Logical Schema Design for Data Quality

Ziheng WEI
Wuhan University
Wuhan, China
ziheng.wei@whu.edu.cn

Sebastian LINK
University of Auckland
Auckland, New Zealand
s.link@auckland.ac.nz

***Abstract*—Among many dimensions of data quality, classical schema design only addresses data integrity. In fact, the design for data quality is in its infancy, and general methodologies are simply absent. We propose the assignment of quality degrees to records of data, derived from internally defined measures that an organization perceives as appropriate. The degrees are an opportunity for classical integrity constraints to stipulate quality thresholds that data ought to comply with. This enables us to tailor logical schema design to a given threshold for data quality, which means that database schemata we compute are optimized for update and join operations on data that are fit for purpose.**

***Index Terms*—Cardinality Constraints; Data Quality; Functional Dependency; Normalization; Schema Design;**

I. Introduction

Modern applications present new challenges and opportunities for data management. As an example, applications pose requirements on data quality dimensions such as accuracy, completeness, consistency, and timeliness [1], [2]. The quality of any insight derived from any data-analytical method is limited by the quality of data that is input to the method.

Some thought leaders have summarized the design for data quality as follows [3]. "The problem of measuring and improving the quality of data has been dealt with in the literature as an activity to be performed a posteriori, instead of during the design of the data. As a consequence, the design for data quality is still a largely unexplored area. In particular, while the data-quality elicitation and the translation of data-quality requirements into conceptual schemas have been investigated, there is a lack of comprehensive methodologies covering all the phases of design of information systems." Our goal is to tailor achievements of logical schema design to data quality.

We create an opportunity for achieving this goal by assigning discrete degrees of quality (q-degree) to records of data, based on how well a record meets data quality requirements. In turn, assigning degrees of compliance (c-degree) to integrity constraints enables them to stipulate which q-degrees the constraints ought to comply with. For instance, a constraint holds with the highest c-degree when it applies to records of data even with the lowest q-degree. This duality is thus an opportunity for tailoring schema design to requirements on any target dimensions of data quality.

In classical schema design, Boyce-Codd Normal Form (BCNF) and Third Normal Form (3NF) cannot measure the effort necessary to maintain data integrity, except when some BCNF decomposition exists that preserves all functional dependencies (FDs). Otherwise, FDs are lost or not transformed into keys. Lost or non-key FDs cause an a priori unbounded number of updates to maintain consistency. These gaps were addressed by the ℓ-Bounded Cardinality Normal Form (ℓ-BCNF) condition [4], which states that, for every non-trivial FD, the cardinality constraint must hold that no instance can feature more than ℓ records in which every combination of values over the left-hand side attributes of the FD can occur. The classical BCNF condition is captured when $\ell = 1$. This allows us to compute schemata that are in ℓ-BCNF for the smallest ℓ possible, and thereby capture the minimum effort necessary to maintaining data consistency [4].

We will optimize achievements of ℓ-Bounded Cardinality Normal Form to records of data that meet requirements on any target dimensions of data quality. In particular: **(1)** For every target c-degree, we introduce an infinite, strict hierarchy of ℓ-BCNF. The smallest ℓ for which a schema is in ℓ-BCNF captures both i) the worst-case effort required for maintaining consistency and ii) the highest number of records an instance can join. Hence, design achievements are tailored to compliance requirements on data quality. **(2)** We establish algorithms for computing schema designs. Given some c-degree, one algorithm computes a schema in ℓ-BCNF where ℓ is the minimum integer attainable across all designs that preserve all FDs that meet the c-degree given. **(3)** Experiments show the ability of cardinality constraints to predict the performance of specific update and join operations on records that meet data quality requirements, already during logical design. This also informs the definition of materialized views, quantifying their ability to support incremental maintenance and join support.

Organization. Section II exemplifies how our framework uniformly captures data quality requirements for different dimensions. We introduce a running example in Section III. The underlying model of data is covered in Section IV. In Section V we define hierarchies of ℓ-BCNF for any c-degree. In Section VI we show that schemata in ℓ-BCNF exhibit useful properties pivoted to q-degree of records. Normalization algorithms are tailored to c-degrees in Section VII. Experiments in Section VIII show that our schema designs predict the physical performance of update and join operations for records that meet compliance requirements. Related work is discussed in Section IX. The conclusion and future work are given in Section X.

II. Modeling Data Quality Dimensions

We outline opportunities for quantifying data quality requirements of applications that benefits logical schema design. We do not propose how organizations measure or aggregate measures for data quality. Indeed, this is a task an organization needs to do internally for the application they target. We simply illustrate what opportunities arise by doing so.

The literature on data quality offers many measures of data quality, whose normalized values are within the [0,1] range. Since the measures are not a focus of our work, we simply mention one for a few important dimensions [2], [5].

We may measure the *accuracy* of a record by the ratio of attributes on which the record is perceived to be inaccurate. A record's *completeness* may be measured by the ratio of NOT NULL attributes on which the record has a null marker occurrence. The *consistency* of a record may refer to the ratio of attributes on which inconsistencies had to be resolved when values of the record were integrated from different sources. For *timeliness* of a record we assume there is information available such as a timestamp. These values may be ranked and normalized to a [0,1] range where 1 (0) refers to the most (least) current value.

An organization may choose how many q-degrees are required to model its requirements for data quality, assign specific weights to different quality dimensions, and specify how to aggregate the scores of individual dimensions into a finite linear chain of q-degrees $\alpha_1, \ldots, \alpha_k$, for example by summing up the weighted scores to a total score in [0,1] and assigning α_i when it falls within the ith out of k quantiles, such as $[0, 1/3] \mapsto \alpha_3, (1/3, 2/3] \mapsto \alpha_2, (2/3, 1] \mapsto \alpha_1$. A special q-degree α_{k+1} can be applied to records for which no score is assigned or applicable. The main point is the opportunity for improving data management by making q-degrees available. For schema design, we will take advantage of the dual c-degrees for integrity constraints, enabling the customization of achievements in schema design to compliance requirements on data quality.

III. Application Scenario

As running example consider the event management schema Hap with each record representing an *E(vent)* at a *V(enue)* and *T(ime)* with some *C(ompany)* in charge.

Hap uses FDs to model business rules. The FD $E \to C$ says only one company is in charge of every event, $V \to C$ says there cannot be different companies in charge at the same venue, $VT \to E$ says no different events happen at the same venue at the same time, $ET \to V$ says no event takes place at different venues at the same time, and $CT \to V$ expresses that no company is in charge of different venues at the same time. The minimal keys are ET, VT and CT. Hence, every attribute of Hap is prime. While Hap is in 3NF it has no FD-preserving BCNF decomposition. Given FD-preservation, classical normalization cannot achieve more.

The company prioritizes events into *high* (h) and *normal* (n), which represent q-degrees assigned to records over Hap. Here, *high* is a literal interpretation for the highest q-degree α_1 and *normal* for α_2. Records not in the current instance are assigned bottom q-degree α_3.

TABLE I: Instance r with confirmed (c) and planned (p) events

Hap				
Event	*Venue*	*Company*	*Time*	*q-deg*
Party	v_1	**Kilo**	t_1	h
⋮	⋮	⋮	⋮	⋮
Party	v_{3k}	**Kilo**	t_{3k}	h
e_1	Dome	**Mega**	t'_1	h
⋮	⋮	⋮	⋮	⋮
e_{300}	Dome	**Mega**	t'_{300}	h

Hap				
Event	*Venue*	*Company*	*Time*	*q-deg*
Party	v_{3001}	**Kilo**	t_{3001}	n
⋮	⋮	⋮	⋮	⋮
Party	v_{5k}	**Kilo**	t_{5k}	n
e_{301}	Dome	**Mega**	t'_{301}	n
⋮	⋮	⋮	⋮	⋮
e_{5m}	Dome	**Mega**	t'_{5m}	n

The q-degrees were obtained by i) assigning equal weights of 0.25 to the four data quality dimensions from Section II, ii) measuring each dimension as outlined in Section II, and iii) mapping the weighted sum to *high* when it was at least 0.75.

The FDs above apply to all records with q-degree α_1 or α_2. That is, every FD has highest c-degree β_1, with literal interpretation *fully compliant*. Constraints that only apply to records with q-degree α_1 have c-degree β_2, with literal interpretation *partially compliant*. Constraints that do not even hold on the set of records with q-degree α_1 have c-degree β_3, and are regarded as *not compliant at all*.

There are no a priori bounds for the 3NF schema Hap on the numbers of i) redundant value occurrences, ii) updates necessary to achieve consistency, nor iii) values that can be joined with any given redundant value, and this is true for events of any priority. Recall that a data value occurrence is *redundant* whenever every change to a different value results in a relation that violates some constraint [6].

In practice, natural cardinality bounds apply to events. There are up to 5k records with matching values on *Event*, and up to 5m records with matching values on *Venue*. These cardinality constraints (CCs) hold with c-degree β_1, so apply to high-priority and normal events. We have CCs for high-priority events only: There are up to 3k records with matching values on *Event*, and up to 5k records with matching values on *Venue*.

The CCs guarantee the following bounds on every instance over Hap. For *high*-priority events only, every instance over Hap can feature up to 3k duplicates of any redundant data value, which means at most 3k occurrences of the same redundant data value need to be updated to maintain consistency, and there can be up to 3k values that can be joined with the same redundant data value. For all events together, every instance over Hap can feature up to 5m duplicates of any redundant data value. An instance in which all of these bounds are met is shown in Table I, with redundant data value occurrences shown in bold.

How can a database schema be computed that attains the lowest possible bound for update inefficiency, given we want to preserve all FDs? We will see that decomposition $\mathcal{D}_1$ of Hap in Table IIa is optimized for *high*-priority events, and achieves a worst-case upper bound of 300; while decomposition $\mathcal{D}_2$ in Table IIb is optimized for all events, and achieves a worst-case upper bound of 5k. In fact, Table II shows the projections of

instance r from Table I onto the decompositions, in particular, how occurrences of redundant data values are minimized.

Throughout the paper, we will illustrate our concepts and results on this example, including the experiments.

