Proceedings of the 23rd International Conference on Principles of Knowledge Representation and Reasoning
Main Track

Almost Certain Query Answering over Incomplete Relational and Graph Data

Heng Liu[1] , **Leonid Libkin**[2] , **Eugenia Ternovska**[1]
[1]Simon Fraser University
[2]RelationalAI & University of Edinburgh
liuhengl@sfu.ca, l@libk.in, ter@sfu.ca

Abstract

Computing certain answers is the standard approach when data or knowledge is incomplete. This very natural concept rooted in logical validity suffers from a severe weakness: its generally intractable computational complexity. Consequently, significant effort has been made to find tractable cases, often at the expense of severe restrictions.

Here we explore a different approach, relaxing not the classes of allowed queries but the very strict notion of certainty. Our starting point is that replacing certainty with asymptotic probability 1 overcomes intractability for large classes of queries. This theoretical observation was previously made for relational queries under a simple probabilistic model of uniform distribution and an infinite domain of equally likely values; these are hardly the realities of querying data. We therefore ask whether this phenomenon is robust enough to extend to other, realistic distributions, to be applicable to other data models, and to help with practical query answering. We answer all of these positively. After extending tractability via naïve evaluation to many distributions, we extend the approach to graph data, and then experimentally show that for relational and graph queries from standard TPC and LDBC benchmarks, convergence to high probability query answers is fast and practical.

1 Introduction

Incomplete data is a recurring challenge when knowledge representation techniques are applied to real-world datasets. A classical and widely used formalization represents missing values by *marked nulls*, which are placeholders intended to stand for *existing but unknown values*, as originally advocated in the relational model to capture missing information (Codd 1979). An incomplete instance denotes a set of *possible worlds* (or completions), obtained by *valuations* that replace every null by a domain element (Imieliński and Lipski 1984; Abiteboul, Kanellakis, and Grahne 1991). This setting appears in multiple application areas: in ontology-based data access (OBDA), where queries are answered over incomplete data sources in the presence of an ontology (Bienvenu and Ortiz 2015; Xiao et al. 2018); in RDF/SPARQL, where blank nodes can be used to represent existing but unknown values (Hernández, Gutierrez, and Hogan 2018); and in data exchange and data integration, whose semantics is naturally defined over spaces of admissible target instances (Arenas et al. 2014; Lenzerini 2002).

The principled approach to querying incomplete data is based on *certain answers* (Imieliński and Lipski 1984), which are answers that hold in *every* possible world. This comes at a significant computational cost. Even for fairly modest extensions beyond conjunctive queries (the $\exists, \wedge$-fragment of first-order logic), computing certain answers becomes computationally hard in data complexity, and coNP-hardness results are well established across standard incomplete-database models and closely related settings (Abiteboul, Kanellakis, and Grahne 1991; Calvanese et al. 2013). While much basic research went into finding (often severe) tractability restrictions, applications that require scalable query evaluation need to adopt alternative strategies to the strict certain query answering.

The alternative we explore here is based on *relaxing the definition* of certainty itself from strict to asymptotic (Libkin 2018). The intuition behind it is as follows. Certainty of a query answer means that it remains an answer *for every valuation* producing a possible world, while *almost certainty* means that it is an answer with asymptotic probability 1 if we randomly pick a valuation.

To define precisely what it means to pick a valuation randomly (as there are a countably infinite number of valuations), (Libkin 2018) adopted the approach from the study of 0–1 laws in logic (Fagin 1976; Glebskii et al. 1969). Concretely, if we fix a domain size k and consider finitely many valuations that map nulls into this domain, we can define the probability that a tuple is in the answer, assuming all valuations are equally likely. An answer is then *almost certain* if this probability converges to 1 as k grows. Of course every certain answer is almost certain, but the notion of almost certainty also permits additional answers that become overwhelmingly likely in large domains.

A justification for this approach came in the form of the *0–1 law* for incomplete relational databases (Libkin 2018):

- the asymptotic probability of an answer always exists and equals either 0 or 1;
- tuples for which it equals 1 are precisely those returned by *naïve evaluation* that treats each marked null as a fresh constant and evaluates the query using standard procedures (Imieliński and Lipski 1984).

While every certain answer is returned by naïve evaluation, the latter can produce *false positives*, i.e., answers that fail in some possible worlds. We also know that naïve evaluation coincides with certain answers for queries that do not use explicit negation and unrestricted universal quantification; for more details see (Console et al. 2020).

To illustrate, consider a clinical trial recruitment scenario. We need to populate a control group for a study on a genetic neurodegenerative disorder identified by code G10. To ensure the study's validity, we must select candidates who do *not* have a diagnosis of G10. Depending on the database model, this query is written in SQL or GQL:

```
SELECT p.name
FROM Patient p
WHERE NOT EXISTS (
    SELECT *
FROM Diagnosis d
WHERE
   d.p_id = p.id
   AND d.code =
     'G10')
```

```
MATCH (p:Patient)
WHERE NOT EXISTS {
  MATCH (p)
  -[:HAS_DIAGNOSIS
      ]->
    (d:Diagnosis)
  WHERE d.code =
        'G10'}
RETURN p.name
```

Under naïve evaluation, a patient's name is returned even if they have an incomplete diagnosis record where the disease code is null (e.g., a pending lab result). This answer is not *certain*: the pending result *could* turn out to be the exclusion criterion G10, which would disqualify the patient. However, there are more than 12,000 diagnosis codes in the ICD-10 standard (World Health Organization 2004). Under the assumption that no single code carries a large probability mass (i.e., the distribution's maximum point probability is small), the chance that an unknown code equals G10 is very small. Hence, the naïve evaluation answer is *highly reliable* even when not strictly certain.

While the results of (Libkin 2018) provide a compelling *initial theoretical justification* for using naïve evaluation to avoid the complexity curse of certain answers, several major gaps are left:

- That investigation relied heavily on the *uniform distribution* assumption, which is rarely met in real-world data. Commonly encountered properties such as salaries, grades, heights, enrollments, etc., are almost never uniformly distributed.
- The work applied exclusively to the relational model; it was not clear if and how it would extend to the graph model, which is heavily gaining in popularity due to its use for knowledge graphs.
- Without understanding how quickly the probability of query answers converges to 0 or 1, it is difficult to determine whether asymptotic certainty becomes meaningful at the domain sizes encountered in practice.

Contributions Our goal is to fill these gaps and ultimately confirm the applicability of almost certainty as a way of avoiding the high complexity of certain answers in both relational and graph querying. Specifically, we do the following:

- **Generalize to Arbitrary Distributions** We extend the deep connection between almost certainty and naïve evaluation beyond the restrictive assumption of uniformity. We introduce the condition of vanishing *Maximum Point Probability* (defined as the probability mass of the most frequent value in the domain) tending to zero. We prove that convergence to naïve evaluation holds for *every* distribution satisfying this condition, including Zipfian (with $s \leq 1$) and discretized continuous models (Normal, Exponential), even in *heterogeneous* settings where distinct nulls follow different distributions.
- **Generalize to GQL** We bridge the gap between relational and graph data by defining conditional tables for relational algebra (Section 2.4) and adapting them to property graphs as *Conditional Path Tables* (Section 4.3). We use this technique to prove that pattern matching in GQL behaves similarly to relational queries, and naïve evaluation can be used to answer GQL patterns with a high degree of certainty when property values in graphs can be nulls.
- **Empirical Evaluation** This investigation is split into two. First, we investigate how often naïve evaluation differs from certain answers. This is enabled by the previously established techniques based on conditional (path) tables, as they annotate queries with symbolic Boolean formulas that track dependencies on missing data, allowing us to use state-of-the-art SMT solvers for comparing the fast database evaluation with intractable certain answers.

  We then switch to the convergence rates exhibited by naïve evaluation. Addressing the lack of insight into convergence speeds, we perform experiments using TPC-H and LDBC benchmarks for relational and graph data. We identify scenarios where naïve evaluation converges rapidly to yield $\geq 80\%$ certainty on domains containing 5% nulls, and we characterize the interaction of negation over numerous marked nulls and adversarial data skew where convergence is too slow for practical utility.

In summary, our results show applicability of the notion of almost certain answers as a way of avoiding the high complexity of certain answers in different types of data models.

2 Preliminaries and Background

We briefly review the data model for incomplete information, relational algebra, the semantics of generic queries, and asymptotic certainty. We also introduce Conditional Tables (c-tables), which serve as the analytical framework throughout the paper. Standard notions from database theory can be found in database texts (Abiteboul, Hull, and Vianu 1995; Arenas et al. 2022); probabilistic concepts we use can also be found in standard textbooks, e.g. (Ross 2008).

2.1 Incomplete Data and Certain Answers

We adopt the standard model of incomplete databases with marked nulls (Imieliński and Lipski 1984; van der Meyden 1998). Let $\mathsf{Const} = \{c_1, c_2, \dots\}$ be a countably infinite set of constants and $\mathsf{Null} = \{\bot_1, \bot_2, \dots\}$ be a countably infinite set of marked nulls.

