplicates the protein table's `organism_taxon`, reached via the `encoded_protein` FK). Ablation removes the named feature column and skips the one-hop join for any FK keyed solely on it, so the duplicated value cannot re-enter one hop away under a different name.

Model and measurement. A histogram-based gradient-boosted classifier (categorical-aware, 150 iterations) is trained twice per task, with and without the confirmed feature, under 5 seeds and a fixed paired 70/30 split. It reports accuracy and macro-F1 with 95% bootstrap intervals (2000 resamples), pooled over all seed-by-test-row pairs, next to the majority-class baseline. Six of the 79 scaled-bar cases (33–49 usable rows) were dropped as too small for a meaningful split. None of the 21 strict cases were dropped, leaving 73 trainable tasks.

Duplicate schema detection. WikiDBs sometimes generates more than one database from the same topic, and a single database can itself contain more than one table describing the same entity population. Both patterns appear among the 73 trained tasks. `00725 PlasmodiumKnowlesiOrthologsDb1` is a re-export of `00060 PlasmodiumKnowlesiOrthologsDb` under new table and column names, with identical Wikidata gene identifiers in the same row order (227/227 rows verified by hand). `01034 PLASMODIUM_MALARIAE_ORTHOLOGS_DB_2` contains two internally duplicate tables (`GENE_ORTHOLOG_ANNOTATIONS` and `ENCODED_BY`) over the same gene population. Both were caught by clustering the 29 distinct root tables on the Jaccard overlap of their declared-key (Wikidata item id) value sets, matching structure and content rather than the database title, which differs across clones. Two clusters emerge, of 4 and 2 roots. No other pair among the 29 roots exceeds Jaccard 0.5, so similarly templated but biologically distinct ortholog databases are correctly not merged. Grouped further by target name, this yields exactly 4 task-groups: one 4-way clone (`ChromosomeId`, all four tasks at $\Delta\text{accuracy} = 0.6927536\ldots$ to seven digits) and three 2-way clones (`GENE_TYPE`, `GENE_SUBCLASS`, `CHROMOSOME_ID`, the latter pair both at $\Delta\text{accuracy} = 0.459406$). These 6 tasks are not independent evidence. They collapse to 2 additional representatives once the largest cluster's own duplication is accounted for, taking 73 tasks to 67. None of the 21 strict-confirmed tasks belong to either cluster.

Result. The distribution, not the maximum, is the finding. Table 23 reports it both as measured and after clone deduplication. The deduplicated column is the prevalence claim, since the as-measured tail counts the same underlying finding more than once. On the 67 deduplicated tasks, a confirmed determination moves a trained model's accuracy by a median of about one point, with the interquartile range running from 0 to about 4 points. Between a third and half of the 67 tasks show no measurable movement. A real tail remains: 13 of 67 tasks (19%) lose more than ten accuracy

points, and 16 of 67 (24%) have a 95% bootstrap interval that excludes zero. The maximum is 69 points. Deduplication leaves the median unchanged and narrows this tail from the as-measured 23% and 27% (17/73 and 20/73) to the deduplicated 19% and 24% (13/67 and 16/67), a small narrowing because only two clone clusters exist among 29 distinct root tables. The strict-confirmed set (21 tasks) contains no clones and is identical before and after deduplication: 29% lose more than ten points and 38% have a 95% interval excluding zero.

Detector behaviour. Across the same tasks, the closure detector's behaviour is a per-task property and is unaffected by duplication. Under keys and foreign keys alone it is silent on every task (ABSTAIN, 67/67 deduplicated and 73/73 as measured), flagging the confirmed feature on none of them. Once the schema-only shared-property copy declaration is supplied, it certifies every one, flagging exactly the confirmed feature (LEAK, 67/67 and 73/73).

These are ordinary public databases carrying ordinary Wikipedia-derived redundancy, so the result measures what a screen would catch, not a defect in the benchmark.

Full per-task numbers, clone-group detail, per-seed raw fits and detector verdicts are mapped to their source files in Appendix Z.

	strict ($n = 21$)	scaled-bar, as-measured ($n = 73$)	scaled-bar, dedup ($n = 67$)
median Δaccuracy (IQR)	0.010 (0.000–0.101)	0.010 (0.000–0.061)	0.010 (0.000–0.042)
median Δmacro-F1 (IQR)	0.018 (0.000–0.163)	0.018 (0.000–0.089)	0.014 (0.000–0.077)
max Δaccuracy / Δmacro-F1	0.235 / 0.567	0.693 / 0.933	0.693 / 0.933
tasks losing $>$10 accuracy points	6/21 (29%)	17/73 (23%)	13/67 (19%)
tasks with 95% CI excluding zero	8/21 (38%)	20/73 (27%)	16/67 (24%)

Table 23: Model impact of removing a confirmed WikiDBs structural-copy feature (Δ = full-feature minus ablated, pooled 5-seed test predictions, 95% bootstrap CI). Dedup keeps one representative per clone group (Jaccard ≥ 0.8 on root-table key values). Two clusters collapse 6 tasks to 2 representatives. The 21 strict-confirmed tasks contain no clones, so their column is unaffected.