IV. Schemata and Constraints

We fix notation for our model of data, including constraints.

Relation schemata, denoted by R, are finite, non-empty sets of *attributes*. Each attribute $A \in R$ has a *domain* of values. A *tuple* t over R maps each attribute to a domain value. For subset $X \subseteq R$, $t[X]$ denotes the *projection* of t onto X. A *relation* r over R is a finite set of tuples over R, see Table I.

We quantify how well records meet requirements on data quality dimensions by assigning some degree of quality to each tuple of a relation. Formally, we have a k-ary linear order $(S, <)$ with $k+1$ elements. We write $S = \{\alpha_1, \ldots, \alpha_{k+1}\}$ to declare that $\alpha_1 > \cdots > \alpha_k > \alpha_{k+1}$. The elements α_i are called *quality degrees*, or q-degrees. The q-degree measures how possible it is for a tuple to be part of the current relation, based on how well it meets quality requirements. Hence, α_1 is for tuples that are of 'highest quality' while α_{k+1} is for tuples that are 'not qualified' to occur in the current relation. Tuples in the left of Table I have q-degree α_1, interpreted as *high*-priority; tuples in the right of Table I have q-degree α_2, interpreted as *normal*-priority. All other tuples are 'not qualified' to occur in the current relation r, and carry q-degree α_3. Classical relations use two q-degrees, i.e. $k = 1$.

A *quality schema* R^k, or q-schema, is a relation schema R with a k-ary order $(S, <)$. A *quality relation*, or q-relation, over R^k is a relation r over R with a function Q that assigns to each tuple $t \in r$ a q-degree $Q(t) \in S - \{\alpha_{k+1}\}$. We may omit Q in q-relations. Our example uses q-schema HAP^2, with schema HAP and order $\alpha_1 > \alpha_2 > \alpha_3$.

Q-relations enjoy a possible world semantics. For $i = 1, \ldots, k$ let r_i be the set of tuples in r with q-degree at least α_i, that is, $r_i = \{t \in r \mid Q(t) \geq \alpha_i\}$. We have $r_1 \subseteq r_2 \subseteq \cdots \subseteq r_k$. If $t \notin r_k$, then $Q(t) = \alpha_{k+1}$. Every tuple of 'highest quality' is part of every possible world, and is thus certain to occur. Hence, relations are also q-relations. Table I shows q-relation (r, Q), with r_1 having tuples with q-degree *high*, and r_2 adding tuples of q-degree *normal*.

We extend CCs and FDs to express requirements on data quality. Let $\mathbb{N}^{\infty}_{\geq 1}$ denote the positive integers with ∞. A *cardinality constraint* over relation schema R is an expression $\mathit{card}(X) \leq \ell$ where $X \subseteq R$, and $\ell \in \mathbb{N}^{\infty}_{\geq 1}$. The CC $\mathit{card}(X) \leq \ell$ over R is satisfied by a relation r over R, denoted by $\models_r \mathit{card}(X) \leq \ell$, if there are no $\ell + 1$ distinct tuples $t_1, \ldots, t_{\ell+1} \in r$ with values matching on all $A \in X$. For example, world r_1 satisfies the CCs $\mathit{card}(V) \leq 300$ and $\mathit{card}(E) \leq 3k$, and world r_2 satisfies the CCs $\mathit{card}(E) \leq 5k$ and $\mathit{card}(V) \leq 5m$.

A *functional dependency* over relation schema R is an expression $X \to Y$ where $X, Y \subseteq R$. The FD $X \to Y$ over R is satisfied by a relation r over R, denoted by $\models_r X \to Y$, if for every two tuples $t_1, t_2 \in r$ the following holds: if $t_1[X] = t_2[X]$, then $t_1[Y] = t_2[Y]$. For example, the world r_2 satisfies the FD $E \to C$ and $V \to C$.

Dually to the order $(S, <)$ of q-degrees α_i for tuples, we use the order $(S^T, <)$ of *compliance degrees*, or c-degrees, β_j for CCs and FDs. The *marginal compliance* $c_r(\sigma)$ by which the CC $\sigma = \mathit{card}(X) \leq \ell$ or FD $\sigma = X \to Y$ holds on the q-relation r is either the top degree β_1 if σ is satisfied by r_k, or the c-degree β_{k+2-i} that corresponds to the smallest possible world r_i in which σ is violated, that is,

$$c_r(\sigma) = \begin{cases} \beta_1 & \text{, if } r_k \text{ satisfies } \sigma \\ \beta_{k+2-i} & \text{, if } r_i \text{ smallest world violating } \sigma \end{cases}.$$

We can now define the semantics of quality CCs and FDs. Let R^k denote a q-relation schema. A quality CC (qCC) over R^k is an expression $(\mathit{card}(X) \leq \ell, \beta)$ where $\mathit{card}(X) \leq \ell$ denotes a CC over R and $\beta \in S^T$. A q-relation (r, Q) over R^k satisfies the qCC $(\mathit{card}(X) \leq \ell, \beta)$ if and only if $c_r(\mathit{card}(X) \leq \ell) \geq \beta$. A quality FD (qFD) over R^k is an expression $(X \to Y, \beta)$ where $X \to Y$ denotes an FD over R and $\beta \in S^T$. A q-relation (r, Q) over R^k satisfies the qFD $(X \to Y, \beta)$ if and only if $c_r(X \to Y) \geq \beta$.

For example, the q-relation from Table I satisfies the qFDs $(E \to C, \beta_1)$, $(V \to C, \beta_1)$, $(VT \to E, \beta_1)$, $(ET \to V, \beta_1)$, $(CT \to V, \beta_1)$, and qCCs $(\mathit{card}(V) \leq 300, \beta_2)$, $(\mathit{card}(E) \leq 3k, \beta_2)$, $(\mathit{card}(E) \leq 5k, \beta_1)$ and $(\mathit{card}(V) \leq 5m, \beta_1)$, but neither $(\mathit{card}(V) \leq 300, \beta_1)$ nor $(\mathit{card}(E) \leq 3k, \beta_1)$.

The success of a schema design framework depends on the ability to decide the implication problem associated with the class of constraints required. We recall previous definitions and results on the implication problem for the combined class of qCCs and qFDs [7].

For a set $\Sigma \cup \{\varphi\}$ of qCCs and qFDs over p-schema R^k we say Σ *implies* φ, denoted by $\Sigma \models \varphi$, if every q-relation over R^k that satisfies every element of Σ also satisfies φ. We use $\Sigma^* = \{\varphi \mid \Sigma \models \varphi\}$ to denote the *semantic closure* of Σ. The *implication problem for qCCs and qFDs* is to decide, given any q-schema, and any set $\Sigma \cup \{\varphi\}$ of qCCs and qFDs over the q-schema, whether $\Sigma \models \varphi$ holds.

The β-*cut* of Σ is $\Sigma_\beta = \{\sigma \mid (\sigma, \beta') \in \Sigma \text{ and } \beta' \geq \beta\}$. Informally, the β-cut Σ_β of Σ contains all CCs and FDs σ such that there is some qCCs or qFD (σ, β') in Σ where β' is at least β. The β-cut can be used to reduce the implication problem for qCCs and qFDs to the implication problem of traditional CCs and FDs [7].

A finite axiomatisation allows us to effectively enumerate all implied qCCs and qFDs, that is, to determine the semantic closure $\Sigma^* = \{\sigma \mid \Sigma \models \sigma\}$ of Σ. A finite axiomatisation facilitates human understanding and ensures all opportunities for using the constraints in applications can be exploited. We determine the semantic closure by applying *inference rules* of the form $\frac{\text{premise}}{\text{conclusion}}$. For a set $\mathfrak{R}$ of inference rules let $\Sigma \vdash_{\mathfrak{R}} \varphi$ denote the *inference* of φ from Σ by $\mathfrak{R}$. That is, there is some sequence $\sigma_1, \ldots, \sigma_n$ such that $\sigma_n = \varphi$ and every σ_i is an element of Σ or is the conclusion that results from an application of a rule in $\mathfrak{R}$ to some premises in $\Sigma \cup$

TABLE II: Logical Designs Tailored to Degrees of Quality

(a) $\mathcal{D}_1 = \{R_1, R_3, R_5\}$ for *high*-priority events

R_1

Event	*Company*
Party	Kilo
e_1	Mega
⋮	⋮
e_{300}	Mega

R_3

Event	*Venue*	*Time*
Party	v_1	t_1
⋮	⋮	⋮
Party	v_{3k}	t_{3k}
e_1	Dome	t'_1
⋮	⋮	⋮
e_{300}	Dome	t'_{300}

R_5

Venue	*Company*	*Time*
v_1	Kilo	t_1
⋮	⋮	⋮
v_{3k}	Kilo	t_{3k}
Dome	**Mega**	t'_1
⋮	⋮	⋮
Dome	**Mega**	$t'_{\mathbf{300}}$

(b) $\mathcal{D}_2 = \{R_2, R_3, R_4\}$ for all events

R_2

Venue	*Company*
v_1	Kilo
⋮	⋮
v_{5k}	Kilo
Dome	Mega

R_3

Event	*Venue*	*Time*
Party	v_1	t_1
⋮	⋮	⋮
Party	v_{5m}	t_{5m}
e_1	Dome	t'_1
⋮	⋮	⋮
e_{5k}	Dome	t'_{5k}

R_4

Event	*Company*	*Time*
Party	**Kilo**	t_1
⋮	⋮	⋮
Party	**Kilo**	$t_{\mathbf{5k}}$
e_1	Mega	t'_1
⋮	⋮	⋮
e_{5m}	Mega	t'_{5m}