A *database schema* σ is a finite set of relation symbols $\{R_1, \dots, R_l\}$, each with a fixed arity. An *incomplete*

database instance D over σ associates with each k-ary relation symbol $R \in \sigma$ a finite subset R^D of $(\mathsf{Const} \cup \mathsf{Null})^k$. The *(active) domain* of D, denoted $\mathsf{dom}(D)$, consists of all constants and nulls appearing in D. We write $\mathsf{Const}(D)$ for $\mathsf{Const} \cap \mathsf{dom}(D)$ and $\mathsf{Null}(D)$ for $\mathsf{Null} \cap \mathsf{dom}(D)$. A database D is *complete* if $\mathsf{dom}(D) \subseteq \mathsf{Const}$.

The semantics of incomplete databases are defined via *valuations* that map nulls to constants; the resulting complete databases are interpreted as possible worlds under the Closed World Assumption (CWA) (Reiter 1981).

Definition 1 (Valuation and Semantics)**.** *A* valuation *is a partial mapping* $v : \mathsf{Const} \cup \mathsf{Null} \to \mathsf{Const}$ *that is the identity on* Const*, that is,* $v(c) = c$ *for each* $c \in \mathsf{Const}$*. For a database* D*, we let* $\mathcal{V}(D)$ *be the set of all valuations defined on* $\mathsf{Null}(D)$*, and for such a valuation we let* $v(D)$ *be the result of applying* v *to all elements of* $\mathsf{dom}(D)$*, i.e.,* $R^{v(D)} = \{v(\bar{t}) \mid \bar{t} \in R^D\}$ *for each relation symbol* R*.*

Relational Algebra Our queries for relational data are expressed in *relational algebra* (RA) under the unnamed perspective (Abiteboul, Hull, and Vianu 1995), where attributes are identified by their position. RA expressions are built from base relations R using the following operators: selection $\sigma_\theta(Q)$, projection $\pi_Z(Q)$, cross product $Q_1 \times Q_2$, union $Q_1 \cup Q_2$, and difference $Q_1 - Q_2$ (the last two between queries of equal arity). Here θ is a Boolean combination of equality atoms of the form $(\#i = \#j)$ or $(\#i = c)$, where $\#$ denotes a position in a tuple and $c \in \mathsf{Const}$, and Z is a list of positions; for a tuple $\bar{t}$, we write $\bar{t}[Z]$ for the tuple obtained by projecting $\bar{t}$ onto the positions in Z. Intersection and natural join can be expressed with these primitives.

Certain Answers Let Q be an RA query of arity n; even more generally, Q may be any query in a language $\mathcal{L}$ that does not invent new values, so Q maps a database D to a subset of $\mathsf{dom}(D)^n$. We adopt the semantics of *certain answers with nulls* (Lipski 1984).

Definition 2 (Certain Answers with Nulls)**.** *A tuple* $\bar{t} \in \mathsf{dom}(D)^n$ *is a* certain answer *to* Q *on* D*, denoted* $\bar{t} \in \mathit{cert}(Q, D)$*, if every valuation yields the tuple in the query result of the corresponding possible world:*

$$\forall v \in \mathcal{V}(D), \quad v(\bar{t}) \in Q(v(D)).$$

It is common in the literature to restrict the above definitions only to tuples of constants (Imieliński and Lipski 1984); for advantages of the above definition we refer, e.g., to (Console et al. 2020).

2.2 Generic Queries and Naïve Evaluation

Genericity is a formalization of the data independence principle: queries do not depend on particular data stored in a database, except perhaps finitely many constants (Abiteboul, Hull, and Vianu 1995; Arenas et al. 2022).

Definition 3 (C-Generic Query)**.** *Let* $C \subset \mathsf{Const}$ *be finite. A query* Q *is* C-generic *if it is preserved under every isomorphism that fixes* C *pointwise. That is, if* D *is a complete database over the schema and* $f : \mathsf{Const} \to \mathsf{Const}$ *is a bijection such that* $f(c) = c$ *for all* $c \in C$*, then*

$$f(Q(D)) \;=\; Q(f(D)),$$

with f *extended to tuples and relations in the natural way.*

Standard query languages like first-order logic, relational algebra, and Datalog express only C-generic queries, with C being the set of constants explicitly mentioned in the query.

For such queries, one can employ *naïve evaluation*: treating nulls as if they were distinct new constants and evaluating the query directly. We formalize naïve evaluation using C-bijective valuations.

A valuation $v \in \mathcal{V}(D)$ is *C-bijective* (on D) if it is injective on $\mathsf{Null}(D)$ and maps nulls to constants distinct from C and $\mathsf{Const}(D)$, i.e., $v(\mathsf{Null}(D)) \cap (\mathsf{Const}(D) \cup C) = \emptyset$.

Definition 4 (Naïve Evaluation)**.** *Let* Q *be a* C*-generic query. Let* v *be any* C*-bijective valuation. The* naïve evaluation *of* Q *over an incomplete database* D *is:*

$$\mathit{naïve}(Q, D) := v^{-1}\left(Q\big(v(D)\big)\right).$$

In other words, nulls are interpreted as new constants according to v, query Q is evaluated on $v(D)$, and then those new constants are turned back into original nulls by v^{-1}. Due to C-genericity, this definition is independent of the specific choice of v (see Proposition 1 in (Libkin 2018)).

2.3 Background: The Relational 0–1 law

This work builds upon the asymptotic analysis of certain answers established in (Libkin 2018). We briefly recall its main result under the uniform distribution assumption.

Fix an enumeration of constants $\mathsf{Const} = \{c_1, c_2, \dots\}$, and for each $k \in \mathbb{N}$ let $C_k = \{c_1, \dots, c_k\}$. The probability space is defined over the set of valuations that map nulls into C_k, denoted $\mathcal{V}_k(D) := \{v \in \mathcal{V}(D) \mid v(\mathsf{Null}(D)) \subseteq C_k\}$. That is, for each k, we have a sample space $\mathcal{V}_k(D)$ of size $k^{|\mathsf{Null}(D)|}$, and each valuation $v \in \mathcal{V}_k(D)$ is assigned a probability $\mathbb{P}_k^D(v) \in [0, 1]$, called its *probability mass*, with $\sum_{v \in \mathcal{V}_k(D)} \mathbb{P}_k^D(v) = 1$. Under the *uniform model*, each valuation is equally likely, i.e., $\mathbb{P}_k^D(v) = k^{-|\mathsf{Null}(D)|}$ for all $v \in \mathcal{V}_k(D)$. This assumption will be relaxed in Section 3.

Definition 5 (Measure of Certainty)**.** *For each* $k \in \mathbb{N}$*, the probability that a tuple* $\bar{t} \in \mathsf{dom}(D)^n$ *belongs to the result of* Q *over domain* C_k *is the total measure of the valuations that yield the tuple:*

$$\mathbb{P}_k(Q, D, \bar{t}) := \sum_{v \in \mathcal{V}_k(D) \text{ and } v(\bar{t}) \in Q(v(D))} \mathbb{P}_k^D(v). \quad (1)$$

We call $\mathbb{P}_k$ the *certainty measure* and focus on tuples that become "almost certainly" true as the domain grows.

Definition 6 (Almost Certain Answers)**.** *The set of* almost certain answers*, denoted* $\mathit{cert}_\infty(Q, D)$*, consists of the candidate tuples whose probability converges to 1:*

$$\mathit{cert}_\infty(Q, D) := \left\{ \bar{t} \in \mathsf{dom}(D)^n \;\middle|\; \lim_{k \to \infty} \mathbb{P}_k(Q, D, \bar{t}) = 1 \right\}.$$

The following theorem establishes that under the assumption of a uniform distribution over valuations, probabilistic certainty for C-generic queries is dichotomous and converges to the naïve evaluation.

Theorem 1 (Uniform 0–1 law (Libkin 2018)). *Let Q be an arbitrary C-generic query, and D an incomplete relational database. Assume the uniform distribution over all $\mathcal{V}_k(D)$.*

1. *If $\bar{t}$ is a tuple in $\mathsf{dom}(D)^n$, then $\lim_{k\to\infty} \mathbb{P}_k(Q, D, \bar{t})$ exists and is either 0 or 1.*
2. *The set of almost certain answers is exactly the result of naïve evaluation: $\mathit{cert}_\infty(Q, D) = \text{naïve}(Q, D)$.*

2.4 Analytical Framework: Conditional Tables

To extend Theorem 1 to GQL, and to enable some aspects of experimental validation, we use *conditional tables* (c-tables) (Imieliński and Lipski 1984) (see also (Abiteboul, Hull, and Vianu 1995; Greco, Molinaro, and Trubitsyna 2019)) to symbolically represent conditions under which a tuple is returned as a query answer. Throughout this section, we consider Relational Algebra queries.