U.1 A DECLARATION FITTED ELSEWHERE, APPLIED TO HELD-OUT SCHEMAS

The measurements above concern determinations confirmed by reading rows. This subsection measures the same quantity for determinations selected without reading any, under a rule fitted on databases other than those it is applied to. The C3d ownership list of §5.3 is cross-fit: a Wikidata property counts as owned when its precision on the training half of WikiDBs is at least 0.5 over at least 20 flags. We take rep 0, fold 0, one of the 20 cross-fit folds (38 owned properties, 2,511 training and 2,511 held-out databases), and apply the resulting declaration to held-out databases only, reading no rows.

Sampling. Held-out databases are scanned in the fold's own order and the first flagged unit of each is kept, so no database contributes more than once. The first 500 such units were written to disk before any model was trained and were not re-drawn, filtered or reordered afterwards; the data-true label, whether the copy holds in the rows, was recorded but took no part in selection. Of the 500, 384 are not trainable under this appendix's own floor: 337 root tables carry fewer than 50 usable rows and 47 targets have a single class. Two trained on fewer than 5 seeds and are excluded. That leaves 114 tasks over 114 distinct root tables. Appendix U's clone test applies here unchanged: two root pairs exceed Jaccard 0.8 on their declared-key value sets, but no two clone-paired tasks share a target, so the grouping rule collapses nothing and the counts below are already per distinct finding.

Protocol. As above: the same one-hop feature view, a histogram-based gradient-boosted classifier, 5 seeds, a fixed paired 70/30 split, accuracy with 95% bootstrap intervals pooled over all seed-by-test-row pairs, and the majority-class baseline beside every number. Bootstrap seeds are pinned. Each task is run against two controls. The first removes an arbitrary non-flagged column of the same root table. That control is a weaker intervention than the treatment rather than a matched one: all 114 flagged columns are foreign-key sources, so removing one also drops its one-hop join, while only 3 of the 114 arbitrary columns are, and the treatment removes a median of 47 feature columns against the control's 1. The second control corrects this, removing a *foreign-key source* column of

the same table, median 20 columns. It exists for 87 of the 114 tasks; for the other 27 the flagged column is the table's only foreign-key source and no matched control can be built.

Result. Against the matched control the flagged column is not distinguishable. On the 87 tasks where one exists, removing the flagged column and removing an arbitrary foreign-key source column move the model equally: 19 tasks up, 49 tied, 19 down, Wilcoxon $p = 0.51$ (medians 0.00 both; 7 against 6 intervals excluding zero under the seed-level construction below, 11 against 6 pooled; 3 against 2 above ten accuracy points; maxima $+26.7$ and $+13.1$). Both exceed the arbitrary-column control, which moves 0 of 87 above ten points, at most $+2.9$. The large losses concentrate in the 27 tasks with no matched control, where removing the flagged column removes the table's only one-hop join: 6 of 27 exceed ten points, to a maximum of 50.6.

Interval construction and multiplicity. The intervals above pool seeds with test rows, which treats five correlated runs on a fixed split as independent. Recomputing each task's interval as a cluster bootstrap over the five per-seed deltas instead moves the counts only slightly: 18 of the 114 flagged units exclude zero rather than 21, the comparison column 1 rather than 2, and the matched column 6 either way. At $N = 114$ and a nominal 95% level chance alone would give 5.7, so 18 is far beyond it ($P(X \geq 18 \mid \mathrm{Bin}(114, 0.05)) = 1.5 \cdot 10^{-5}$; the pooled 21 gives $2.2 \cdot 10^{-7}$). The comparison column's single firing is what a null should look like ($P = 0.997$). The flagged set therefore contains real effects well beyond chance, which is consistent with, and not a retraction of, the matched-control result above: those effects are not specific to the columns the rule flags.

Provenance of the frozen sample. The 150-unit sample was written at 14:55:46 with SHA-256 beginning `e0898063`, and the first result file for it at 15:14:13. The 500-unit extension was written at 15:25:07 with SHA-256 beginning `6eedd9cd`, its first 150 rows byte-identical to the original, and its result files at 16:05:20 and 16:16:16. Every result file therefore postdates the sample that defines its units. Reaching 500 units required scanning 1,053 held-out databases. The fold population is 5,022 rather than the corpus's 5,094 because 72 databases contribute no evaluable authority unit.

Multiplicity and resolution. At $N = 87$ chance alone gives 4.35 intervals excluding zero: pooled, the flagged column gives 11 ($P = 0.004$), the matched column 6 ($P = 0.27$), the comparison column 2 ($P = 0.94$); over the five per-seed deltas the three are 7, 6 and 1 by cluster bootstrap ($P = 0.15$, 0.27, 1.00) and 4, 3 and 0 by a Student interval on four degrees of freedom, which over the 114 gives 15 flagged ($P = 5.8 \times 10^{-4}$) and 0 comparison. Both constructions are reported, the seed-level one primary: under it the flagged column's excess over chance on the matched subset is not resolved, and the flagged and matched counts stay within one task of each other. With $B = 2{,}000$ resamples the finest one-sided probability a task can attain, $1/(B+1) = 5.0 \times 10^{-4}$, exceeds the Bonferroni level $0.05/114 = 4.4 \times 10^{-4}$: no family-wise claim over the 114 is resolvable at this resampling budget, and at $N = 87$, level 5.7×10^{-4}, only a task whose every resample falls on one side reaches it.