TABLE III: Finite axiomatisation $\mathfrak{C}$ of qCCs and qFDs

$$\frac{}{(XY \to X, \beta_1)}\ \text{(reflexivity)} \qquad \frac{(X \to Y, \beta)}{(X \to XY, \beta)}\ \text{(extension)} \qquad \frac{(X \to Y, \beta) \quad (Y \to Z, \beta)}{(X \to Z, \beta)}\ \text{(transitivity)}$$

$$\frac{}{(\mathit{card}(R) \leq 1, \beta_1)}\ \text{(top)} \qquad \frac{(\mathit{card}(X) \leq \ell, \beta)}{(\mathit{card}(X) \leq \ell + 1, \beta)}\ \text{(relax)} \qquad \frac{(X \to Y, \beta) \quad (\mathit{card}(Y) \leq \ell, \beta)}{(\mathit{card}(X) \leq \ell, \beta)}\ \text{(pullback)}$$

$$\frac{}{(\mathit{card}(X) \leq \infty, \beta_1)}\ \text{(unbounded)} \qquad \frac{(\mathit{card}(X) \leq 1, \beta)}{(X \to R, \beta)}\ \text{(key)} \qquad \frac{}{(\sigma, \beta_{k+1})}\ \text{(bottom)} \qquad \frac{(\sigma, \beta)}{(\sigma, \beta')}\ \beta' \leq \beta\ \text{(weakening)}$$

$\{\sigma_1, \ldots, \sigma_{i-1}\}$. Let $\Sigma^+_{\mathfrak{R}} = \{\varphi \mid \Sigma \vdash_{\mathfrak{R}} \varphi\}$ be the *syntactic closure* of Σ under inferences by $\mathfrak{R}$. $\mathfrak{R}$ is *sound* (*complete*) if for every set Σ over every R^k we have $\Sigma^+_{\mathfrak{R}} \subseteq \Sigma^*$ ($\Sigma^* \subseteq \Sigma^+_{\mathfrak{R}}$). The (finite) set $\mathfrak{R}$ is a (finite) *axiomatisation* if $\mathfrak{R}$ is both sound and complete. Table III shows an axiomatization for pCCs and pFDs [7]. For example, the inference

$$\frac{(X \to R, \beta) \qquad \dfrac{(\mathit{card}(R) \leq 1, \beta_1)}{(\mathit{card}(R) \leq 1, \beta)}}{(\mathit{card}(X) \leq 1, \beta)}$$

shows qCC $\mathit{card}(X) \leq 1, \beta)$ can be inferred from qFD $(X \to R, \beta)$ using the top axiom, weakening rule, and pullback rule. The key rule shows qCC $(\mathit{card}(X) \leq 1, \beta)$ and qFD $(X \to R, \beta)$ are equivalent.

For $k = 1$, the implication problem for qCCs and qFDs reduces to the implication problem for CCs and FDs, which has been characterised by the set $\mathfrak{L}$ of inference rules in [8].

For a set Σ of CCs and FDs, let Σ[FD] denote the set of FDs in Σ together with the FDs $X \to R$ for every $\mathit{card}(X) \leq 1 \in \Sigma$. Hence, Σ[FD] consists of $\{X \to Y \mid X \to Y \in \Sigma\} \cup \{X \to R \mid \mathit{card}(X) \leq 1 \in \Sigma\}$. For an attribute set $X \subseteq R$, let $X^+_{\Sigma[\text{FD}]} = \{A \in R \mid \Sigma[\text{FD}] \models X \to A\}$ denote the attribute set closure of X under Σ[FD], which can be computed in linear time in the input set Σ[FD].

For a set Σ of qCCs and qFDs over R^k, the following holds: [7]

(i) Σ implies $(X \to Y, \beta)$ iff $Y \subseteq X^+_{\Sigma_\beta[\text{FD}]}$, and

(ii) Σ implies $(\mathit{card}(X) \leq \ell, \beta)$ iff $X^+_{\Sigma_\beta[\text{FD}]} = R$, or there is some $\mathit{card}(Y) \leq \ell' \in \Sigma_\beta$ such that $Y \subseteq X^+_{\Sigma_\beta[\text{FD}]}$ and $\ell' \leq \ell$.

An instance $\Sigma \models \varphi$ of the implication problem for qCCs and qFDs over R^k can be decided in time $\mathcal{O}(|R| \times |\Sigma \cup \{\varphi\}|)$ [7].

V. Bounded Cardinality Normal Form

We establish the family of ℓ-Bounded Cardinality Normal Forms and some computational properties.

A. Normal Form Definition

A schema is in BCNF for c-degree β if the left-hand side (LHS) X of every non-trivial FD $(X \to Y, \beta) \in \Sigma^+_{\mathfrak{C}}$ is a β-key, that is, $(X \to R, \beta) \in \Sigma^+_{\mathfrak{C}}$. As X is a β-key if $(\mathit{card}(X) \leq 1, \beta)$ holds, CCs can be used to relax the BCNF condition for β as follows.

Definition 1: For a set Σ of qCCs and qFDs over q-schema R^k, $\ell \in \mathbb{N}^\infty_{\geq 1}$, and $\beta \in S^T$, (R^k, Σ) is in *ℓ-Bounded Cardinality Normal Form for β* iff for every non-trivial qFD $(X \to Y, \beta) \in \Sigma^+_{\mathfrak{C}}$, we have $(\mathit{card}(X) \leq \ell, \beta) \in \Sigma^+_{\mathfrak{C}}$.

Definition 1 captures ℓ-BCNF for a set of CCs and FDs [4] as the special case $k = 1$. Σ_0 denotes qCCs and qFDs over HAP^2 as follows: $(E \to C, \beta_1)$, $(V \to C, \beta_1)$, $(VT \to E, \beta_1)$, $(ET \to V, \beta_1)$, $(CT \to V, \beta_1)$, $(\mathit{card}(V) \leq 300, \beta_2)$, $(\mathit{card}(E) \leq 3k, \beta_2)$, $(\mathit{card}(E) \leq 5k, \beta_1)$ and $(\mathit{card}(V) \leq 5m, \beta_1)$. Then (HAP^2, Σ_0) is in $5m$-BCNF for β_1, and in $3k$-BCNF for β_2, respectively. Using β-cuts, a q-schema (R^k, Σ) is in ℓ-BCNF for c-degree β iff (R, Σ_β) is in ℓ-BCNF.

Proposition 1: For all q-schemata (R^k, Σ) and all c-degrees β over R^k, we have: (R^k, Σ) is in ℓ-BCNF for β iff (R, Σ_β) is in ℓ-BCNF.

Proof: This follows from the fact that for every CC and FD σ the following holds: $(\sigma, \beta) \in \Sigma^+_{\mathfrak{C}}$ if and only if $\sigma \in (\Sigma_\beta)^+_{\mathfrak{L}}$, see [7]. ∎

The β_1-cut of Σ_0 consists of $E \to C$, $V \to C$, $VT \to E$, $ET \to V$, $CT \to V$, and $\mathit{card}(E) \leq 5k$ and $\mathit{card}(V) \leq 5m$. Indeed, $(\text{HAP}, (\Sigma_0)_{\beta_1})$ is in $5m$-BCNF. The β_2-cut of Σ_0 results from the β_1-cut of Σ_0 by replacing $\mathit{card}(E) \leq 5k$ and $\mathit{card}(V) \leq 5m$ by $\mathit{card}(V) \leq 300$ and $\mathit{card}(E) \leq 3k$. Hence, $(\text{HAP}, (\Sigma_0)_{\beta_2})$ is in $3k$-BCNF.

Being in ℓ-BCNF for a c-degree is independent of different representations for the underlying set of qCCs and qFDs.

Theorem 1: For all covers Σ and Θ over R^k, c-degrees β, and all $\ell \in \mathbb{N}^\infty_{\geq 1}$, (R^k, Σ) is in ℓ-BCNF for β iff (R^k, Θ) is in ℓ-BCNF for β.

Proof: This follows directly from Proposition 1, the fact that Σ_β and Θ_β are covers of CCs and FDs of one another whenever Σ and Θ are covers of qCCs and qFDs, and the fact that (R, Σ) is in ℓ-BCNF if and only if (R, Θ) is in ℓ-BCNF [4, Theorem 3.3]. ■

For every fixed c-degree, the parameter ℓ introduces an infinite strict hierarchy of normal form conditions.

Theorem 2: For all Σ over R^k, all $\ell \in \mathbb{N}^\infty_{\geq 1}$, and β over R^k, every (R^k, Σ) in ℓ-BCNF for β is in $\ell + 1$-BCNF for β, but not vice versa.

Proof: This follows directly from Proposition 1 and the fact that for every relation schema (R, Σ) that is in ℓ-BCNF is also in $\ell + 1$-BCNF but that there are schemata (R, Σ) that are in $\ell + 1$-BCNF but not in ℓ-BCNF [4, Theorem 3.4]. ■

B. Local Efficiency

Invariance under covers leaves open the question whether the property of being in ℓ-BCNF can be decided efficiently, by reference to $\Sigma^+_{\mathfrak{C}}$, which can be exponential in Σ. However, it suffices to consider all qFDs in Σ whose associated c-degree is at least as high as the target c-degree.

Theorem 3: Let Σ be a set of qCCs and qFDs over q-schema R^k, $\ell \in \mathbb{N}^\infty_{\geq 1}$, and β a c-degree of R^k. Then (R^k, Σ) is in ℓ-BCNF for β iff for every non-trivial qFD $(X \to Y, \beta') \in \Sigma$ with $\beta' \geq \beta$, we have $(\mathit{card}(X) \leq \ell, \beta) \in \Sigma^+_{\mathfrak{C}}$.

Proof: (R^k, Σ) is in ℓ-BCNF for β if and only if (R^k, Σ_β) is in ℓ-BCNF if and only if for all non-trivial $X \to Y \in (\Sigma_\beta)^+_{\mathfrak{L}}$ we have $\mathit{card}(X) \leq \ell \in (\Sigma_\beta)^+_{\mathfrak{L}}$. The latter is equivalent to the condition that for all non-trivial $X \to Y \in (\Sigma_\beta)$ we have $\mathit{card}(X) \leq \ell \in (\Sigma_\beta)^+_{\mathfrak{L}}$. However, $X \to Y \in \Sigma_\beta$ is equivalent to $(X \to Y, \beta') \in \Sigma$ for some $\beta' \geq \beta$. ■

It is not always obvious for which smallest ℓ a given schema is in ℓ-BCNF. Let Σ'_0 result from Σ_0 by adding the qCCs $(\mathit{card}(C) \leq 25, \beta_2)$ and $(\mathit{card}(C) \leq 2300, \beta_1)$. By the pullback rule we can infer $(\mathit{card}(E) \leq 25, \beta_2)$, $(\mathit{card}(V) \leq 25, \beta_2)$, $(\mathit{card}(E) \leq 2300, \beta_1)$, $(\mathit{card}(V) \leq 2300, \beta_1)$. Hence, the smallest ℓ for which $(\textsc{Hap}^2, \Sigma'_0)$ is in ℓ-BCNF for β_1 would be $\ell = 2300$, and the smallest ℓ for which $(\textsc{Hap}^2, \Sigma'_0)$ is in ℓ-BCNF for β_2 would be $\ell = 25$.