Definition 7 (Conditional Tables (c-tables)). *A* c-table T *is a finite set of* c-tuples $\langle \bar{t}, \varphi \rangle$*, where $\bar{t}$ is a tuple over* $\mathsf{Const} \cup \mathsf{Null}$ *and φ is a* condition*, i.e., a quantifier-free Boolean formula built from* atoms *of the form* $(a = b)$*,* $(a \neq b)$*,* true*, and* false *(with $a, b \in \mathsf{Const} \cup \mathsf{Null}$) using $\wedge$, $\vee$, and $\neg$. We require that for every tuple $\bar{t}$, there be at most one c-tuple $\langle \bar{t}, \varphi \rangle \in T$.*

A valuation v satisfies *condition φ, denoted $v \models \varphi$, if φ evaluates to true when every null z is replaced by $v(z)$.*

We define the *conditional evaluation* of a query Q, denoted $[\![Q]\!]^{\text{cond}}_D$, as a function that takes an RA query Q and a database D and returns a c-table. The inductive definition is in Table 1, where T_i abbreviates $[\![Q_i]\!]^{\text{cond}}_D$. Table 1 shows how any RA query is compiled into a c-table, producing, for every candidate answer tuple $\bar{t}$, a condition φ that captures exactly when $\bar{t}$ is in the result of Q under a given valuation of the nulls. This symbolic representation will be used both in our theoretical proofs (Section 3) and as the input encoding for the Z3 SMT solver in our experiments (Section 5).

Example 1. *Consider the query $Q = \big(\sigma_{\#2\neq 3}(R_1)\big) - R_2$ evaluated over the relations $R_1 = \{(a, 1), (b, \bot_1)\}$ and $R_2 = \{(b, 2)\}$, where $\bot_1 \in \mathsf{Null}$. The conditional evaluation yields the following c-table:*

$$[\![Q]\!]^{\mathit{cond}}_D = \big\{\langle (a,1), \mathsf{true}\rangle,\ \langle (b, \bot_1), (\bot_1 \neq 3) \wedge (\bot_1 \neq 2)\rangle\big\}$$

Theorem 2 (Correctness of Conditional Evaluation (Imieliński and Lipski 1984)). *For every RA query Q, database D, valuation $v \in \mathcal{V}(D)$, and tuple $\bar{t} \in \mathsf{dom}(D)^n$:*

$$v(\bar{t}) \in Q(v(D)) \iff \exists \langle \bar{t}, \varphi \rangle \in [\![Q]\!]^{\mathit{cond}}_D \text{ such that } v \models \varphi.$$

Application: Proving the 0–1 law Formulas produced by query evaluation over c-tables will also be the key for us to set up the experimental analysis. We now conclude this section by giving a simple explanation of why Theorem 1 is true for RA queries Q, that relies on c-tables.

Fix an RA query Q, a database D, and a tuple $\bar{t} \in \mathsf{dom}(D)^n$. Let $\langle \bar{t}, \varphi \rangle$ be the c-tuple for $\bar{t}$ in $[\![Q]\!]^{\text{cond}}_D$ (if no such c-tuple exists, set $\varphi := \mathsf{false}$). By Theorem 2, $\mathbb{P}_k(Q, D, \bar{t})$ is exactly the probability that a random valuation $v \in \mathcal{V}_k(D)$ satisfies φ. As $k \to \infty$, the probability of v mapping distinct nulls to the same constant, or a null to a constant in $(C \cup \mathsf{Const}(D))$, vanishes as being at most inversely proportional to k under the uniform assumption. Thus, almost all valuations are C-bijective. But for any C-bijective valuation v, the condition φ evaluates to a constant truth value because all equality checks between distinct terms fail and all inequalities succeed. Therefore, $\lim_{k\to\infty} \mathbb{P}_k(Q, D, \bar{t}) = 1$ if φ evaluates to true, and 0 otherwise. Since naïve evaluation is the same as evaluation under a C-bijective valuation, the limit is 1 if and only if $\bar{t} \in \text{naïve}(Q, D)$.

Base relation R**:** $[\![R]\!]^{\text{cond}}_D := \{\langle \bar{t}, \mathsf{true}\rangle \mid \bar{t} \in R^D\}$

Selection $\sigma_\theta(Q_1)$**:** $[\![\sigma_\theta(Q_1)]\!]^{\text{cond}}_D := \{\langle \bar{t},\ \varphi \wedge \theta(\bar{t})\rangle \mid \langle \bar{t}, \varphi\rangle \in T_1\}$

Projection $\pi_Z(Q_1)$**:** $[\![\pi_Z(Q_1)]\!]^{\text{cond}}_D :=$
$$\Big\{\big\langle u,\ \bigvee_{\substack{\langle \bar{t},\varphi\rangle \in T_1 \\ \bar{t}[Z]=u}} \varphi\big\rangle \ \Big|\ u \in \{\bar{t}[Z] \mid \langle \bar{t}, \varphi\rangle \in T_1\}\Big\}$$

Cross Product $Q_1 \times Q_2$**:** $[\![Q_1 \times Q_2]\!]^{\text{cond}}_D :=$
$$\Big\{\langle (\bar{t}_1, \bar{t}_2),\ \varphi_1 \wedge \varphi_2\rangle \mid \langle \bar{t}_1, \varphi_1\rangle \in T_1, \langle \bar{t}_2, \varphi_2\rangle \in T_2\Big\}$$

Union $Q_1 \cup Q_2$**:** $[\![Q_1 \cup Q_2]\!]^{\text{cond}}_D :=$
$$\Big\{\langle \bar{t},\ \varphi_{T_1}(\bar{t}) \vee \varphi_{T_2}(\bar{t})\rangle \ \Big|\ (\exists \psi.\ \langle \bar{t}, \psi\rangle \in T_1) \vee (\exists \psi.\ \langle \bar{t}, \psi\rangle \in T_2)\Big\}$$

Difference $Q_1 - Q_2$**:** $[\![Q_1 - Q_2]\!]^{\text{cond}}_D :=$
$$\Big\{\langle \bar{t},\ \varphi \wedge \Psi_{\bar{t}}\rangle \mid \langle \bar{t}, \varphi\rangle \in T_1\Big\},$$
$$\textit{where}\ \ \Psi_{\bar{t}} := \bigwedge_{\langle \bar{t}', \varphi'\rangle \in T_2} \neg\big(\varphi' \wedge (\bar{t} = \bar{t}')\big).$$

Tuple equality $(\bar{t} = \bar{t}')$ abbreviates the conjunction of componentwise equalities.

Table 1: Conditional evaluation $[\![Q]\!]^{\text{cond}}_D$ for relational algebra. Let $T_i = [\![Q_i]\!]^{\text{cond}}_D$. For a c-table T, define $\varphi_T(\bar{t}) := \varphi$ if $\langle \bar{t}, \varphi\rangle \in T$, and false otherwise.

3 0–1 Law Beyond Uniformity

We now generalize the 0–1 law to arbitrary distributions, and also move beyond the identically distributed assumption, by showing that the 0–1 Law holds even when distinct nulls follow different probability distributions. This result makes one mild assumption that the distribution is not very skewed.

3.1 Probabilistic Framework

We closely follow the setting of Section 2.3. Recall that $\mathcal{C} := \{C_k\}_{k\in\mathbb{N}}$ is a sequence of finite domains with $C_k = \{c_1, \ldots, c_k\}$. Let D be an incomplete database with $\mathsf{Null}(D) = \{\bot_1, \ldots, \bot_m\}$. A *heterogeneous model* of incompleteness is given by a family $\mathcal{P}_k = \{P_k^{(1)}, \ldots, P_k^{(m)}\}$ of independent probability distributions on C_k, for each k. In this model, each null $\bot_j$ is drawn independently from the set C_k according to the distribution $P_k^{(j)}$. The probability mass function $p_k^{(j)}(c) := P_k^{(j)}(\{c\})$, gives the probability that the value $c \in C_k$ is assigned to the null $\bot_j$.

As in Section 2.3, we lift these distributions to a distribution on the space of valuations $\mathcal{V}_k(D)$ that assign values in C_k to nulls in $\mathsf{Null}(D)$. Since each null $\bot_j$ is drawn independently from $P_k^{(j)}$ on C_k, with probability mass function

$p_k^{(j)}$, by independence the joint probability of a valuation v is the product of individual probabilities:

$$\mathbb{P}_k^D(v) \;:=\; \prod_{j=1}^{m} p_k^{(j)}\big(v(\bot_j)\big) \qquad \text{for } v \in \mathcal{V}_k(D). \tag{2}$$

When all $p_k^{(j)}$ are uniform on C_k, Eq. (2) gives $\mathbb{P}_k^D(v) = k^{-|\mathsf{Null}(D)|}$, which is precisely the setting of the uniform model in Section 2.3.

Then again just as in Section 2.3, for a query Q and a tuple $\bar{t}$ we define $\mathbb{P}_k(Q, D, \bar{t})$ by Eq. (1), and set $\mathrm{cert}_\infty(Q, D)$ to be the set of tuples $\bar{t}$ for which $\lim_{k\to\infty} \mathbb{P}_k(Q, D, \bar{t}) = 1$.

To analyze the asymptotic behavior of $\mathbb{P}_k^D$, we define a metric that dictates convergence.