A null for the rule itself. A sham list of the same size, 38 properties drawn uniformly from the 172 carrying at least 20 training-half flags and sharing 4 with the real list, applied schema-only to the same held-out databases in the same scan order through the same code path, yields 500 units from 931 databases, 140 of them trainable. Its ablated arm excludes zero on 14 of 140 intervals (7.00 expected, $P = 0.011$; 9 by Student interval, $P = 0.27$), moves 3 of 140 above ten accuracy points and reaches $+25.9$. The two delta distributions are not separated (Mann-Whitney $p = 0.82$), nor the interval rates (21 of 114 against 14 of 140, Fisher $p = 0.067$; seed-level $p = 0.085$); the ten-point tail rates differ nominally (9 of 114 against 3 of 140, $p = 0.039$, uncorrected). A random property set of the same size selects columns whose removal costs a model at the rate the ownership list does.

Scope. The experiment shows that a rule fitted on other databases and applied to schemas it has never seen selects columns whose removal costs a model, and it does not show that the rule selects them better than removing any foreign-key source would, nor better than a random property list of the same size. What the flagged columns and the matched controls share is structural: each carries a one-hop join. The ownership list is fitted on data from other databases, so the declaration is schema-only at application and not in origin. We report this as a negative result for schema-blind selection

on this corpus, not as evidence that the certified determinations of §5.1 are structural artefacts: those are certified from declarations, verified on rows, and measured against permutation controls.

V THE JOIN: A 639-DATABASE SCHEMA CENSUS

THE JOIN ships 639 relational databases in RelBench manifest format, the largest public corpus of declared relational schemas we are aware of. We read the 639 manifests and the footers of all 3,842 parquet tables, reading no data rows.

What the format can express. The corpus declares 2,766 primary keys over 3,842 tables, 3,773 foreign keys, no foreign key referencing a non-key and no dangling foreign key. It declares *zero* `UNIQUE` constraints and *zero* copy relationships, because the manifest format has no field for either. Membership in Φ_0 is therefore decided by primary keys alone: 35.0% of databases raw, 28.8% after deduplicating by schema signature, and 27.6% macro-averaged over 63 domains.

Verdicts. Over the 10,612 autocomplete tasks and their 17,110 candidate target columns, C0 certifies 0 leaks and C2 certifies 0. C1 flags 877 columns across 208 tasks, falling to 512 columns and 137 tasks after deduplication. The flags concentrate sharply: one database supplies 18% of them, and the 73 Stack Exchange databases contribute 365 flagged columns that collapse to *five* distinct (table, column) findings repeated across sites, which is why we report deduplicated counts throughout. A further 6,023 tasks are forecast tasks whose target column exists in no table at all, placing 36% of a shipped task suite outside the class by construction.

Overlap with CTU. The corpus card excludes databases overlapping RelBench and dbinfer but not CTU, and 113 of the 639 are CTU re-imports, 105 with identical table sets. These include every CTU database from which Appendix X reports a finding. THE JOIN therefore enlarges the C0 null to the largest population we have tested, but it is not an independent replication of our CTU results, and it is not an independent replication.

W FUZZY-NAME CANDIDATES, AND WHAT BETTER PROPOSAL BUYS

C1 requires two columns to carry the same name. That is why it cannot propose the decisive case, where the child column is `ITEMINCOTERMSCLASSIFICATION` and the parent is `HEADERINCOTERMSCLASSIFICATION`. We add **C1f**, which keeps C1's foreign-key and type conditions and replaces name equality with a dictionary-free character-trigram Jaccard at threshold τ. C1f is a strict superset of C1.

The threshold is calibrated off the evaluation corpora. $\tau = 0.10$ is the largest two-decimal value admitting at least 90% of the 130 *renamed* copies that ERPNext and Odoo declare in their own source (Appendix P), a population independent of every benchmark we score on. It was fixed before any result was computed. The calibration is also informative: real declared renames share far less of their spelling than one would guess, the median about a third of its trigrams and three of the 130 none at all.

C1f finds copies the exact rule cannot express. On the harness and stratum we use for C1 (SchemaPile, `all_but_echo`, `keys_unique`), the dependency behind a C1f leak holds non-trivially for 6.7% of 2,713 pairs against 4.6% of 2,394 matched controls. That is $+2.12$pp, 95% interval $[0.48, 3.79]$, $p = 0.005$. C1, rerun on the same sample alongside C1f, shows nothing (4.41% against 4.56%, -0.16pp, $p = 0.573$; the main text quotes the original C1 run, 4.5% against 5.2%). The gain is entirely in the newly reachable pairs: C1f leaks that C1 also proposes reproduce C1's null (4.20% vs 4.60%, $p = 0.625$), while C1f-only leaks run 8.17% vs 4.53% ($p < 0.001$). On RelBench v2's task targets C1 proposes none of the designer-removed columns and C1f proposes two, both removed, including the decisive item-incoterms pair.