The smallest ℓ reveals the minimum level of effort required to maintain data consistency. The smallest ℓ will also turn out to be the equilibrium level at which the schema can exhibit a level ℓ join efficiency, and this holds for any q-degree that data should be associated with for the target application.

Corollary 1: We can compute in time $\mathcal{O}(|\Sigma|^2 \times ||\Sigma||)$ the smallest $\ell \in \mathbb{N}^\infty_{\geq 1}$ for which a given (R^k, Σ) is in ℓ-BCNF for a given β.

Proof: Given (R^k, Σ) and the c-degree β, one can compute the smallest $\ell \in \mathbb{N}^\infty_{\geq 1}$ for which (R, Σ_β) is in ℓ-BCNF in time $\mathcal{O}(|\Sigma_\beta|^2 \times ||\Sigma_\beta||)$, where $|\Sigma_\beta| \leq |\Sigma|$ and $||\Sigma_\beta|| \leq ||\Sigma||$. Following Proposition 1, this is the smallest $\ell \in \mathbb{N}^\infty_{\geq 1}$ for which (R^k, Σ) is in ℓ-BCNF for β. ■

TABLE IV: (r, Q)

E	*C*	*T*	q-degree
Party	Kilo	t_1	α_1
⋮	⋮	⋮	
Party	Kilo	t_{3k}	α_1
Party	Kilo	t_{3001}	α_2
⋮	⋮	⋮	
Party	Kilo	t_{5k}	α_2

C. Likely Global Inefficiency

The main problem is computing a decomposition that minimizes the level of effort required to maintain data consistency, limiting data to those associated with the target q-degree or higher. Hence, we need to decide whether a given sub-schema is in ℓ-BCNF for a c-degree β, given ℓ and β. This problem is likely intractable.

Theorem 4: Let (R^k, Σ) be a q-schema, $S \subset R$, $\ell \in \mathbb{N}^\infty_{\geq 1}$, and β a c-degree of R^k. Given $(R, S, \Sigma, \ell, \beta)$, it is coNP-complete to decide whether $(S^k, \Sigma[S])$ is in ℓ-BCNF for β.

Proof: Decide whether $(S^k, \Sigma[S])$ is in ℓ-BCNF for β is equivalent to deciding whether $(S, \Sigma_\beta[S])$ is in ℓ-BCNF. However, given (R, S, Σ, ℓ), it is coNP-complete to decide whether $(S, \Sigma[S])$ is in ℓ-BCNF [4, Theorem 3.7]. ■

VI. Desirable Properties of Schemata

We will show now what ℓ-BCNF actually achieves. The normal form captures relations that require at most ℓ updates to maintain data consistency, given a q-degree as lower bound for records we consider for an application. Similarly, schemata in ℓ-BCNF for the smallest ℓ possible for a given c-degree β, can feature relations that exhibit best-case join efficiency.

A. Levels of α-data redundancy

Intuitively, Vincent [6] defined a single occurrence of a data value as *redundant* whenever *every* change to this occurrence results in a relation that violates some given constraint. Given some positive integer ℓ, a data value occurrence was defined to be *ℓ-redundant* in [4] whenever there are ℓ distinct tuples in which the value occurs and every update to at least one of these ℓ occurrences results in a relation that violates a given constraint.

Clearly, the q-degree associated with tuples has an impact on ℓ-redundancy. For example, the value *Kilo* in q-relation r of Table IV is 2999- but not $3k$-redundant for q-degree α_1, while the same value is 4999- but not $5k$-redundant for q-degree α_2: As Table Va illustrates, concealing between 1 and 2999 occurrences of the value *Kilo* in the *Company*-column still allows us to determine each of the concealed occurrences as $E \to C$ must hold in the world r_1 and there is some tuple left that has a matching E-value and C-value *Kilo*. However, *Kilo* is clearly not $3k$-redundant for α_1 since concealment of all $3k$ occurrences means that the tuples could have any value on C (as long as they all match on C).

Our goal is to formalize these ideas, define a normal form that guarantees the absence of ℓ-redundancy from tuples that

TABLE V: Illustrating ℓ-Redundancy and ℓ-Join Efficiency on Possible Worlds of Quality Relations

(a) 2999-redundancy in the possible world r_1

r_1		
E	C	T
Party	?	t_1
⋮	⋮	⋮
Party	?	t_{2999}
Party	Kilo	t_{3k}

$E \to C$
? = Kilo

r_1		
E	C	T
Party	Kilo	t_1
⋮	⋮	⋮
Party	Kilo	t_{2999}
Party	Kilo	t_{3k}

(b) 5k-join strength in the possible world r_2

r_2		
E	C	T
Party	Kilo	t_1
⋮	⋮	⋮
Party	Kilo	t_{5k}

$=$

$r_2(EC)$	
E	C
Party	Kilo

$\bowtie$

$r_2(ET)$	
E	T
Party	t_1
⋮	⋮
Party	t_{5k}

meet any given q-degree in a q-relation, and show that the absence of ℓ-redundancy for a given q-degree is equivalent to the q-schema being in ℓ-BCNF for the dual c-degree.

We first define the concept of an (α_i, ℓ)-transaction, which causes an actual update to at least one of ℓ given values $t_1(A), \ldots, t_\ell(A)$ in world r_i and at most all of them. Let Σ denote a set of qCCs and qFDs over a q-schema R^k, r a q-relation over R^k that satisfies Σ, $A \in R$, α_i a q-degree of R^k and ℓ a positive integer. Let $t_1, \ldots, t_\ell$ denote ℓ distinct tuples in r_i. An (α_i, ℓ)-*transaction* of $t_1, \ldots, t_\ell$ for A is a set $\{t'_1, \ldots, t'_\ell\}$ of tuples over R such that for all $i = 1, \ldots, \ell$ and for all $A' \in R - \{A\}$, $t'_i(A') = t_i(A')$ and $Q(t'_i) = Q(t_i)$, and there is some $j \in \{1, \ldots, \ell\}$ such that $t'_j(A) \neq t_j(A)$.

Intuitively, the following definition captures ℓ-redundancy for any given q-degree by saying that every (α_i, ℓ)-transaction results in a q-relation that violates some given constraint.

Definition 2: Given (R^k, Σ), a p-relation r over R^k that satisfies Σ, tuple $t \in r$, column $A \in R$, positive integer ℓ and q-degree α_i, the data value occurrence $v = t(A)$ is ℓ-redundant for α_i and $\sigma \in \Sigma^+_{\mathfrak{C}}$ iff there are ℓ distinct tuples $t_1, \ldots, t_\ell \in r_i$ such that $t_1(A) = \ldots = t_\ell(A) = v$, and for every (α_i, ℓ)-transaction $\{t'_1, \ldots, t'_\ell\}$ of $t_1, \ldots, t_\ell$ for A, the q-relation $r' := (r - \{t_1, \ldots, t_l\}) \cup \{t'_1, \ldots, t'_\ell\}$ violates σ.

For example, *Kilo* is 4999-redundant but not 5k-redundant for α_2 and $(E \to C, \beta_1)$, while it is 2999-redundant but not 3k-redundant for α_1 and $(E \to C, \beta_1)$.

A q-schema is in ℓ-Redundancy Free Normal Form for a given q-degree iff it does not admit any q-relations in which any ℓ-redundancy for the q-degree can ever occur.

Definition 3: (R^k, Σ) is in *ℓ-Redundancy Free Normal Form* for α (*ℓ-RFNF*) iff there are no q-relation r over R^k that satisfies Σ, no $\sigma \in \Sigma^+_{\mathfrak{C}}$, and no attribute $A \in R$ for which there is a data value $v \in r(A)$ that is ℓ-redundant for α and σ. If (R^k, Σ) is in *ℓ-RFNF* for α, then ℓ is a *level of α-data redundancy that (R^k, Σ) prevents*. Otherwise, (R^k, Σ) *permits this level*. If there is no positive integer ℓ for which (R^k, Σ) is in *ℓ-RFNF* for α, then (R^k, Σ) permits a level of α-data redundancy that is a priori unbounded. In this case, $\ell := \infty$.

For examples, the smallest level of $\alpha_1(\alpha_2$, respectively)-data redundancy that (HAP^2, Σ) prevents is $5k$ ($3k$, respectively).

Note that results from [4] are captured for $k = 1$. Using β-cuts we can show that a q-schema is in ℓ-RFNF for q-degree α_i iff the schema is in ℓ-RFNF for the β-cut of the dual c-degree.

Proposition 2: Let R^k denote a q-schema, Σ a set of qCCs and qFDs over R^k, and α_i some q-degree of R^k. Then (R^k, Σ) is in ℓ-RFNF for α_i iff $(R, \Sigma_{\beta_{k+1-i}})$ is in ℓ-RFNF.

B. Levels of α-update inefficiency

Preventing ℓ-redundant data value occurrences for some q-degree α is synonymous with maintaining data consistency by level-ℓ α-update inefficiency. Indeed, preventing ℓ-redundant data value occurrences for tuples of q-degree α or higher means there is some (α, ℓ)-transaction that can consistently update the current q-relation. Take the world r_1 in Table V. Updating all 3k occurrences of *Kilo* consistently to a new value in r_1 will satisfy $E \to C$. Based on *card*$(E) \leq 3k$ in r_1, we require updates of at most 3k tuples to maintain consistency for $E \to C$ in r_1, in any q-relation r that satisfies the given constraints. Hence, we say (R^k, Σ) is in *ℓ-Update Inefficiency Normal Form* (UINF) for α iff (R^k, Σ) is in ℓ-RFNF for α.