Definition 8 (MPP Metric)**.** *For a domain C_k of size k, we define the* Maximum Point Probability *(MPP) for each null $\bot_j$ as*

$$M_k^{(j)} := \max_{c \in C_k} p_k^{(j)}(c)$$

We say that the heterogeneous model has a vanishing MPP *if $\lim_{k\to\infty} M_k^{(j)} = 0$ for every $j \in \{1, \ldots, m\}$.*

Intuitively, $M_k^{(j)}$ measures how concentrated the distribution for $\bot_j$ is. If MPP vanishes, then no single value that can be assigned to a null dominates when the number of possible values is large.

3.2 The General 0–1 Law

With the probabilistic framework in place, we present our main result, which generalizes Theorem 1 to a heterogeneous model with arbitrary independent distributions for nulls.

Theorem 3 (General 0–1 Law)**.** *Let Q be a C-generic query and D an incomplete database with $\mathsf{Null}(D) = \{\bot_1, \ldots, \bot_m\}$. Under the heterogeneous model defined above, and the vanishing MPP assumption, we have:*

1. *$\lim_{k\to\infty} \mathbb{P}_k(Q, D, \bar{t})$ exists and is either 0 or 1 for every tuple $\bar{t} \in \mathsf{dom}(D)^n$, and*
2. *$\mathit{cert}_\infty(Q, D) = \mathit{naïve}(Q, D)$.*

Proof: It is known (Libkin 2018) that the 0–1 law follows if the probability of a valuation being C-bijective converges to 1. This is because for any C-generic query Q, all C-bijective valuations v yield the same result, which corresponds to the naïve evaluation naïve(Q, D). What remains is to show that $\mathbb{P}_k^D(v \text{ is } C\text{-bijective}) \to 1$.

For this, as the first step, we define $\mathrm{Col}_k^{(i,j)}$ as the probability that the values independently drawn for $\bot_i$ and $\bot_j$ coincide. Formally, $\mathrm{Col}_k^{(i,j)} := \sum_{c\in C_k} p_k^{(i)}(c)\, p_k^{(j)}(c)$. We claim that $\mathrm{Col}_k^{(i,j)} \le \min\big(M_k^{(i)}, M_k^{(j)}\big)$, and hence that $\lim_{k\to\infty} \mathrm{Col}_k^{(i,j)} = 0$ under the vanishing MPP assumption. Indeed, since $p_k^{(i)}(c) \le M_k^{(i)}$ for every $c \in C_k$, we have $\mathrm{Col}_k^{(i,j)} \le \sum_{c\in C_k} M_k^{(i)}\, p_k^{(j)}(c) = M_k^{(i)} \cdot \sum_{c\in C_k} p_k^{(j)}(c) = M_k^{(i)}$; the symmetric bound $\mathrm{Col}_k^{(i,j)} \le M_k^{(j)}$ follows by exchanging the roles of i and j. Thus $\mathrm{Col}_k^{(i,j)} \le \min\big(M_k^{(i)}, M_k^{(j)}\big)$, which tends to 0 as k grows under the vanishing MPP assumption.

Let $X := C \cup \mathsf{Const}(D)$ be the finite set of constants that a C-bijective valuation must avoid when assigning fresh values to nulls. A valuation $v \in \mathcal{V}_k(D)$ is not C-bijective if and only if either (i) it maps some null to an element of X, i.e., $v(\bot_j) \in X$ for some j, or (ii) it identifies two distinct nulls, i.e., $v(\bot_i) = v(\bot_j)$ for some $i < j$. Write $M_k^{\max} := \max_{j\le m} M_k^{(j)}$. For (i), for any $x \in X$ we have $\mathbb{P}_k^D(v(\bot_j) = x) = p_k^{(j)}(x) \le M_k^{(j)} \le M_k^{\max}$, hence by a union bound $\mathbb{P}_k^D(\exists j :\ v(\bot_j) \in X) \le m|X|\, M_k^{\max}$. For (ii), $\mathbb{P}_k^D(v(\bot_i) = v(\bot_j)) = \mathrm{Col}_k^{(i,j)} \le M_k^{(j)} \le M_k^{\max}$ for each pair $i < j$, and thus $\mathbb{P}_k^D(\exists i < j :\ v(\bot_i) = v(\bot_j)) \le \binom{m}{2} M_k^{\max}$. Combining the two failure events gives $\mathbb{P}_k^D(v \text{ is not } C\text{-bijective}) \le (m|X| + \binom{m}{2}) M_k^{\max} \to 0$ since $M_k^{\max} \to 0$ by the hypothesis and $m, |X|$ do not depend on k, proving $\mathbb{P}_k^D(v \text{ is } C\text{-bijective}) \to 1$. □

3.3 Applicability to Specific Distributions

We now assess the applicability of Theorem 3 to probability distributions encountered in practice. Recall that our framework relies on one sufficient condition: the *Maximum Point Probability* (M_k) must vanish as the domain size k grows. In this section, we verify this condition for common discrete (binomial, Zipf) and discretized continuous distributions.

For aggregate counters or summation queries, data is often modeled by the *Binomial Distribution* $B(k, p)$, where the asymptotic parameter is the number of trials k.

Proposition 1. *The Binomial distribution satisfies the condition $\lim_{k\to\infty} M_k = 0$.*

Proof: We analyze the Maximum Point Probability for the binomial distribution $B(k, p)$ as the number of trials k grows. By the De Moivre-Laplace theorem, for large k, the binomial distribution converges to a Gaussian PDF: $\mathbb{P}(X = x) \approx \big(2\pi kp(1-p)\big)^{-1/2} \cdot \exp\left(-\frac{(x-kp)^2}{2kp(1-p)}\right)$. The maximum probability (M_k) occurs at the mode ($x \approx kp$), where the exponential term is 1. When p is constant, M_k is proportional to the inverse square root of k, i.e., $M_k \approx (2\pi kp(1-p))^{-1/2} \propto k^{-1/2}$. Hence, $\lim_{k\to\infty} M_k = 0$. □

The *Zipf (Power-Law)* distribution models skewed data, such as node degrees in social graphs or word frequencies in text. The probability mass function over a domain of size k is $\mathbb{P}(X = x) = \frac{x^{-s}}{H_{k,s}}$, where s is the skew parameter and $H_{k,s}$ is the generalized harmonic number.

Proposition 2. *The Zipf distribution satisfies the condition for the 0–1 Law if the skew $s \le 1$.*

Proof: The maximum point probability is $M_k = 1/H_{k,s}$. For $s > 1$, the series converges to the Riemann zeta function $\zeta(s)$, so $\lim_{k\to\infty} M_k > 0$, while for $s \le 1$, the series diverges ($\lim_{k\to\infty} H_{k,s} = \infty$), thus $\lim_{k\to\infty} M_k = 0$. □

Hence, for nulls distributed according to the binomial or Zipf distributions with $s \le 1$, naïve evaluation captures almost certain answers.

Extension to Continuous Distributions Our framework naturally accommodates continuous probability models via *discretization* of continuous PDFs. Given a parameter k, this is done in a two-step process:

1. **Truncation:** Restrict the distribution to a fixed bounded interval $[A, B]$.
2. **Discretization:** Partition this interval into k intervals (or "segments"). Let $\Pi_k = \{A_1, \ldots, A_{k-1}\}$ so that $A = A_0$, and $A_0 < A_1 < \cdots < A_{k-1} < A_k$, where $B = A_k$. This forms a partition of $[A, B]$ into k intervals, defined as $S_i = [A_{i-1},\ A_i]$ for $i = 1, \ldots, k$.

This approximates the continuous PDF with a discrete distribution on $\{S_1, \ldots, S_k\}$ where the probability $p_i^{(k)}$ of the ith element is $\Big(\int_{[A_{i-1},A_i]} f(x)dx\Big) / \big(\int_{[A,B]} f(x)dx\Big)$, and puts us back in the realm of discrete measures and makes it possible to talk about the maximum point probability M_k.

To ensure it asymptotically approaches zero, we require the *mesh size* (or norm) of the partition, denoted $\|\Pi_k\| = \max_{i \le k}(A_i - A_{i-1})$, to vanish as $k \to \infty$.

Lemma 1 (Vanishing Mesh implies Vanishing M_k)**.** *Let $f(x)$ be any PDF on $\mathbb{R}$. If a bounded domain $[A, B]$ is discretized by a sequence $\{\Pi_k\}$ as above such that $\lim_{k\to\infty} \|\Pi_k\| = 0$, then $\lim_{k\to\infty} M_k = 0$.*

Proof: Let f_{tr} be the normalized PDF on $[A, B]$. Since $f_{tr} \in L^1([A, B])$, the integral is absolutely continuous: for every $\varepsilon > 0$, there exists $\delta > 0$ such that for any set S with measure $m(S) < \delta$, the integral $\int_S f_{tr}$ is bounded above by ε. Since the partition norm vanishes ($\lim_{k\to\infty} \|\Pi_k\| = 0$), for sufficiently large k, every segment $S_i \in \Pi_k$ has measure at most δ. Consequently, the probability mass of every segment is bounded by ε, implying $M_k = \max_i p_i^{(k)} \le \varepsilon$. Thus, $\lim_{k\to\infty} M_k = 0$. □

With this technical lemma established, we can now state the main result for continuous data.