What it costs. C1f flags 85,776 of SchemaPile's 1,137,685 targets against C1's 40,524, and declares 252,831 copies against 63,016. The structural cost is larger than the flag count suggests. (S3)

violations rise from 21,328 to 168,183, almost all of them a target given several copy sources at once. Membership in Φ_0 falls from 57.4% to 50.4% on SchemaPile, 57.1% to 40.3% on CTU, and 37.7% to 6.7% on a 300-database WikiDBs sample. On much of the corpus the permissive guess does not merely lose precision; it stops being a well-formed declaration set. Verified in rows on RelBench, 12 of 97 C1f copies hold against 6 of 10 for C1.

Reading. A better candidate generator is worth having, and our C1 understated what name evidence can propose. It does not close the declaration gap. On WikiDBs the three proposers trade off monotonically: a name-and-type gate (C1n) reaches 12.99% precision at 46.5% recall, the shared-property gate (C1) 5.41% at 85.1% on this sample (5.3% at 85.6% on the full set, §5.3), and C1f 3.97% at 99.0%, flagging 591,173 units to reach 23,476 true ones. Loosening the proposer buys recall and costs precision, and none of the three buys authority. That is what Theorem 3 prices.

W.1 Inclusion-dependency discovery and schema matching as candidate generators

A cross-table copy is a unary inclusion dependency, so dedicated IND discovery and instance-based schema matching are the natural competitors to our candidate generation, and are cheaper than the FD profilers of Table 6.

The family has a structural ceiling here. Of the determinations among RelBench v2's removed columns, exactly one is cross-table. The rest, among them `results.positionOrder`, `salesdocumentitem.SHIPPINGPOINT` and `eligibilities.maximum_age`, are same-row equalities within one table. IND discovery and schema matching quantify over column pairs in *different* tables, so six of the seven lie outside what they can express, at any threshold.

Where it can look, containment is not evidence. An exact unary IND sweep over all 4,748,942 rel-salt rows (8.9 s) tests 930 column pairs and returns 127 exact inclusion dependencies, 84 of them cross-table; 17 are declarable as copies and 1 is true, a precision of 0.059. The reason is visible in the output, since `salesdocumentitem.PLANT` $\subseteq$ `customer.CUSTOMER` holds with containment 1.0 because every descriptive column is a small-domain integer code. Valentine's `DistributionBased` ranks a false pair above the true item-incoterms pair, and a containment scorer ties them; what separates the true pair cleanly is name evidence (`Coma` $28.9\times$, a dictionary-free trigram match $8.4\times$ reading no rows), which is what motivates C1f (Appendix W).

The pipeline runs end to end, and still needs verification. Matcher-proposed copies passed through Algorithm 1 reproduce our item-incoterms verdict exactly, LEAK by primary-key lookup, with no hand-written declaration, where C0 and C1 abstain. But 69 of the 70 proposed copies are false in the rows, so six of seven scorers would yield sound-but-false certificates. Matching supplies candidates; Theorem 3 supplies authority. On cost, the matchers read fewer rows than FD discovery (1,832 against 20,097 per task) and take longer (13–4,404 ms against 151 ms).

One incidental finding. `salesdocumentitem.CREATIONTIMESTAMP` equals `salesdocument.CREATIONTIMESTAMP` on 100.0000% of all 2,319,540 items and appears in no RelBench removal list. No rel-salt target is plausibly determined by it, so we report it not as a leak but as an undeclared exact copy shipped in a public benchmark.

W.2 The rel-salt determination matrix

rel-salt poses eight autocomplete tasks over the same eight business attributes, and each task's manifest removes the other seven (§5). That is a complete grid of 56 ordered (target, feature) pairs, every one of which the benchmark treats as a leaker. We ran both of our instruments over the grid: the closure on the schema alone, and the re-insertion protocol of Table 2 on the data.

Control. The four cells already published in Table 2 were re-measured first and reproduce to five decimals (item-incoterms $+0.28038$, sales-incoterms $+0.22448$, item-shippoint $+0.00164$, item-plant -0.00135). The remaining 52 were then run: 325 fits, 0 failures.

Closure, no rows read. Over all three copy sources the closure certifies a forced determination in 0 of 56 cells under C0, 0 of 56 under C1, and 1 of 56 once the ownership declaration of §5 is supplied, which is an input to the run, not an output of it. Of the 56 cells, 15 are not in the feature view at all: foreign keys run from item to header, so a header-rooted task never reaches its items' columns. All 168 verdicts took 14.7 ms in total, a median of 0.04 ms per cell-arm, reading zero rows.

Measured impact. Eight cells move held-out accuracy by at least 2 percentage points. Each is sign-consistent across all five seeds, with an across-seed standard deviation between 0.045 and 0.333pp: item-incoterms ← HEADERINCOTERMS +28.04, sales-incoterms ← ITEMINCOTERMS +22.45, sales-shipcond ← HEADERINCOTERMS +5.80 and ← ITEMINCOTERMS +5.77, item-incoterms ← SHIPPINGCONDITION +5.46, sales-group ← CUSTOMERPAYMENTTERMS +4.96, sales-payterms ← SALESGROUP +4.82, and sales-incoterms ← SHIPPINGCONDITION +4.61. The remaining 48 stay inside ±2pp on every seed; 28 of them change sign across seeds, which at a median seed-to-seed span of 0.18pp reflects an effect indistinguishable from zero rather than an unstable measurement. The sales-group ← CUSTOMERPAYMENTTERMS effect is the mirror of sales-payterms ← SALESGROUP, and was predicted from that symmetry before measurement.