If (R^k, Σ) is in *ℓ-UINF* for α, then ℓ is a *level of α-update inefficiency that is sufficient for (R^k, Σ) to maintain data consistency*. Furthermore, if (R^k, Σ) is in *ℓ-UINF* for α for the smallest ℓ possible, then ℓ is the level of α-update inefficiency that is required for (R, Σ) to maintain data consistency. If there is no positive integer ℓ for which (R, Σ) is in *ℓ-UINF* for α, the level of α-update inefficiency required for (R, Σ) to maintain data consistency is a priori unbounded. In this case, $\ell := \infty$.

Proposition 3: Let R^k denote a q-schema, Σ a set of qCCs and qFDs over R^k, and α_i some q-degree of R^k. Then (R^k, Σ) is in ℓ-UINF for α_i iff $(R, \Sigma_{\beta_{k+1-i}})$ is in ℓ-UINF.

C. Levels of α-join efficiency

We will see now that the minimum level ℓ of α-redundancy we can prevent also quantifies the maximum level of α-join efficiency we can attain by any q-relation that satisfies the given constraints over a given q-schema. Intuitively, we define α-join efficiency as the maximum number of tuples with q-degree α or higher that have matching values on the LHS of a non-trivial qFD of degree β_{k+1-i} or higher.

Formally, if r satisfies the non-trivial qFD $(X \to Y, \beta_{k+1-i})$ over q-schema R^k, then r_i is the lossless join of its projections on XY and $X(R-Y)$. That is, $r_i = r_i(XY) \bowtie r_i(X(R-Y))$. Intuitively, the X-value of any tuple t in r_i joins the value $t(Y)$ with the $R - Y$-value of all the tuples $t_1, \ldots, t_\ell$ in r_i that have matching values with t on X. This number ℓ denotes the α_i-join strength of t, given $(X \to Y, \beta_{k+1-i})$.

Definition 4: Let r denote a q-relation that satisfies the set Σ of qCCs and qFDs over q-schema R^k, and let α_i denote a q-degree of R^k. The α_i-*join strength* of a tuple t in r_i with respect to a non-trivial qFD $\sigma = (X \rightarrow Y, \beta) \in \Sigma_{\mathfrak{C}}^{+}$ is the positive integer $\ell_{t,r}^{\sigma}$ that denotes the number of tuples in r_i that have matching values with t on all the attributes in X, that is, $\ell_{t,r}^{\sigma} = |\{t' \in r_i \mid t'(X) = t(X)\}|$.

Let $\Sigma = \{(CT \rightarrow E, \beta_1), (E \rightarrow C, \beta_1), (card(E) \leq 5k, \beta_1), (card(E) \leq 3k, \beta_2)\}$ denote a set of qCCs and qFDs over q-schema $\{E, C, T\}^2$. The q-relation r in Table IV satisfies Σ, in particular $(E \rightarrow C, \beta_1)$. As illustrated in Table Vb, r_2 is therefore its lossless join over its projections $r_2(EC)$ and $r_2(ET)$. Evidently in Table Vb, there are 5k tuples in r each of which has an α_2-join strength of $5k$ with respect to $(E \rightarrow C, \beta_1)$ since r_2 joins the redundant C-value *Kilo* with the $5k$ different values $t_1, \ldots, t_{5k}$.

Definition 5: For q-schema (R^k, Σ) and q-degree α over R^k, (R^k, Σ) is in ℓ-*Join Efficiency Normal Form* (JENF) for α iff there are some q-relation r over R^k that satisfies Σ, some non-trivial qFD $\sigma \in \Sigma_{\mathfrak{C}}^{+}$, and some tuple $t \in r_\alpha$ such that the α-join strength of t in r_α with respect to σ is ℓ. If (R^k, Σ) is in ℓ-*JENF* for α, then ℓ is a *level of* α-*join efficiency that* (R^k, Σ) *permits*. Otherwise, ℓ is a *level of* α-*join efficiency that* (R^k, Σ) *prevents*. If there is no positive integer ℓ for which (R^k, Σ) is in ℓ-*JENF* for α, then the level of α-join efficiency that (R^k, Σ) permits is a priori unbounded. Then, $\ell := \infty$.

In our example, (ECT^2, Σ) is in ℓ-JENF for α_1 iff $\ell \leq 3k$, and in ℓ-JENF for α_2 iff $\ell \leq 5k$. Hence, (ECT^2, Σ) permits a level of α_1-join efficiency up to $3k$. Similarly, (ECT^2, Σ) prevents any level of α_2-join efficiency above $5k$.

Proposition 4: Let R^k denote a q-schema, Σ a set of qCCs and qFDs over R^k, and α_i some q-degree of R^k. Then (R^k, Σ) is in ℓ-JENF for α_i if and only if $(R, \Sigma_{\beta_{k+1-i}})$ is in ℓ-JENF.

D. Design Desiderata

We will now be able to show what q-schemata in ℓ-BCNF actually achieve: i) the minimum effort required to maintain data consistency and ii) the maximum opportunity for joining data values, for a given q-degree.

First, we establish the equivalence between ℓ-BCNF for β_i and ℓ-RFNF (ℓ-UINF) for α_{k+1-i}, for every ℓ.

Theorem 5: For all (R^k, Σ), c-degree β_i of R^k, and all $\ell \in \mathbb{N}_{\geq 1}^{\infty}$, the following are equivalent:

1) (R^k, Σ) is in ℓ-RFNF (ℓ-UINF) for α_{k+1-i}
2) (R^k, Σ) is in ℓ-BCNF for β_i.

Proof: (R^k, Σ) is in ℓ-RFNF (ℓ-UINF) for α_{k+1-i} if and only if (R, Σ_{β_i}) is in ℓ-RFNF iff (R, Σ_{β_i}) is in ℓ-BCNF iff (R^k, Σ) is in ℓ-BCNF for β_i. ■

Our framework can quantify the trade-off between update and join efficiency for every q-schema, tailored to the q-degree, or c-degree dually, targeted for an application.

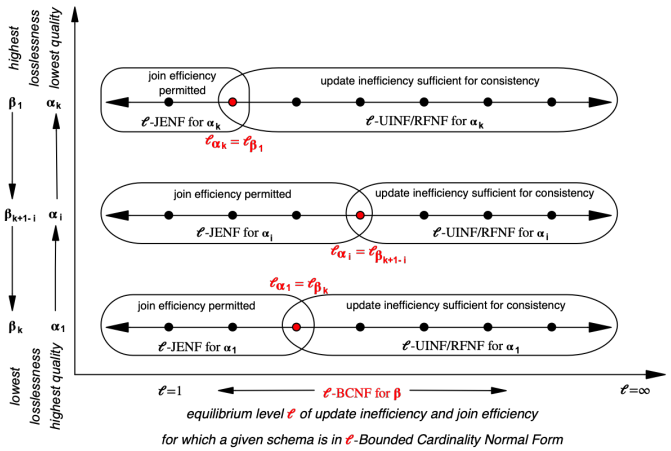


Fig. 1: Equilibrium Level ℓ_{α_i} that captures both the smallest level of α_i-update inefficiency sufficient to maintain data consistency and the largest level of α_i-join efficiency attainable, which is syntactically characterized by the smallest level for which a given schema is in ℓ-BCNF for c-degree β_{k+1-i}

Theorem 6: For all (R^k, Σ) the minimum level of α-update inefficiency that is sufficient for maintaining data consistency is the maximum level of α-join efficiency that it permits.

As a consequence of Theorem 5 and Theorem 6, every q-schema exhibits for every q-degree α, a unique level ℓ that quantifies the equilibrium between the smallest level of α-update inefficiency sufficient for data consistency maintenance and the largest level of α-join efficiency that is attainable.

Corollary 2: For every q-schema (R^k, Σ) and every q-degree α_i over R^k, there is a unique level $\ell \in \mathbb{N}_{\geq 1}^{\infty}$ such that (R^k, Σ) is in i) ℓ-BCNF for β_{k+1-i}, ii) ℓ-RFNF (UINF) for α_i, and iii) ℓ-JENF for α_i.

For each q-degree α_i, we call the unique level ℓ_{α_i} in Corollary 2 *the equilibrium between* α_i*-update inefficiency and* α_i*-join efficiency exhibited by q-schema* (R^k, Σ). The achievement of the equilibrium level are illustrated in Figure 1.

Our framework combines two control mechanisms: 1) The duality between q- and c-degrees captures trade-offs between data losslessness and data integrity that is targeted with the normalization. Indeed, lower q-degrees mean more data is normalized as $r_{\alpha_1} \subseteq \cdots \subseteq r_{\alpha_i} \subseteq \cdots \subseteq r_{\alpha_k}$. Lower q-degrees also mean fewer constraints are used for normalization since $\Sigma_{\beta_1} \subseteq \cdots \subseteq \Sigma_{\beta_i} \subseteq \cdots \subseteq \Sigma_{\beta_k}$. 2) The smallest level ℓ for which a given q-schema is in ℓ-BCNF for a given c-degree β_{k+1-i} captures the trade-off between the worst-case α_i-update inefficiency required and the best-case α_i-join efficiency attainable.

VII. Normalization

We establish algorithms that tailor normalization to requirements of applications on data dimensions. We first describe our strategy of normalization and its impact on current design practice. Subsequently, we address decompositions into variants of ℓ-BCNF.

A. Strategy and Impact

The input to the database normalization process is a p-schema R^k together with a set Σ of qCCs and qFDs. As in the classical design process, data is not required for the process. In particular, we do not require any q-relations as part of the input. However, having data available is likely to benefit the design process as the data can help designers identify those qCCs and qFDs that are meaningful for the domain of their applications. As in classical design, we assume that the input Σ to the database normalization process consists of those qCCs and qFDs that are meaningful for the given domain.

Indeed, the availability of c-degrees as part of the input provides us with many choices for database normalization. The choice we make is determined by the requirements for data quality. In fact, based on the requirements that applications have on the q-degrees of tuples or the c-degrees of constraints, we fix the appropriate c-degree β that determines which possible world we classically normalize with respect to the set of classical CCs and FDs that apply to it. Note that the duality between q-degrees and c-degrees allows us to fix the dual c-degree, even if the data quality requirements are only available in terms of the q-degrees.