Theorem 4 (0–1 law for Discretized Continuous Data)**.** *Let Q be a C-generic query and D an incomplete database where nulls are drawn from independent continuous distributions discretized according to a partition sequence with vanishing norm. Then, the set of almost certain answers is exactly the result of naïve evaluation.*

Proof: By Lemma 1, the discretization ensures that $\lim_{k\to\infty} M_k = 0$. This satisfies the sufficient condition of the General 0–1 Law (Theorem 3). Thus, the naïve evaluation is asymptotically correct. □

Summary of Applicability In summary, we have shown that all of these distributions—the Binomial distribution, the Zipf distribution (with $s \le 1$), and every discretized continuous distribution with a vanishing partition norm—satisfy the condition for the 0–1 Law.

4 Robustness: Extension to Property Graphs

We now demonstrate that the general 0–1 law applies beyond the relational model. Specifically, we look at a very popular model of *property graphs* that led to two new recent ISO standards SQL/PGQ and GQL. We work with a recently proposed theoretical core of GQL that also captures many features of other graph languages such as Cypher, PGQL, GSQL etc. (Francis et al. 2018; van Rest et al. 2016; Deutsch et al. 2020).

4.1 Incomplete Property Graphs

We consider a model where the graph's topology and labels are complete (have no nulls), but its properties may be incomplete. The reason for this choice is twofold. First, in real-life implementations of the property graph models, nulls only apply at the level of properties: node and edge identifiers act as primary keys and by design cannot be null. Second, it is known that if nulls are allowed at the level of node and edge ids, then even very simple queries become intractable and naïve evaluation no longer helps (Barceló, Libkin, and Reutter 2014).

Assume disjoint sets of labels L, keys K, and sets Const of constants and Null of marked nulls as in the relational case. An *incomplete property graph* is a tuple $G = (N_G, E_G, \mathrm{lab}_G, \mathrm{src}_G, \mathrm{tgt}_G, \mathrm{prop}_G)$. Here, N_G and E_G are finite sets of node and edge ids; $\mathrm{lab}_G : N_G \cup E_G \to 2^{\mathsf{L}}$ assigns label sets to nodes and edges; $\mathrm{src}_G, \mathrm{tgt}_G : E_G \to N_G$ define source and target of an edge, and $\mathrm{prop}_G : (N_G \cup E_G) \times \mathsf{K} \to (\mathsf{Const} \cup \mathsf{Null})$ is a partial function that associates values (constants or nulls) with keys. We assume that for every key k mentioned in a query, $\mathrm{prop}_G(\cdot, k)$ is defined on all nodes/edges that the query may access; missing information is represented by marked nulls.

The *active domain* of G, denoted $\mathsf{dom}(G)$, is the set of all constants and nulls in the image of prop_G. Sets $\mathsf{Const}(G)$ and $\mathsf{Null}(G)$ are defined as for the relational model.

A graph G is *complete* if the property map prop_G does not map to any marked null. Valuations v are mappings from $\mathsf{Const} \cup \mathsf{Null}$ to Const as defined in the relational case (Definition 1). Let $\mathcal{V}(G)$ denote the set of all such valuations. For $v \in \mathcal{V}(G)$, we define $v(G)$ as the property graph $(N_G, E_G, \mathrm{lab}_G, \mathrm{src}_G, \mathrm{tgt}_G, v \circ \mathrm{prop}_G)$, with v applied to all property values.

4.2 CoreGQL Syntax and Semantics

As a theoretical model, we use the recently proposed language CoreGQL that captures many features of the GQL standard including the essentials of pattern matching while being amenable to theoretical analyses (Gheerbrant et al. 2025). We focus on path patterns here, as the remaining components of the language consist of standard relational operations to which results in Section 3 apply directly.

Syntax Let $x, y \in \mathsf{Vars}$, $k \in \mathsf{K}$, $c \in \mathsf{Const}$, and $\ell \in \mathsf{L}$. Let $1 \le n \le m \le \infty$. Variables in node (x) and edge patterns $\xrightarrow{x}, \xleftarrow{x}$ can be omitted. The syntax for CoreGQL path patterns π and graph conditions θ is:

$$\pi := (x) \mid \xrightarrow{x} \mid \xleftarrow{x} \mid \pi_1 \cdot \pi_2 \mid \pi_1 + \pi_2 \mid \pi\langle\theta\rangle \mid \pi^{n..m}$$
$$\theta := x.k = y.k \mid x.k = c \mid x : \ell \mid \theta_1 \wedge \theta_2 \mid \theta_1 \vee \theta_2 \mid \neg\theta$$

$[\![()]\!]_G^{\text{cond}}$	$\{\langle \text{path}(n), \mu_\emptyset, \text{true}\rangle \mid n \in N_G\}$ *(anonymous node pattern binds no variable)*
$[\![(x)]\!]_G^{\text{cond}}$	$\{\langle \text{path}(n), \{x \mapsto n\}, \text{true}\rangle \mid n \in N_G\}$
$[\![\overset{x}{\rightarrow}]\!]_G^{\text{cond}}$	$\{\langle \text{path}(u,e,v), \{x \mapsto e\}, \text{true}\rangle \mid e \in E_G, \text{src}_G(e) = u, \text{tgt}_G(e) = v\}$ *($\overset{x}{\leftarrow}$ is defined similarly)*
$[\![\pi_1 \cdot \pi_2]\!]_G^{\text{cond}}$	$\{\langle p_1 \odot p_2, \mu_1 \cup \mu_2, \varphi_1 \wedge \varphi_2\rangle \mid \langle p_i, \mu_i, \varphi_i\rangle \in [\![\pi_i]\!]_G^{\text{cond}}, \mu_1 \sim \mu_2, p_1 \odot p_2 \text{ defined}\}$
$[\![\pi_1 + \pi_2]\!]_G^{\text{cond}}$	$\left\{\langle p, \mu, \varphi_1^{p,\mu} \vee \varphi_2^{p,\mu}\rangle \mid (p,\mu) \in M_1 \cup M_2\right\}$ *where* $M_i := \{(p,\mu) \mid \exists\varphi.\ \langle p,\mu,\varphi\rangle \in [\![\pi_i]\!]_G^{\text{cond}}\}$ *and* $\varphi_i^{p,\mu} := \varphi$ *if* $\langle p,\mu,\varphi\rangle \in [\![\pi_i]\!]_G^{\text{cond}}$ *else* false.
$[\![\pi\langle\theta\rangle]\!]_G^{\text{cond}}$	$\{\langle p, \mu, \varphi \wedge \theta^\mu\rangle \mid \langle p,\mu,\varphi\rangle \in [\![\pi]\!]_G^{\text{cond}}\}$ where $(x{:}\ell)^\mu := \text{true}$ if $\ell \in \text{lab}_G(\mu(x))$ else false and $(x.k = c)^\mu := (\text{prop}_G(\mu(x), k) = c)$; others $(x.k = y.k, \wedge, \vee, \neg)$ defined similarly.
$[\![\pi^{n..m}]\!]_G^{\text{cond}}$	$\bigcup_{k=n}^{m} \mathcal{M}^k$, where $\mathcal{M}^k$ is defined recursively: $\mathcal{M}^0 := \{\langle \text{path}(n), \mu_\emptyset, \text{true}\rangle \mid n \in N_G\}$ $\mathcal{M}^k := \left\{ \left\langle p, \mu_\emptyset, \bigvee_{\substack{(p_1,\ldots,p_k) \\ p_1 \odot \cdots \odot p_k = p}} \left(\bigwedge_{j=1}^k \varphi_j\right)\right\rangle \;\middle|\; \exists\langle p_1,\mu_1,\varphi_1\rangle,\ldots,\langle p_k,\mu_k,\varphi_k\rangle \in [\![\pi]\!]_G^{\text{cond}} \right\}$

Table 2: Conditional Path Table construction of CoreGQL patterns ($[\![\pi]\!]_G^{\text{cond}}$).

Free variables A pattern π exposes a set $\mathsf{FV}(\pi) \subseteq \mathsf{Vars}$ of *free variables* that become the columns of the relation returned by pattern matching. For atomic patterns, we have $\mathsf{FV}(()) = \mathsf{FV}(\rightarrow) = \emptyset$, and $\mathsf{FV}((x)) = \mathsf{FV}(\overset{x}{\rightarrow}) = \{x\}$ (and similarly for $\leftarrow$ and $\overset{x}{\leftarrow}$). Inductively, $\mathsf{FV}(\pi_1{\cdot}\pi_2) = \mathsf{FV}(\pi_1) \cup \mathsf{FV}(\pi_2)$ and $\mathsf{FV}(\pi\langle\theta\rangle) = \mathsf{FV}(\pi)$. Finally, $\mathsf{FV}(\pi^{n..m}) = \emptyset$, in order to disallow lists as possible atomic output values. To further guarantee that pattern matching returns strictly relational outputs, we use set semantics, and only allow disjunctions $\pi_1 + \pi_2$ when $\mathsf{FV}(\pi_1) = \mathsf{FV}(\pi_2)$.