What the two instruments say together. The eight leaks divide by *why* a schema cannot reach them. One is a functional determination, holding on 0.998 of rows, and the closure certifies it. Two are one-to-many aggregates from item to header, which Proposition 2 proves no copy-closed view can express, so the abstention is forced. The other five have dependency strengths between 0.047 and 0.31: they are statistical couplings, not determinations, and certifying them would be unsound. The closure certifies no non-leaking cell anywhere in the grid.

X A VALUE-FIRST AUDIT OF THE CTU AND RELBENCH-V2 POOLS

This appendix reports an independent search of the same public pools used by our other audits (Appendices R and T), conducted *without* importing or replicating the closure detector's candidate grammar. The goal is to check whether those audits' agreeing near-zero counts reflect the data or the shared detector's narrowness.

Method. For every task in the 559-task CTU pool of Appendix T (22 databases, cached locally), we built the candidate universe directly from declared foreign/primary keys (used only to locate joins, never to filter candidates). The universe covers same-table pairs, depth-1/2 many-to-one joins, and one-to-many aggregates (mode/mean of a child column grouped onto its parent), the shape a declared-copy closure provably cannot reach. Every candidate was scored with the evidence functions of Appendix R, copied verbatim: a 50/50 held-out majority-lookup margin over the majority baseline, its 1000-resample row-level bootstrap CI, and an exact functional-dependency check on repeated feature-value groups. A pair DETERMINES iff the margin is at least 2 percentage points with a CI excluding zero *and* the dependency is exact ($\geq 95\%$ constant) on at least 100 repeated groups. A second, more permissive *structural* count records whether the dependency holds on 100% of whatever repeated groups are actually observed, with no floor on their number, the same distinction Appendix T's positive control forced on us. A fast, locally-authored crosstab-based prescreen (never used for a reported number) pruned 74,513 candidate checks to the 8,309 promoted to the full, verbatim test.

Results. 108 of 8,309 promoted candidates DETERMINE, collapsing to 10 distinct findings after grouping near-duplicate feature columns (e.g. dozens of correlated box-score columns feeding one underlying pattern). This was cross-checked against the declared-schema closure (read only for this check, never executed as part of the search). Every one of the 10 is a genuine dependency in the data, with no declared foreign key, row equality, or value map that implies it. Named mechanisms are as follows. (i) A same-row value-map / threshold derivation, verified independently against the raw values (a clinical risk category bucketized from a continuous lab value). (ii) Same-entity repeated-row constancy, where a column keyed by a declared foreign/primary key is constant across the multiple rows that share it, a cross-row invariant the declared key does not itself imply. (iii) Undeclared same-table denormalization, where two independently named columns are both attributes

of one real-world entity with no normalized dimension table in the schema. (iv) A distributional confound, where an unmodeled discrete factor (a league/era boundary) happens to separate the value ranges of an otherwise unrelated numeric aggregate. None of the 108 pairs overlaps with the 355 (feature, target) pairs the schema-only screen ever flagged for this same pool. Nor do a further 12 pairs that *are* plain declared foreign-key copies of a low-cardinality attribute (100% exact on every repeated group observed, but only 2–6 such groups). Fifteen of the pool's 22 databases contributed zero schema-flagged candidates to that screen at all. One of them (CDESchools, 26 tasks) alone supplies four of our ten findings. A structural count across all 8,309 promoted candidates finds 1,404 with a perfect dependency on the repeated groups observed, but 1,023 of those have fewer than ten such groups. The 100-group floor is frequently unreachable because the feature's own cardinality is too low, independent of data volume, exactly as the positive control of Appendix T showed. Only 57 candidates ever reach the floor. 45 also clear the margin bar, and the remaining 12 have a perfect dependency on a near-constant target whose margin cannot clear 2 points by construction.

Scope and negative results. A parallel, smaller stratum on one RelBench v2 task (rel-ratebeer's official `beer_ratings-total_score` autocomplete target, sampled 200,000 rows, 34 same-table and depth-1 candidates excluding the task's own already-documented removed sub-scores) found zero determinations. This corroborates the independent, considerably deeper value-level audit of that benchmark family reported in Appendix G. 4DBInfer's joined and aggregate candidate space was not re-examined here, since Appendix R's own third pass already applies this identical, closure-independent evidence bar to that corpus (39 checks, 0 determinations). One-to-many aggregates on RelBench v2 remain untested by any audit to date and are flagged as an open scope gap, not a finding either way. None of the CTU findings reported here concerns an official, curated benchmark target or a model's declared input at prediction time. They are value-level dependencies present in a mechanically generated task pool that a declared-schema grammar cannot express, not a claim that any specific public benchmark leaks through its shipped model input.