Once we have fixed the compliance degree β, we can simply perform classical normalization with respect to the set of classical CCs and FDs whose c-degrees are at least as high as the target c-degree, that is, with respect to the β-cut Σ_β.

Indeed, we pursue normalizations into ℓ-Bounded Cardinality Normal Form for β_i which attain the smallest ℓ possible among all decompositions that are α_{k+1-i} lossless and β_i-FD preserving. By design, such schemata do only permit relations with worst-case ℓ-update inefficiency and best-case ℓ join efficiency for any records associated with q-degree α_{k+1-i} or higher.

Different choices of c-degrees address different data quality requirements. Hence, the availability of the c-degrees provides organizations with a variety of normalized database schemata. In this sense, the degree of compliance is a parameter that allows stakeholders to control trade-offs between data quality and data losslessness, and level ℓ quantifies the trade-off between update and join efficiency, as illustrated by Figure 1.

Relational normalization appears as a special case of this process where only two c-degrees are available, namely the top and the bottom c-degree. Here, the only CC and FD set we can use to perform classical normalization results from qCCs and qFDs that have the top c-degree. If more c-degrees are available for a domain, then different requirements for data quality in this domain identify different possible worlds and therefore different sets of cardinality constraints and functional dependencies and tuples they apply to.

Finally, we stress that the instances over the output of the normalization process are classical relations. In particular, the tuples do not carry any q-degrees. By the results of the previous section, the relations exhibit worst-case ℓ-update inefficiency and best-case ℓ-join efficiency. Indeed, the underlying application, according to its requirements, only considers tuples that meet the q-degree threshold, and for such tuples, the level ℓ quantifies the worst effort necessary to maintain data consistency as well as the best join support attainable.

B. Lossless, FD-preserving Decompositions

A *decomposition* of relation schema R is a set $\mathcal{D}$ of relation schemata such that $\bigcup_{S\in\mathcal{D}} S = R$. A decomposition $\mathcal{D}$ of R with CC/FD set Σ is *lossless* if for every relation r over R that satisfies Σ, $r = \bowtie_{S\in\mathcal{D}} r[S]$. Here, $r[S] = \{t(S) \mid t \in r\}$. An *$\ell$-BCNF decomposition* of R with CC/FD set Σ is a decomposition $\mathcal{D}$ of R where for every $S \in \mathcal{D}$, $(S, \Sigma[S])$ is in ℓ-BCNF. A decomposition $\mathcal{D}$ of (R, Σ) is *FD-preserving* if and only if for all FDs $\sigma \in \Sigma$ $\left(\bigcup_{S\in\mathcal{D}} \Sigma[S] \models \sigma\right)$.

Definition 6: An decomposition of a q-schema (R^k, Σ) for β is a decomposition of (R, Σ_β). A decomposition of some q-schema (R^k, Σ) for β_i is α_{k+1-i}-lossless (β_i-FD preserving, respectively) if the decomposition of (R, Σ_{β_i}) is lossless (FD-preserving, respectively).

Decompositions may suffer from FDs that are lost or not transformed into keys. Indeed, decompositions into Boyce-Codd Normal Form are not always FD-preserving, and decompositions into 3NF do not always transform all FDs into keys. However, lost FDs and non-key FDs affect update inefficiency. Hence, it becomes necessary to quantify the levels of update inefficiency and join efficiency that decompositions actually achieve, based on a given degree of compliance.

C. Quantifying Workload Efficiency

We assume we are given some decomposition $\mathcal{D}$ that results from a q-schema (R^k, Σ) and some c-degree β. We then assess the quality of $\mathcal{D}$ for both updates and joins as follows.

Join efficiency. The join efficiency of $\mathcal{D}$ results from only those FDs in Σ_β preserved by $\mathcal{D}$. These may have been transformed into keys or not. We define the set of *β-join-supportive attribute subsets* as

$$JS_{\mathcal{D}}^{(R,\Sigma_\beta)} = \{S : X_k \mid \exists S \in \mathcal{D} \exists X \to Y \in \Sigma_\beta[S], \Sigma_\beta \models X \to S\} \cup \{S : X \mid \exists S \in \mathcal{D} \exists X \to Y \in \Sigma_\beta[S], \Sigma_\beta \not\models X \to S\}.$$

Next, the β-join strength of a β-join-supportive attribute set is given by the minimal upper bounds for any CC that applies to it. Hence, for (R, Σ_β) and $X \subseteq R$,

$$\ell_X^\beta := \min\{\ell \in \mathbb{N}_{\geq 1}^\infty \mid \Sigma_\beta \models \mathit{card}(X) \leq \ell\}$$

is the minimum level of data redundancy for X. Now we define the level of β-join efficiency for $\mathcal{D}$ as the maximum among all minimum levels of data redundancy for a join-supportive attribute set, that is,

$$\ell_{\mathcal{D}}^{J,\beta} := \max\{\ell_X^\beta \mid S : X \in JS_{\mathcal{D}}^{(R,\Sigma_\beta)}\} \cup \{1 \mid S : X_k \in JS_{\mathcal{D}}^{(R,\Sigma_\beta)}\}.$$

In essence, FDs $X \to Y$ transformed into keys X contribute level 1, and other FDs contribute level ℓ_X^β. Note that $\ell_{\mathcal{D}}^{J,\beta}$ is 1 when Σ_β contains no FDs.

Update inefficiency. The β-update inefficiency of $\mathcal{D}$ is determined by all FDs of the input Σ_β. In particular, any

lost FD will need to be enforced by joining elements of $\mathcal{D}$ whenever updates occur. Hence, we define the set of *β-update-critical attribute subsets* as

$$UC_{\mathcal{D}}^{(R,\Sigma_\beta)} = JS_{\mathcal{D}}^{(R,\Sigma_\beta)} \cup \{R : X \mid \exists X \to Y \in (\Sigma_\beta - (\cup_{S \in \mathcal{D}} \Sigma_\beta[S])^+)\}.$$

Now we define the level of β-update inefficiency for $\mathcal{D}$ as the maximum among all minimum levels of data redundancy for a β-update-critical attribute set, that is,

$$\ell_{\mathcal{D}}^{U,\beta} := \max\{\ell_X^\beta \mid S : X \in UC_{\mathcal{D}}^{(R,\Sigma_\beta)}\} \cup \{1 \mid S : X_k \in UC_{\mathcal{D}}^{(R,\Sigma_\beta)}\}.$$

The ideas enable us to quantify the quality of schema decompositions in terms of update inefficiency and join efficiency, for any given target c-degree or, dually, q-degree.

Definition 7: The level of β-update inefficiency for decomposition $\mathcal{D}$ of a schema (R^k, Σ) is $\ell_{\mathcal{D}}^{U,\beta}$. The level of β-join efficiency for decomposition $\mathcal{D}$ of a schema (R^k, Σ) is $\ell_{\mathcal{D}}^{J,\beta}$.

Notably, whenever a decomposition $\mathcal{D}$ of a schema (R^k, Σ) for β is FD-preserving, then $\ell_{\mathcal{D}}^{U,\beta} = \ell_{\mathcal{D}}^{J,\beta}$.

As an example, consider the α_2-lossless, β_1-FD preserving decomposition $\mathcal{D}_g$ with elements $(R_1 = EC, \Sigma[R_1] = \{E \to C\})$ and $(R_3 = ETV, \Sigma[R_3] = \{TV \to E, ET \to V\})$. Here, we lost the FDs $V \to C$ and $CT \to V$. Since the β_2-join-supportive attribute sets are E, TV, and ET and their β_2-join strengths are all 1, we have $\ell_{\mathcal{D}_g}^{J,\beta_2} = 1$. Due to the lost FDs, we have the β_2-update critical attribute subsets of V and CT, and since $\ell_V^{\beta_2} = 300$ and $\ell_{CT}^{\beta_2} = 1$, we have $\ell_{\mathcal{D}_g}^{U,\beta_2} = 300$.

We will now present three algorithms for customizing schema normalization to data quality requirements.

D. Optimal FD-preserving Decompositions

First, we take FD-preservation as an imperative and derive a method for computing a schema decomposition in ℓ-BCNF with the minimum ℓ attainable. Given the results of our normalization framework, we can effectively apply the algorithm OPT from [4] to the relation schema of the given schema and the β_i-cut of the input set of qCCs and qFDs.

PROBLEM:	Optimal ℓ-BCNF decomposition
INPUT:	q-schema R^k set Σ of qCCs and qFDs over R^k c-degree β_i over R^k
OUTPUT:	an α_{k+1-i}-lossless, β_i-FD-preserving ℓ-BCNF decompososition of (R^k, Σ) with a minimum ℓ possible
METHOD:	OPT(R, Σ_{β_i}) [4]

The strong advantages of this approach include full control in terms of requiring the minimum level of effort in maintaining consistency for the target q-degree of data. All FDs that meet the target c-degree can be enforced locally on schemata of the decomposition.

E. Greedy Decomposition with Few Schemata

The previous approach may require a high number of schemata in the decomposition to guarantee FD-preservation. Sometimes, it may not be necessary to preserve all FDs that apply to the target c-degree. For example, most update operations may not have any effect on the validity of an FD. As an alternative, one may consider the following greedy approach, in which a set of FDs that apply to the target c-degree are prioritized based on the level of data redundancy they may cause.

PROBLEM:	$\ell_{\mathcal{D}}^{U,\beta}$-BCNF decomposition
INPUT:	q-schema R^k set Σ of qCCs and qFDs over R^k c-degree β_i over R^k
OUTPUT:	an α_{k+1-i}-lossless $\ell_{\mathcal{D}}^{U,\beta}$-BCNF decompososition $\mathcal{D}$ of (R^k, Σ)
METHOD:	GREED(R, Σ_{β_i}) [4]

The advantage of this approach is the number of elements in the decomposition. However, no guarantee is possible for the level of update inefficiency that can actually be achieved. The reason is that FDs may well be lost, and therefore need to be enforced on the join of elements of the decomposition. Note that the level $\ell_{\mathcal{D}}^{U,\beta}$ simply denotes the level of update inefficiency of the decomposition, taking into account any lost and non-key FDs that apply to the target c-degree.