Semantics. A *match* for a path pattern π is a pair (p, μ) with

- **Path (p):** An alternating sequence $n_0, e_1, n_1, \ldots, e_k, n_k$ from G, where each edge e_i connects n_{i-1} and n_i.
- **Binding (μ):** A partial mapping $\mu : \mathsf{Vars} \to (N_G \cup E_G)$ assigning graph elements to variables.

For a complete graph G, the evaluation of a pattern π, denoted $[\![\pi]\!]_G$, yields the set of all matches (p, μ) that satisfy π (Gheerbrant et al. 2025). We now provide the definition while also generalizing to conditional path tables.

4.3 Conditional Path Tables (CPT)

These extend the framework of Conditional Tables (Section 2.4) to property graphs.

Definition 9 (Conditional Match)**.** *A* conditional match *is a triple $\langle p, \mu, \varphi\rangle$ where p is a path, μ is a binding, and φ is a condition (as defined in Section 2.4).*

A *Conditional Path Table* (CPT) is a set of such conditional matches. The evaluation of a pattern π, denoted $[\![\pi]\!]_G^{\text{cond}}$, produces a CPT according to the rules in Table 2, that use the following conventions and notations.

The *empty binding* is denoted $\mu_\emptyset$. With a slight abuse of notation, we use $\mathsf{dom}(\mu)$ to denote the set of variables defined in μ. Two mappings are compatible, written $\mu_1 \sim \mu_2$, if $\mu_1(v) = \mu_2(v)$ for all $v \in \mathsf{dom}(\mu_1) \cap \mathsf{dom}(\mu_2)$. We write $p_1 \odot p_2$ for the concatenation of two paths, defined when the last node of p_1 is the first node of p_2.

Following the relational case, Table 2 pairs each candidate match (p, μ) with a Boolean formula φ that encodes exactly when the match holds under a given valuation.

By induction on the structure of the expressions, in the spirit of the proof of Theorem 2, one can show:

Theorem 5 (Correctness of Conditional GQL Evaluation)**.** *For every CoreGQL pattern π, incomplete property graph G, valuation $v \in \mathcal{V}(G)$, and path pattern match (p, μ):*

$$(p,\mu) \in [\![\pi]\!]_{v(G)} \Leftrightarrow \exists\langle p,\mu,\varphi\rangle \in [\![\pi]\!]_G^{cond} \text{ such that } v \models \varphi.$$

4.4 The 0–1 law for GQL

We now show that, similarly to the relational case, CoreGQL queries satisfy the 0–1 law, and almost certainty can be captured by naïve evaluation. As in the relational case, we work over a sequence of finite domains $C_k = \{c_1, \ldots, c_k\}$ and the valuation space $\mathcal{V}_k(G)$ mapping the nulls of a graph G to C_k. The *certainty measure* $\mathbb{P}_k(\pi, G, p, \mu)$ is the total measure of valuations $v \in \mathcal{V}_k(G)$ such that $(p, \mu) \in [\![\pi]\!]_{v(G)}$.

As for the naïve evaluation of a graph pattern π, notice that pairs (p, μ) it produces refer only to nodes and edges, which are not affected by valuations of nulls. Let C be the set of all constants mentioned in π, and let v, v' be two C-bijective valuations. It is then easy to see that $(p, \mu) \in [\![\pi]\!]_{v(G)}$ if and only if $(p, \mu) \in [\![\pi]\!]_{v'(G)}$. Hence, we define $[\![\pi]\!]_G^{\text{naïve}}$ as the set of all pairs (p, μ) that are in $[\![\pi]\!]_{v(G)}$ for some (and hence for all) C-bijective valuations v.

Theorem 6 (GQL 0–1 Law)**.** *Let π be a CoreGQL pattern and G an incomplete property graph with* $\mathsf{Null}(G) = \{\bot_1, \ldots, \bot_m\}$*. Under the heterogeneous model and the vanishing MPP assumption of Section 3.1, we have:*

1. $\lim_{k\to\infty} \mathbb{P}_k(\pi, G, p, \mu)$ *exists and is either* 0 *or* 1 *for every path* p *and binding* μ*, and*
2. $(p, \mu) \in [\![\pi]\!]_G^{naïve} \iff \lim_{k\to\infty} \mathbb{P}_k(\pi, G, p, \mu) = 1$.

The proof follows the approach of the proof for relational algebra using c-tables, sketched at the end of Section 2.4.

5 Empirical Experiments on Convergence

Sections 3 and 4 establish a qualitative guarantee: for C-generic queries under independent null valuations with vanishing maximum point probability, naïve evaluation characterizes exactly the almost certain answers. We now address quantitative questions that go beyond our theoretical results:

1. How often do naïve answers differ from certain answers?
2. At what domain sizes do naïve answers become reliable in practice?

Regarding the first question, had naïve evaluation almost always produced certain answers in practice, there would not have been any need to find solutions to the certain answer problem, except in rare cases. We shall see, however, that this is not the case; the challenge in this first set of experiments is to solve the generally intractable certain answer problem for real-life queries.

Regarding the second question, recall that the 0–1 law talks about asymptotic behavior. The convergence to it may be fast, or slow, and it depends on how many domain elements are there for each null to be mapped into. What we would like to see is the high values of the certainty measure at realistically achievable ratios of domain values to nulls.

5.1 Naïve Evaluation versus Certainty

We now analyze the frequency of false positives. While naïve evaluation is easy to compute, certain answers for queries we consider are coNP-complete, rendering verification computationally prohibitive for large datasets. To overcome this, our strategy is to combine conditional tables with powerful solvers. Specifically, if $\langle \bar{t}, \varphi \rangle$ is a c-tuple in $[\![Q]\!]_D^{\text{cond}}$ (see notations in Section 2.4), then it is certain if and only if φ is valid. To ensure such formulas, whose size now depends on the size of the database, can be handled by a solver, we restrict our exact ground truth validation to small scale factors (SF 0.01 for the TPC-H benchmark, and SF 0.1 for the LDBC benchmark) with 1%–5% injected nulls. We are interested in queries that have negation in some form; otherwise we know that certain answers and naïve evaluation coincide. Thus, we took Q1 – Q4 from (Guagliardo and Libkin 2016), which are queries over the TPC-H schema slightly modified to retain only features expressible in relational algebra, alongside one graph query (GQL Q5).

We implemented a three-step evaluation pipeline[1]:

1. **Engines:** We use PostgreSQL and Neo4j to execute naïve evaluation.
2. **C-Table Construction:** For each query, we apply the conditional (path) tables construction from Sections 2.4 and 4.3 to derive the logical condition φ for each naïve evaluation answer.
3. **Verification:** We use the Z3 SMT Solver (de Moura and Bjørner 2008) to simplify and check the validity of φ, separating certain answers from false positives.

[1]Source code and datasets are available at https://github.com/liuhengl/KR26_01laws

The following table details the results across varying null rates. In each cell, we report the absolute number of false positives identified by the SMT solver, followed by the false positive rate in parentheses.

Null Rate	Q1 (SQL)	Q2 (SQL)	Q4 (SQL)	Q5 (GQL)
1%	1 (0.4%)	165 (100%)	294 (0.4%)	243 (36.2%)
2%	4 (1.6%)	165 (100%)	617 (0.8%)	337 (50%)
3%	2 (0.8%)	165 (100%)	1035 (1.4%)	409 (60.2%)
4%	7 (3.0%)	165 (100%)	1418 (1.9%)	440 (64.2%)
5%	11 (4.9%)	165 (100%)	1888 (2.5%)	469 (67.9%)

*Q3 had 0 false positives and is omitted from the table.

This confirms that in general naïve evaluation could be unsound in practice. One query (Q2) fails for every answer (100% false positive rates), and the graph query (Q5) sees rates climb from 36.2% to 67.9% as null density increases. This justifies more detailed analysis of convergence rates, to see how quickly these false positives become almost certain.

5.2 Evaluating Convergence

Before presenting our experimental results, we discuss how the experiments need to be set up, and what would be a good result. There are two hypotheses we would like to test:

- **RH1 (Typical Fast Convergence):** For typical benchmark queries and common probability distributions, the certainty of naïve answers rises quickly toward 1 as the domain grows.
- **RH2 (Existence of Slow Convergence):** There exist at the same time adversarial cases, with a significant skew in distributions, where convergence is too slow for naïve answers to remain reliable.