Finding	Margin (pp)	Baseline	Exact FD (groups)	Certifiable?	Mechanism
Atherosclerosis Contr.HODNSK ∼ ICO	53.4	0.466	100% (850)	No	same-entity repeated row
PremierLeague Actions.Team1 ∼ PlayerID	94.4	0.045	95.7% (506)	No	same-entity repeated row
Atherosclerosis Entry.CHOLRISK ∼ CHLST	22.9	0.771	100% (171)	No	value map (threshold)
CDESchools schools.DOC ∼ District	40.9	0.591	99.8% (1274)	No	undeclared denormalization
Basketball_men draft.lgID ∼ teams.min	15.3	0.847	100% (120)	No	era/league confound
Basketball_men draft.lgID ∼ teams.lgID	15.1	0.849	100% (2)	**Yes**	declared FK copy, below the floor

Table 24: Value-first CTU findings that clear the bar of Appendix R (margin $\geq$ 2pp, CI excludes 0, exact FD on ≥ 100 repeated groups) but were never proposed by the closure's declared-schema candidate grammar, plus one certified copy rejected only by the group-count floor. Baseline = majority-class accuracy on the same held-out split.

Y EVERY EVIDENCE THRESHOLD USED IN THE PAPER

Several appendices each fix a numeric bar for treating an association as determining, a candidate as a clone, or a population restriction as acceptable. Table 25 collects every such threshold in one place. Column 4 cites our test of that value where one exists and states *not tested* where none does, rather than supplying one after the fact.

Table 25: Every evidence threshold used anywhere in the paper. "Tested" cites our check of that value; "not tested" means no such check is reported.

threshold (value)	where used	why that value	sensitivity
$\geq$100 repeated groups (strict/eligibility floor)	Strict rule, App. R, App. X, Table 22	Uniform floor for treating a functional dependency as exact, applied identically everywhere so no case gets a bespoke bar	Tested: grid of group floor $\{100, 30, 10\}$ by margin $\{2, 1, 0.5\}$ unchanged until the floor drops to 6 or below; planted control recovers 0 of 20 known exact copies (App. T)
$\geq$2-point margin, 95% bootstrap CI excludes 0	Same strict rule, same locations	Companion significance criterion, so a nominal majority-lookup margin is not counted as evidence	Tested on the same grid (App. T)

continued on next page

(evidence thresholds, continued)

threshold (value)	where used	why that value	sensitivity
$\geq$95% constant on repeated groups ("exact")	Strict rule; "det." criterion of Table 1 (held-out lookup ≥ 0.95 or exact FD)	Sets how close to a perfect dependency counts as exact rather than a strong but non-deterministic association	Not tested: the grid varied the group floor and the margin but held this cutoff fixed
Structural purity $= 100\%$, no group-count floor	Table 22, App. X	Used because the positive control shows the strict floor cannot recover a known copy at low target cardinality	Validated on the planted control (recovers all 20 exact and recoded plants, rejects all 20 noisy); not swept, since 100% is the ceiling
Clopper–Pearson-scaled purity ≥ 0.5	Table 22	One of two confidence levels reported side by side rather than one chosen silently; a one-sided 95%-confidence lower bound thresholded so pairs with few repeated groups can still be accepted	Measured on the same planted control alongside the 0.8 level (recovers 8 of 20 exact and recoded, rejects all 20 noisy); not swept beyond the two reported levels
Clopper–Pearson-scaled purity ≥ 0.8	Table 22	Stricter alternative to the 0.5 level, reported side by side	Same control (6 of 20 recovered, all 20 noisy rejected); not swept beyond the two reported levels
95% confidence ($\delta = 0.05$) declaration-defect certification	Theorem 3's Clopper–Pearson variant, Table 15, App. M	Standard one-sided exact confidence bound on a sampled violation rate, used to certify an accepted declaration's defect	Tested: coverage measured over 200 seeds under the split-sample value-map statistic, 0.985/0.970/0.935, each with a 95% CI at or above the nominal 0.95
Jaccard ≥ 0.5 (duplicate root-table clustering)	App. U, "Duplicate schema detection"	Cleanly separates the two known duplicate clusters from all 29 candidate root tables; no other pair exceeds it	Not tested beyond the observed separation; no sweep across alternative cutoffs
Jaccard ≥ 0.8 (clone-group dedup)	Table 23, App. U	Keeps one representative per clone group so near-duplicate tasks are not double-counted	Not tested
20-most-frequent-class restriction (mag/venue)	Table 21, App. S	The full 349-way model did not converge reliably within the round budget	Not tested; the row is itself marked post-hoc (P) in Table 21
Near-unique guard > 0.9 (distinct/row ratio)	App. T, WikiDBs paragraph	Rules out a determination that is really a near-unique key producing singleton groups	Not swept; all 21 strict-confirmed WikiDBs pairs measured at 0.46–0.78, well under the cutoff
Declared-type match for a C1 copy candidate ("type gate")	App. P (ERPNext/Odoo)	Restricts name-matched candidates further by requiring declared column types to agree, cutting coincidental name matches	Tested: dropping it raises recall to 84%/74% while agreement moves only to 10%/3%
WikiDBs ownership-list frequency ≥ 0.5	App. I	A property joins the schema-only "owned" list if it behaves as a copy source on at least half of a 20-fold cross-fit training split ($\approx$39 properties), then is applied schema-only to the held-out half	Not tested
C1f trigram similarity $\tau = 0.10$	App. W	Largest two-decimal value admitting $\geq 90\%$ of the 130 renamed copies ERPNext and Odoo declare, a population independent of every corpus scored on	Fixed before any result was computed; a post-hoc sweep of τ is reported in App. W and was not adopted
External baseline defaults: probe ≥ 0.95, PPS ≥ 0.5	Table 1	Shipped defaults of the compared value-level baselines, not chosen or tuned here	Not tested (not our parameter)