F. Hybrid Decomposition

Yet another approach is to combine OPT and GREED to a hybrid strategy. First, we can apply OPT to obtain a schema in optimal ℓ-BCNF among FD-preserving decompositions for the target c-degree. We may then sacrifice the preservation of selected FDs for the ability to lower the minimum ℓ further, using decompositions in GREED [4].

PROBLEM:	$\ell_{\mathcal{D}}^{U,\beta}$-BCNF decomposition
INPUT:	q-schema R^k set Σ of qCCs and qFDs over R^k c-degree β_i over R^k
OUTPUT:	an α_{k+1-i}-lossless $\ell_{\mathcal{D}}^{U,\beta}$-BCNF decompososition $\mathcal{D}$ of (R^k, Σ)
METHOD:	HYBRID(R, Σ_{β_i}) [4]

The advantage of this approach is an even lower level of update inefficiency for the target c-degree. However, this comes at the cost of losing some FDs and an even larger number of schemata in the final decomposition.

G. Materialized Views

Once a database becomes operational, frequent access patterns emerge over time. Adding materialized views can help accelerate queries but will occupy additional storage space and time to maintain data consistency, based on any target q-degree and its dual c-degree. Hence, the definition and selection of views should be informed by quantifying the support of joins and the costs associated with maintaining data consistency. We define the (total) level of join efficiency and update inefficiency of a given view $\mathcal{V}$ (namely, a set of attributes) as $\ell_{\mathcal{V}}^{J,\beta(,\text{ total})}$ and $\ell_{\mathcal{V}}^{U,\beta(,\text{ total})}$, respectively.

For our greedy schema $\mathcal{D}_g$ as example, a frequent query may ask at what times companies work on a venue for

TABLE VI: Schema Designs for Running Example

Method	target c-degree	dual q-degree	*Schema*	*Lost FDs*	ℓ^U	ℓ^J	Materialized View
OPT	β_2	α_1	$\mathcal{D}_1$: $R_1 = EC$ with $E \to C$ $R_3 = ETV$ with $TV \to E$, $ET \to V$ $R_5 = CTV$ with $CT \to V$, $V \to C$	none	300	300	$\mathcal{V}_{\mathcal{D}_1} = \pi_{ECT}(R_1 \bowtie R_3)$ $\ell_{\mathcal{V}}^{U,\beta_2} = 3k = \ell_{\mathcal{V}}^{J,\beta_2}$
OPT	β_1	α_2	$\mathcal{D}_2$: $R_2 = VC$ with $V \to C$ $R_3 = ETV$ with $TV \to E$, $ET \to V$ $R_4 = CET$ with $CT \to E$, $E \to C$	none	5k	5k	$\mathcal{V}_{\mathcal{D}_2} = \pi_{VCT}(R_2 \bowtie R_3)$ $\ell_{\mathcal{V}}^{U,\beta_1} = 5m = \ell_{\mathcal{V}}^{J,\beta_1}$
GREED	β_2	α_1	$\mathcal{D}_g$: $R_1 = EC$ with $E \to C$ $R_3 = ETV$ with $TV \to E$, $ET \to V$	$V \to C$ $CT \to V$	300	1	$\mathcal{V}_{\mathcal{D}_g} = \pi_{ECT}(R_1 \bowtie R_3)$ $\ell_{\mathcal{V}}^{U,\beta_2} = 3k = \ell_{\mathcal{V}}^{J,\beta_2}$
GREED	β_1	α_2	$\mathcal{D}'_g$: $R_2 = VC$ with $V \to C$ $R_3 = ETV$ with $TV \to E$, $ET \to V$	$E \to C$ $CT \to V$	5k	1	$\mathcal{V}_{\mathcal{D}'_g} = \pi_{VCT}(R_2 \bowtie R_3)$ $\ell_{\mathcal{V}}^{U,\beta_1} = 5m = \ell_{\mathcal{V}}^{J,\beta_1}$
HYBRID	β_2	α_1	$\mathcal{D}_h$: $R_1 = EC$ with $E \to C$ $R_2 = CV$ with $V \to C$ $R_3 = ETV$ with $TV \to E$, $ET \to V$	$CT \to V$	1	1	$\mathcal{V}_{\mathcal{D}_h} = \pi_{ECT}(R_2 \bowtie R_3)$ $\ell_{\mathcal{V}}^{U,\beta_2} = 3k = \ell_{\mathcal{V}}^{J,\beta_2}$
HYBRID	β_1	α_2	$\mathcal{D}'_h$: $R_1 = EC$ with $E \to C$ $R_2 = CV$ with $V \to C$ $R_3 = ETV$ with $TV \to E$, $ET \to V$	$CT \to V$	1	1	$\mathcal{V}_{\mathcal{D}'_h} = \pi_{VCT}(R_2 \bowtie R_3)$ $\ell_{\mathcal{V}}^{U,\beta_1} = 5m = \ell_{\mathcal{V}}^{J,\beta_1}$

some planned event. Hence, we may introduce the following materialized view $\mathcal{V}$.

CREATE MATERIALIZED VIEW $\mathcal{V}$ AS (
SELECT $R_1.V, R_1.C, R_3.T$ FROM R_1, R_3
WHERE $R_1.E = R_3.E$);

In this case, the lost FD $V \to C$ can directly be enforced on the view instead.

CREATE ASSERTION LostFD_V_to_C
CHECK(NOT EXISTS(SELECT V FROM $\mathcal{V}$
GROUP BY V HAVING COUNT(C)>1));

Due to $V \to C$, $CT \to V$ and $card(V) \leq 300$, the view has level $\ell_{\mathcal{V}}^{J,\beta_2} = 300 = \ell_{\mathcal{V}}^{U,\beta_2}$ update inefficiency and join efficiency, respectively. Join support is effective, but requires propagation of a C-update on base table R_1 to up to 300 updates on $\mathcal{V}$. Hence, the equilibrium we quantify between the smallest level of update inefficiency and the largest level of join efficiency is also intrinsic to materialized views, and should inform their definition and selection.

VIII. EXPERIMENTS

We apply our framework to the running example. Firstly, we apply normalization algorithms to obtain schema designs tailored to the target c-degree. We also include view definitions in each case. We list the levels of update inefficiency and join efficiency achieved by each design, both for the schema decomposition and the views. Secondly, we report on the run times of updates and joins on each design, with and without views, and compare trade-offs. Finally, we show how well the properties of logical schema designs predict the physical performance of updates and joins once the database is operational. We implemented algorithms in Python. We ran operations on an Intel Xeon W-2123, 3.6 GHz, 256GB, Windows 10 PC, 2017 SQL Server Community Edition.

A. Schema Designs at Logical Level

Starting the with schema (HAP^2, Σ_0), we applied each of the Algorithms OPT, GREED, and HYBRID to the schema and each of the c-degrees β_1 and β_2, respectively. The resulting relational database schemata are listed in Table VII, together with any FDs that were lost, as well as the levels of update inefficiency and join efficiency for the decompositions, and some materialized view definition for each the designs (and their levels of incremental update maintenance and join support). Notably, the trade-offs mentioned before are already visible on the examples. OPT minimizes the level of update inefficiency under FD-preservation, while GREED has fewer schemata at the cost of losing two FDs. HYBRID minimizes the level of update inefficiency further by losing one FD. We stress the impact of target c-degrees, and their dual q-degrees, on the results. In particular, OPT results in significantly different schemata based on minimizing the levels of update inefficiency, which are different due to the bounds of cardinality constraints that hold with different c-degrees.

B. Schema Designs at Physical Level

We compare update and join operations on projections of the relation r from Table I onto schema designs from Table VI. To illustrate the impact of FDs on updates and joins, the numbers of α-redundant values in r coincide with their levels of α-data redundancy on the schema. For the experiments, we consider 4 operations that are affected by our qCCs and qFDs:

- Update u_1 replaces all occurrences of a C-value associated with a given E-value,
- Update u_2 replaces all occurrences of a C-value associated with a given V-value,
- Query q_1 returns all CTE combinations, and
- Query q_2 returns all CTV combinations.

Each operation is run 100 times on each of the schema designs in Table VI, and the average value reported.

For u_1, $3k$ occurrences of an α_1-redundant C-value are updated, and $5k$ occurrences of an α_2-redundant C-value. For u_2, 300 occurrences of an α_1-redundant C-value are updated, and $5m$ occurrences of an α_2-redundant C-value. Updates on base tables are propagated to materialized views, if present. As $V \to C$ is lost on $\mathcal{D}_g$, u_2 updates C-values on $R_1 \bowtie R_3$

TABLE VII: Queries q_1 and q_2 on Schema Designs for Running Example

q	$\mathcal{D}_1$	$\mathcal{D}_1 + \mathcal{V}_{\mathcal{D}_1}$
q_1	$\pi_{ECT}(R_1 \bowtie R_3)$	$\mathcal{V}_{\mathcal{D}_1}$
q_2	R_5	R_5

(a) Decomposition $\mathcal{D}_1$

q	$\mathcal{D}_2$	$\mathcal{D}_2 + \mathcal{V}_{\mathcal{D}_2}$
q_1	R_4	R_4
q_2	$\pi_{VCT}(R_2 \bowtie R_3)$	$\mathcal{V}_{\mathcal{D}_2}$

(b) Decomposition $\mathcal{D}_2$

q	$\mathcal{D}_g$	$\mathcal{D}_g + \mathcal{V}_{\mathcal{D}_g}$
q_1	$\pi_{ECT}(R_1 \bowtie R_3)$	$\mathcal{V}_{\mathcal{D}_g}$
q_2	$\pi_{VCT}(R_1 \bowtie R_3)$	$\pi_{VCT}(R_1 \bowtie R_3)$

(c) Decomposition $\mathcal{D}_g$

q	$\mathcal{D}'_g$	$\mathcal{D}'_g + \mathcal{V}_{\mathcal{D}'_g}$
q_1	$\pi_{ECT}(R_2 \bowtie R_3)$	$\pi_{ECT}(R_2 \bowtie R_3)$
q_2	$\pi_{VCT}(R_2 \bowtie R_3)$	$\mathcal{V}_{\mathcal{D}'_g}$

(d) Decomposition $\mathcal{D}'_g$

q	$\mathcal{D}_h$	$\mathcal{D}_h + \mathcal{V}_{\mathcal{D}_h}$
q_1	$\pi_{ECT}(R_2 \bowtie R_3)$	$\mathcal{V}_{\mathcal{D}_h}$
q_2	$\pi_{VCT}(R_2 \bowtie R_3)$	$\pi_{VCT}(R_2 \bowtie R_3)$

(e) Decomposition $\mathcal{D}_h$

q	$\mathcal{D}'_h$	$\mathcal{D}'_h + \mathcal{V}_{\mathcal{D}'_h}$
q_1	$\pi_{ECT}(R_2 \bowtie R_3)$	$\pi_{ECT}(R_2 \bowtie R_3)$
q_2	$\pi_{VCT}(R_2 \bowtie R_3)$	$\mathcal{V}_{\mathcal{D}'_h}$

(f) Decomposition $\mathcal{D}'_h$

and projects them onto R_1. As $E \rightarrow C$ is lost on $\mathcal{D}'_g$, u_1 updates C-values on $R_2 \bowtie R_3$ and projects them onto R_2.