The parameter we are interested in is the number of possible values a null can take. More specifically, along the

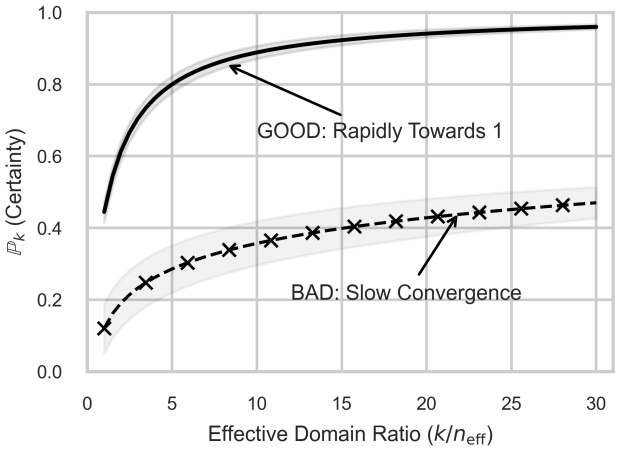
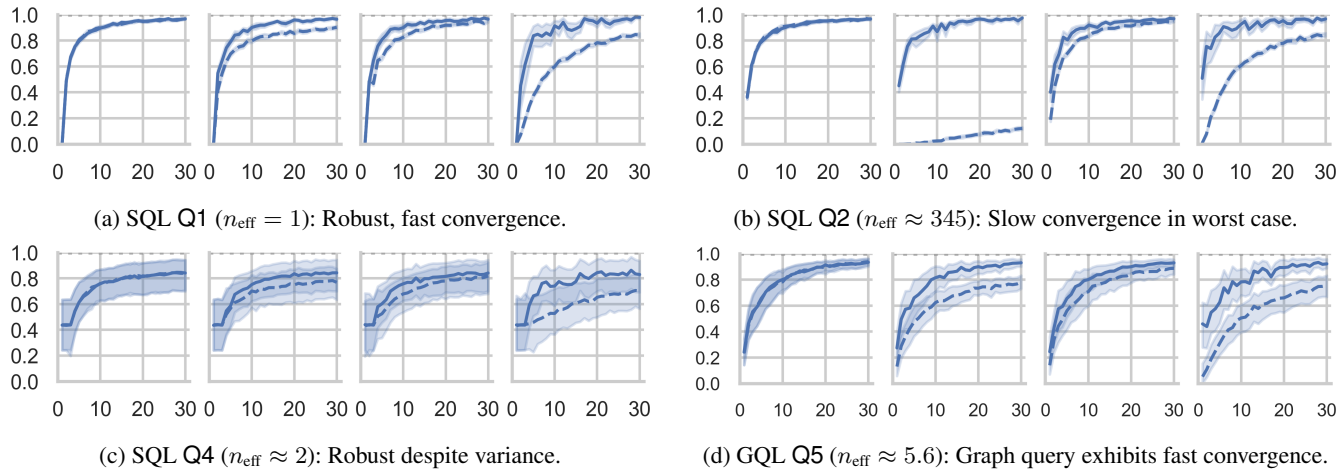


Figure 1: **Illustrating Convergence.** Comparison between fast (good) and slow (bad) behavior.



(a) SQL Q1 ($n_{\text{eff}} = 1$): Robust, fast convergence.

(b) SQL Q2 ($n_{\text{eff}} \approx 345$): Slow convergence in worst case.

(c) SQL Q4 ($n_{\text{eff}} \approx 2$): Robust despite variance.

(d) GQL Q5 ($n_{\text{eff}} \approx 5.6$): Graph query exhibits fast convergence.

Figure 2: **Convergence Analysis (SQL & GQL).** Each subfigure (a–d) shows the certainty measure $\mathbb{P}_k$ (y-axis) as a function of the effective domain ratio k/n_{eff} (x-axis), across four plots representing Uniform, Zipfian, Normal, and Exponential distributions (left to right). For non-uniform distributions, we compare average-case (solid lines; random probability-mass assignment) and worst-case (dashed lines; adversarial assignment chosen to lower $\mathbb{P}_k$). Shading indicates 95% confidence intervals.

x-axis, we consider the *effective domain ratio* k/n_{eff}, where k is the domain size, and n_{eff} is the count of *distinct* nulls appearing in the (Z3-simplified) condition φ for a specific candidate tuple obtained by the c-table evaluation. On the y-axis, we track the *certainty measure* $\mathbb{P}_k$ (Definition 5), i.e., how close to 1 the answer is. Note that conditions φ associated with different candidate answers contain varying numbers of nulls, and thus convergence behaviors are not directly comparable using the absolute domain size k. Hence, normalizing k by n_{eff} standardizes the scale, effectively measuring the inverse of the *null-to-constant ratio*. While n_{eff} is bounded by the total number of nulls in the database, it is typically a much smaller number reflecting only the subset of nulls relevant to the specific candidate answer.

The Figure 1 illustrates what we would expect to see in the case of good vs. bad behavior.

Success Criteria As these experiments do not optimize for runtime, we judge outcomes by whether naïve answers become reliable at *practically plausible* ratios. We use $\mathbb{P}_k \geq 0.8$ as an interpretable marker of "high reliability" and report the ratios at which curves cross this threshold.

Results and Conclusions We sampled 20 distinct formulas φ per query (across the four queries with false positives) to capture structural diversity. We computed the certainty measure $\mathbb{P}_k$ using the valuation distribution $\mathbb{P}_k^D$ from Section 3, with domain sizes k reported via the *effective domain ratio* k/n_{eff}. The distributions include Uniform, Zipfian ($s = 0.5$), Normal ($\mathcal{N}(50, 25)$), and Exponential ($\lambda = 0.05$) types, discretized and mapped via *Average-Case* (random) or *Worst-Case* (adversarial) strategies. Plots show the mean of $\mathbb{P}_k$ with 95% confidence intervals.

The aggregated convergence results are shown in Figure 2; in the figure captions we report n_{eff} averaged over the sampled formulas.

RH1: Typical Fast Convergence. Across all cases, we observe fast convergence under Uniform, Normal, average-case Exponential, and average-case Zipfian distributions. The certainty measure $\mathbb{P}_k$ exceeds 0.8 once the effective domain ratio $k/n_{\text{eff}} \approx 20$. This suggests that when the domain is moderately larger than the number of relevant nulls (roughly 5% or smaller null density, which is very common), naïve answers are highly reliable.

RH2: Skew and Slow Convergence. For the query Q2 with a high count of relevant nulls ($n_{\text{eff}} \approx 345$), worst-case Zipfian skew dramatically delays convergence, as seen in Figure 2b. The ratio required to reach $\mathbb{P}_k \geq 0.8$ increases from ≈ 4.5 (Uniform) to ≈ 1735 (Worst-case Zipfian). This identifies a clear boundary: while the 0–1 law guarantees eventual convergence to 1, the rate is impractically slow due to high n_{eff} and adversarial skew.

6 Related Work and Conclusion

Related Work The high complexity of certain answers motivated a wide range of alternative approaches. A prominent line of work addresses incompleteness via *data imputation*. Techniques range from statistical methods such as Multiple Imputation (MI) (Rubin 1987; van Buuren and Groothuis-Oudshoorn 2011) to learning-based approaches, including deep generative models such as GAIN (Yoon, Jordon, and van der Schaar 2018), and their integration into database systems (Perini and Nikolic 2024). These methods estimate missing values in order to obtain a completed dataset on which standard query processing can be applied. In contrast, we do not attempt to repair or complete the data, providing instead guarantees for query answering directly over incomplete databases.

Another well-studied strategy for regaining tractability is to compute *sound under-approximations* of certain answers (Guagliardo and Libkin 2016; Greco, Molinaro, and Trubitsyna 2019; Feng et al. 2019). These approaches rely on query rewriting or restricted evaluation techniques that

sacrifice completeness in exchange for efficiency. Our approach is orthogonal: we obtain efficiency by shifting to asymptotic certainty, eliminating the need for rewriting.

The most relevant theoretical parallel of our results is classical zero–one laws in logic (Glebskii et al. 1969; Fagin 1976). However, the source of randomness and the technical setting are fundamentally different. Classical zero–one laws apply to fully random finite structures whose size tends to infinity, whereas our setting involves a fixed incomplete structure in which randomness arises from valuations of unknown values over an expanding domain. Both lines of work share the phenomenon that asymptotic reasoning lowers complexity: finite validity of first-order sentences is undecidable, but almost-sure validity is PSPACE-complete (Grandjean 1983). Our results can be seen as a database counterpart of this.

Our work also differs from probabilistic database models such as the tuple-independent semantics of (Dalvi, Miklau, and Suciu 2005), where uncertainty concerns the presence or absence of tuples and the expected database size remains bounded. Such models do not satisfy a zero–one law and lead to fundamentally different complexity and semantic properties. The zero–one law for semiring semantics (Grädel et al. 2022) relies on axioms that enforce truth values for facts, whereas uncertainty in our setting arises from the standard model of nulls in a database, without assuming probabilistic independence at the fact/tuple level. The probabilistic database model was generalized to the setting of different distributions for individual nulls rather than tuples (Console, Libkin, and Peterfreund 2023), in the same way we do, but their focus was on establishing measurability of query answers rather than certainty.

Conclusions This work establishes asymptotic certainty as a principled, experimentally validated foundation for reasoning over incomplete relational and graph-structured data. We showed that query answering over incomplete data admits a robust zero-one behavior once certainty is interpreted asymptotically: under minimal probabilistic assumptions, every candidate answer is almost surely true or almost surely false, and the almost-sure answers coincide exactly with those produced by naïve evaluation. This substantially generalizes prior results by removing uniformity assumptions and accommodating realistic uncertainty models, and extends asymptotic reasoning beyond the relational setting to property graphs and GQL. Our empirical study supports these results. The frequency of false positives, established via SMT solvers computing the generally intractable certain answers, justifies the need for asymptotic reasoning. Convergence to almost-sure correctness is rapid even at moderate ratios of domain sizes to the number of nulls, holding across many common distributions, although large skews could limit practical applicability. Overall, this work positions asymptotic certainty as a viable practical alternative to classical certain answers in both relational and graph-based knowledge representation.