Z REPRODUCIBILITY MAP

Table 26: Every number in the paper, its script and its output file. Hardware: one laptop-class CPU (Apple M3 Max), at most 4 threads.

number(s)	script	output file
checker counts (App. D)	checker/run_main.py, exhaustive.py, run_extra.py	checker/out/main_totals.json, exhaustive_*.json, extra_seed21_N300.json
agreement with the checker (§4)	closure/test_vs_checker.py	closure/out/test_vs_checker_{11,12,21}_*.json
Table 6a, Table 7a, Table 8, Fig. 2	planted_v2/run_bench.py, timing_frontier.py, ablation_fk_delete.py; closure/run_planted.py; p03b_timing.py	run_v2/results_long.csv, timing_frontier_summary.csv, ablation_fk_delete_summary.csv; closure/out/planted/summary.csv, rows.csv; p03b_timing.json
Table 6b, Table 7b, Table 9	planted_real/run_financial.py, run_hockey.py, run_credit.py, run_relevent.py, run_reltrial.py, run_relavito.py, run_authority.py, aggregate.py, model_impact.py	detector_scores.csv, pass3_summary.csv, pass3_verdicts_by_dataset.csv, model_impact_summary.csv
SchemaPile in §4, Table 10, App. F	closure/run_schemapile.py, analyse_schemapile.py, verify_schemapile.py	closure/out/schemapile_v12_fillpass/run.json, closure/out/schemapile/selectivity.csv, analysis.json, dv_summary.csv
certificates on real schemas	closure/certify_real.py, certify_real_leaks.py	closure/out/certify_real_pass1.json, certify_real_leaks.json
CTU in Table 10, §7, App. J	closure/run_ctu.py, ctu_clean_compare.py; ctu/ctu_performance_impact_full.py, audit_sec53.py, gnn_hetero.py	closure/out/ctu/summary.json, clean_compare.json, cost.csv; ctu/out/impact_all.csv, placebo_blocks.csv, value_detector_eval.json, cost_value_methods.csv, audit_sec53.json, gnn/*.json
Fig. 9; WikiDBs	closure/run_wikidbs.py, run_wikidbs_c3.py, run_wikidbs_c3d.py; scripts/make_figures.py	closure/out/wikidbs_full_v12/authority.csv, run.json
Table 1; injection sweep	closure/run_relbench.py; relbench/x5_baselines.py, x5_summarize.py	closure/out/relbench/task_verdicts.csv, g2g3_summary.csv, injection_sweep.csv; x5/x5_summary.csv, x5_cost_totals.json
Table 2, Fig. 4	x4_impact/x4_run.py, x4_analyse.py, x4_detector.py	x4_impact/x4_summary.csv, x4_results.csv, out/detector_verdicts.csv
Table 12, Fig. 5 (GNN vs. boosting)	gnn/gnn_x4.py, gnn_analyse.py, gbm_x4_contrasts.py	gnn/results/gnn_results.csv, compare_table.csv
Fig. 3 (declaration ablation)	x7_declaration_ablation.py; scripts/make_figures.py	x7/x7_curve_by_seed.csv, x7_run.json
rel-salt 0.989 / 0.961; 3/23; ratebeer	relbench/relsalt_fd.py, value_audit.py	fd/relsalt_table3_repro.csv, value/M3_*.csv, value/ratebeer_sentinel.json
Table 1 by category; the 7 data-true removals	relbench/table4_by_category.py	g2/g2g3_by_removed_column.csv, x5/x5_flags_*.csv, closure/out/relbench/g2g3_by_removed_column.csv
Table 13, Fig. 6 (Deep Feature Synthesis)	workflow_demo/dfs_f1.py, dfs_salt.py, dfs_detector.py, make_deliverables.py	workflow_demo/out/dfs_table.csv, f1_results.json, salt_results.json, detector_flags.json
screening without rows (App. H)	closure/run_schemapile.py; workflow_demo/zero_rows_screen.py	workflow_demo/out/zero_rows_summary.json, zero_rows_table.csv
blind-declaration impact (§5.3, App. U.1)	blind_impact/freeze_sample.py, run_blind_impact.py, analyse.py	blind_impact/out/sample_ext.csv, sample_meta_ext.json, results.csv, results_gap.csv, results_ext.csv
blind declarations, LM detector, Table 11	blind_decl/isolated/run_isolated.py; blind_decl/score_blind.py, score_detector.py	blind_decl/out/iso_blind_v1/, iso_blind_renamed_v1/, iso_lmdet_v1/, iso_lmdet_renamed_v1/, lmdet_baseline_other_targets/
binary targets (Rem. 2)	checker/binary_target_check.py	binary_target_check_seed*_N3000.json
59.8% (S1 census, Table 4)	prior-art census script	census/pk_census_summary.json
Thm. 3, Tables 14, 15, Fig. 7	pac_ownership/run_synth.py, shared_test_all.py, relbench_checks.py, coverage_check.py	pac_ownership/out/synth_summary.csv, empirical_check.csv, relbench_item_incoterms.csv, coverage_200seeds.csv

continued on next page

(reproducibility map, continued)

number(s)	script	output file
Thm. 4, Fig. 8, Cors. 6, 7	separation/sweep_minn.py, sweep_curve.py, kbound_check.py	separation/out/sweep_minn.json, sweep_curve.json, sweep_extra.json, kbound_check.json, propB_check.json
Cor. 1, App. L (1.1 ms at k=200; 48,787 pairs; $\beta \geq 2$ share)	complexity/dag_closure.py, compare_vs_reference.py, blowup_family.py	complexity/out_agreement.json, out_blowup.json, out_schemapile_beta.json
App. O (4 kinds, adapters, 0.30 decl./task)	decl_format/declfmt_core.py, relbench_adapter.py, dbinfer_adapter.py, validator.py, authoring_cost.py	decl_format/declaration_schema.json, authoring_cost.json, rel_salt_verdicts.json, rel_f1_verdicts.json
Table 16, App. P (ERPNext, Odoo, OData)	vendor_decl/read_erpnext.py, read_odoo.py, read_odata.py, run_closure.py	vendor_decl/out/counts.json, compare.json, summary.json, examples.json, relsalt/relsalt_closure.json
Table 20, App. R (135 targets; 39 checks)	bench_audit/audit_4dbinfer.py, data_audit_pass3.py, pac_verifier_check.py	bench_audit/out/closure_verdicts.csv, data_audit_pass3_joined.csv, pac_verifier.csv
Table 21, App. S (seznam, mag, zero candidates)	boundary/seznam_prepay_impact.py, mag_venue_impact.py, salt_header_verifier.py	boundary/out/seznam_prepay_impact_summary.json, mag_venue_impact_summary.json, salt_header_verifier_summary.json
Table 22, App. T (positive control, WikiDBs bars)	curation/plant_control.py, rebar.py, run_models.py	curation/out/positive_control_rebar.csv, rebar_summary.json, wikidbs_confirmation_rebar.csv, coverage_summary.json
Table 24, App. X (108 pairs, 10 findings)	value_first/search_ctu.py, score_candidates.py, ratebeer_stratum.py	value_first/out/ctu_value_first_full.csv, ctu_value_first_summary.json, relbench_ratebeer_totalscore.csv
Table 23, App. U (67-task WikiDBs model impact, clone dedup)	wikidbs_impact/run_impact.py, dedup_clones.py	wikidbs_impact/task_impact.csv, task_impact_dedup.csv, clone_groups.csv, clone_root_pairs.csv, model_results_raw.csv, detector_behaviour.csv, distribution_summary_dedup.json
Tables 17, 18, App. Q (ERPNext dump: 0/9,169; Odoo dump=C0)	visibility/dump_level_erpnext.py	visibility/out/erpnext_dump/fk_counts.json, summary_dump_C0.json, summary_app_C0_recheck.json
CTU schema count (App. J: 145 content dbs, 119/26 FK split, 15+11, 8.5% of 130)	visibility/ctu_fk_coverage.py	visibility/out/ctu_fk_coverage.json
Table 19, App. Q (Financial hosted four times)	visibility/ctu_fk_coverage.py (schema_cache readout)	visibility/out/ctu_financial_four_copies.json
App. V (639 databases, 3,842 tables, 10,612 tasks, 0 leaks under C0/C2, 113 CTU re-imports)	the_join/{fetch.py,census.py,run_join.py}	the_join/out/{summary.json,per_db_census.csv,task_verdicts.csv,ctu_overlap.csv,clone_families.csv}
App. W.2 (56-cell grid, 8 leaks, 48 nulls, 168 closure verdicts in 14.7 ms)	salt_matrix/{matrix_run.py,closure_grid.py}	salt_matrix/out/{salt_matrix_56.csv,headline.json,closure_timing.json,runs_*.jsonl}
App. W (C1f, $\tau = 0.10$ calibration, 6.7% vs 4.6%, (S3) violations, Φ_0 drop)	c1_fuzzy/c1f.py (a same= predicate for closure_detector.make_copies)	c1_fuzzy/out/{schemapile_summary.csv,dv_pooled_ci.csv,dv_c1f_only_ci.csv,tau_calibration.json,relbench/,wikidbs/}
App. W.1 (IND sweep, matcher ranking, end-to-end pipeline, cost)	ind_matching/{exact_ind.py,match_rank.py,e2e.py,cost.py}	ind_matching/out/{exact_ind_relsalt.csv,decisive_margins.csv,e2e_spillover.csv,verification_s0.csv,cost_summary.csv}