Future work may explore extensions to other graph and ontology-based query languages, dependencies between unknowns, and tighter connections with finite model theory.

Acknowledgments

The authors would like to thank the anonymous reviewers for their valuable comments. Part of this work was done while the second author was at the IRIF lab of Université Paris-Cité, supported by ANR project ANR-21-CE48-0015 and a direct grant from RelationalAI. The first and the third author are supported by a grant from NSERC. All authors acknowledge support from a France-Canada collaboration grant.

AI Declaration

The authors have not employed any Generative AI tools.

References

Abiteboul, S.; Hull, R.; and Vianu, V. 1995. *Foundations of Databases*. Addison-Wesley.

Abiteboul, S.; Kanellakis, P.; and Grahne, G. 1991. On the representation and querying of sets of possible worlds. *Theor. Comput. Sci.* 78(1):159–187.

Arenas, M.; Barceló, P.; Libkin, L.; and Murlak, F. 2014. *Foundations of Data Exchange*. Cambridge University Press.

Arenas, M.; Barceló, P.; Libkin, L.; Martens, W.; and Pieris, A. 2022. *Database Theory*. Open source at https://github.com/pdm-book/community.

Barceló, P.; Libkin, L.; and Reutter, J. L. 2014. Querying regular graph patterns. *J. ACM* 61(1):8:1–8:54.

Bienvenu, M., and Ortiz, M. 2015. Ontology-mediated query answering with data-tractable description logics. In Faber, W., and Paschke, A., eds., *Reasoning Web. Web Logic Rules - 11th International Summer School 2015, Berlin, Germany, July 31 - August 4, 2015, Tutorial Lectures*, volume 9203 of *Lecture Notes in Computer Science*, 218–307. Springer.

Calvanese, D.; De Giacomo, G.; Lembo, D.; Lenzerini, M.; and Rosati, R. 2013. Data complexity of query answering in description logics. *Artificial Intelligence* 195:335–360.

Codd, E. F. 1979. Extending the database relational model to capture more meaning. *ACM Trans. Database Syst.* 4(4):397–434.

Console, M.; Guagliardo, P.; Libkin, L.; and Toussaint, E. 2020. Coping with incomplete data: Recent advances. In *Proceedings of the 39th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, PODS'20, 33–47. New York, NY, USA: Association for Computing Machinery.

Console, M.; Libkin, L.; and Peterfreund, L. 2023. Querying incomplete numerical data: Between certain and possibile answers. In Geerts, F.; Ngo, H. Q.; and Sintos, S., eds., *Proceedings of the 42nd ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2023, Seattle, WA, USA, June 18-23, 2023*, 349–358. ACM.

Dalvi, N.; Miklau, G.; and Suciu, D. 2005. Asymptotic conditional probabilities for conjunctive queries. In Eiter,

T., and Libkin, L., eds., *Database Theory - ICDT 2005*, volume 3363 of *Lecture Notes in Computer Science*, 289–305. Springer.

de Moura, L., and Bjørner, N. 2008. Z3: An efficient SMT solver. In *Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, volume 4963 of *Lecture Notes in Computer Science*, 337–340. Springer.

Deutsch, A.; Xu, Y.; Wu, M.; and Lee, V. E. 2020. Aggregation support for modern graph analytics in tigergraph. In Maier, D.; Pottinger, R.; Doan, A.; Tan, W.; Alawini, A.; and Ngo, H. Q., eds., *Proceedings of the 2020 International Conference on Management of Data, SIGMOD Conference 2020, online conference [Portland, OR, USA], June 14-19, 2020*, 377–392. ACM.

Fagin, R. 1976. Probabilities on finite models. *The Journal of Symbolic Logic* 41(1):50–58.

Feng, S.; Huber, A.; Glavic, B.; and Kennedy, O. 2019. Uncertainty annotated databases - a lightweight approach for approximating certain answers. In *Proceedings of the 2019 International Conference on Management of Data*, SIGMOD '19, 1313–1330. New York, NY, USA: Association for Computing Machinery.

Francis, N.; Green, A.; Guagliardo, P.; Libkin, L.; Lindaaker, T.; Marsault, V.; Plantikow, S.; Rydberg, M.; Selmer, P.; and Taylor, A. 2018. Cypher: An evolving query language for property graphs. In *International Conference on Management of Data (SIGMOD)*, 1433–1445. ACM.

Gheerbrant, A.; Libkin, L.; Peterfreund, L.; and Rogova, A. 2025. GQL and SQL/PGQ: theoretical models and expressive power. *PVLDB* 18(6):1798–1810.

Glebskii, Y. V.; Kogan, D. I.; Liogon'kii, M. I.; and Talanov, V. A. 1969. Range and degree of realizability of formulas in the restricted predicate calculus. *Cybernetics* 5(2):142–154.

Grädel, E.; Helal, H.; Naaf, M.; and Wilke, R. 2022. Zero-one laws and almost sure valuations of first-order logic in semiring semantics. In *Proceedings of the 37th Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '22. New York, NY, USA: Association for Computing Machinery.

Grandjean, E. 1983. Complexity of the first-order theory of almost all finite structures. *Inf. Control.* 57(2/3):180–204.

Greco, S.; Molinaro, C.; and Trubitsyna, I. 2019. Approximation algorithms for querying incomplete databases. *Information Systems* 86:28–45.

Guagliardo, P., and Libkin, L. 2016. Making SQL queries correct on incomplete databases: A feasibility study. In *Proceedings of the 35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, PODS '16, 211–223. New York, NY, USA: Association for Computing Machinery.

Hernández, D.; Gutierrez, C.; and Hogan, A. 2018. Certain answers for SPARQL with blank nodes. In Vrandečić, D.; Bontcheva, K.; Suárez-Figueroa, M. C.; Presutti, V.; Celino, I.; Sabou, M.; Kaffee, L.-A.; and Simperl, E., eds., *The Semantic Web – ISWC 2018*, 337–353. Cham: Springer International Publishing.

Imieliński, T., and Lipski, W. 1984. Incomplete information in relational databases. *J. ACM* 31(4):761–791.

Lenzerini, M. 2002. Data integration: A theoretical perspective. In Popa, L.; Abiteboul, S.; and Kolaitis, P. G., eds., *Proceedings of the Twenty-first ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 3-5, Madison, Wisconsin, USA*, 233–246. ACM.

Libkin, L. 2018. Certain answers meet zero-one laws. In *Proceedings of the 37th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, PODS '18, 195–207. New York, NY, USA: Association for Computing Machinery.

Lipski, W. 1984. On relational algebra with marked nulls. In Rosenkrantz, D. J., and Fagin, R., eds., *Proceedings of the Third ACM SIGACT-SIGMOD Symposium on Principles of Database Systems, April 2-4, 1984, Waterloo, Ontario, Canada*, 201–203. ACM.

Perini, M., and Nikolic, M. 2024. In-database data imputation. *Proc. ACM Manag. Data* 2(1).

Reiter, R. 1981. On closed world databases. In Webber, B. L., and Nilsson, N. J., eds., *Readings in Artificial Intelligence*, 119–140. Morgan Kaufmann.

Ross, S. M. 2008. *A First Course in Probability*. Upper Saddle River, NJ: Pearson, 8 edition.

Rubin, D. B. 1987. *Multiple Imputation for Nonresponse in Surveys*. Wiley Series in Probability and Statistics. Nashville, TN: John Wiley & Sons.

van Buuren, S., and Groothuis-Oudshoorn, K. 2011. mice: Multivariate imputation by chained equations in r. *Journal of Statistical Software* 45(3):1–67.

van der Meyden, R. 1998. *Logical approaches to incomplete information: a survey*. USA: Kluwer Academic Publishers. 307–356.

van Rest, O.; Hong, S.; Kim, J.; Meng, X.; and Chafi, H. 2016. PGQL: a property graph query language. In Boncz, P., and Larriba-Pey, J. L., eds., *Proceedings of the Fourth International Workshop on Graph Data Management Experiences and Systems, Redwood Shores, CA, USA, June 24 - 24, 2016*, 7. ACM.

World Health Organization. 2004. *International Statistical Classification of Diseases and Related Health Problems, Tenth Revision (ICD-10)*. 2nd edition. Available online at https://icd.who.int/.

Xiao, G.; Calvanese, D.; Kontchakov, R.; Lembo, D.; Poggi, A.; Rosati, R.; and Zakharyaschev, M. 2018. Ontology-based data access: a survey. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, IJCAI'18, 5511–5519. AAAI Press.

Yoon, J.; Jordon, J.; and van der Schaar, M. 2018. GAIN: Missing data imputation using generative adversarial nets. In Dy, J., and Krause, A., eds., *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, 5689–5698. PMLR.