Table VII shows how the queries q_1 and q_2 are implemented on each of our schemata from Table VI, without and with materialized views. On the relation over 3NF schema HAP, q_1 and q_2 are simple projections and do not require joins.

Table VIII shows median times (in seconds) for 25 runs of each operation on relation r projected onto schemata of Table VI. Runtime in bold font for a schema design without view means there was some gain over the runtime on HAP for the same operation. However, runtime in bold font on a schema design with view means there was some gain for a join operation while there was a loss for an update. For example, u_2 and q_1 enjoy fast runtimes on $\mathcal{D}_2$, but while adding the view to $\mathcal{D}_2$ speeds up q_2 it significantly slows down u_2. Comparing runtimes for the same operation on the same schema quantifies the variation in levels of update inefficiency and query efficiency across the different bounds of the CCs that hold for different target c-degrees.

Figures 2 and 3 rank schema designs from fastest (left) to slowest (right) for a given operation. Bars are color-coded to group them. Groups have different levels of performance, and designs in the same group may not be distinguished further.

There are several key observations: 1) It is evident that different operations are best supported by different designs. 2) The same schema design is ranked the same for any given operation, independently of the possible world to which the operation is applied. This is evident by comparing ranks on the left- and right-hand side of the figures. 3) The runtime efficiency of each operation is proportional to the level of α-redundancy caused by an FD that affects the operation. 4) This behavior translates to materialized views, as well. In particular, a gain in join efficiency by a view is traded in for a proportional loss in update efficiency.

C. Predicting Performance at the Logical Level

We illustrate how the levels of update inefficiency and join efficiency at logical schema design time predict the physical performance of operations on actual instances of the schemata.

The color-coded groups in Figure 2 and Figure 3 can be predicted due to the levels of α-data redundancy caused by FDs affecting the operation. Tables IX and X list the levels of α-data redundancy caused by the FDs that affect the operations.

TABLE VIII: Runtime of Operations (in s) on Schema Designs

	HAP	
op	α_1	α_2
u_1	0.1689	4.650
u_2	0.03000	226.830
q_1	0.02490	0.055
q_2	0.01680	16.950

(a) HAP

	$\mathcal{D}_1$		$\mathcal{D}_1 + \mathcal{V}_{\mathcal{D}_1}$		$\mathcal{D}_2$		$\mathcal{D}_2 + \mathcal{V}_{\mathcal{D}_2}$	
op	α_1	α_2	α_1	α_2	α_1	α_2	α_1	α_2
u_1	**0.0111**	**2.95**	**0.1821**	**5.15**	0.1674	4.550	0.1674	4.550
u_2	0.0294	224.05	0.0294	224.05	**0.0132**	**1.150**	**0.0381**	**228.750**
q_1	0.0333	0.70	**0.0318**	**0.06**	**0.0225**	**0.045**	0.0225	0.045
q_2	**0.0159**	**13.80**	0.0159	13.80	0.0435	20.450	**0.0171**	**17.350**

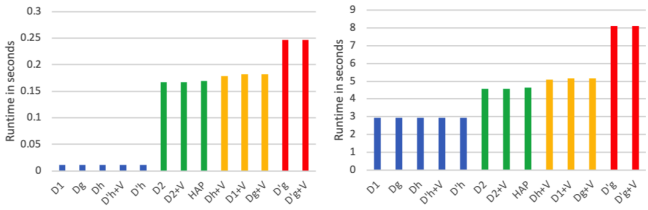
(b) Decompositions $\mathcal{D}_1$ and $\mathcal{D}_2$

	$\mathcal{D}_g$		$\mathcal{D}_g + \mathcal{V}_{\mathcal{D}_g}$		$\mathcal{D}'_g$		$\mathcal{D}'_g + \mathcal{V}_{\mathcal{D}'_g}$	
op	α_1	α_2	α_1	α_2	α_1	α_2	α_1	α_2
u_1	**0.0111**	**2.95**	**0.1821**	**5.15**	0.2469	8.10	0.2469	8.10
u_2	0.0567	419.10	0.0567	419.10	**0.0147**	**1.00**	**0.0381**	**228.75**
q_1	0.0333	0.70	**0.0318**	**0.06**	0.0348	0.85	0.0348	0.85
q_2	0.0423	21.60	0.0423	21.60	0.0435	20.45	**0.0159**	**17.35**

(c) Decompositions $\mathcal{D}_g$ and $\mathcal{D}'_g$

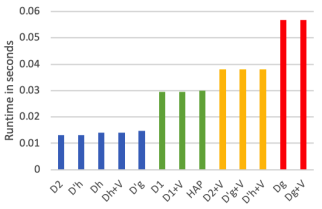
	$\mathcal{D}_h$		$\mathcal{D}_h + \mathcal{V}_{\mathcal{D}_h}$		$\mathcal{D}'_h$		$\mathcal{D}'_h + \mathcal{V}_{\mathcal{D}'_h}$	
op	α_1	α_2	α_1	α_2	α_1	α_2	α_1	α_2
u_1	**0.0111**	**2.95**	**0.1787**	**5.08**	**0.0111**	**2.95**	0.0111	2.95
u_2	**0.0141**	**1.05**	0.0141	1.050	**0.0132**	**1.15**	**0.0381**	**228.75**
q_1	0.0348	0.85	**0.0318**	**0.060**	0.0348	0.85	0.0348	0.85
q_2	0.0435	20.45	0.0435	20.45	0.0435	20.45	**0.0171**	**17.35**

(d) Decompositions $\mathcal{D}_h$ and $\mathcal{D}'_h$

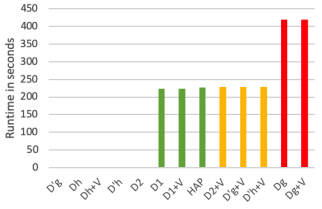


(a) Op u_1 on high-priority events

(b) Op u_1 on all events

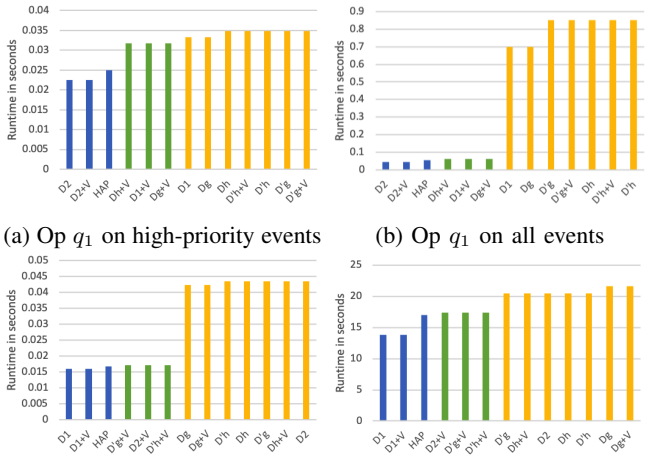


(c) Op u_2 on high-priority events



(d) Op u_2 on all events

Fig. 2: Ranking Schema Designs by Runtime of Updates



(a) Op q_1 on high-priority events (b) Op q_1 on all events

(c) Op q_2 on high-priority events (d) Op q_2 on all events

Fig. 3: Ranking Schema Designs by Runtime of Queries

TABLE IX: Predicting Update Performance at Design Time

(a) Rank Prediction of u_1

$\mathcal{D}$	ℓ_E^{U,α_1}	ℓ_E^{U,α_2}	color
$\mathcal{D}_1$	1	1	blue
$\mathcal{D}_g$	1	1	blue
$\mathcal{D}_h$	1	1	blue
$\mathcal{D}'_h$	1	1	blue
$\mathcal{D}'_h + V$	1	1	blue
$\mathcal{D}_2$	3k	5k	green
$\mathcal{D}_2 + V$	3k	5k	green
HAP	3k	5k	green
$\mathcal{D}_1 + V$	3k	5k	yellow
$\mathcal{D}_g + V$	3k	5k	yellow
$\mathcal{D}_h + V$	3k	5k	yellow
$\mathcal{D}'_g$	3k	5k	red
$\mathcal{D}'_g + V$	3k	5k	red

(b) Rank Prediction of u_2

$\mathcal{D}$	ℓ_V^{U,α_1}	ℓ_V^{U,α_2}	color
$\mathcal{D}_2$	1	1	blue
$\mathcal{D}'_g$	1	1	blue
$\mathcal{D}_h$	1	1	blue
$\mathcal{D}_h + V$	1	1	blue
$\mathcal{D}'_h$	1	1	blue
$\mathcal{D}_1$	300	5m	green
$\mathcal{D}_1 + V$	300	5m	green
HAP	300	5m	green
$\mathcal{D}_2 + V$	300	5m	yellow
$\mathcal{D}'_g + V$	300	5m	yellow
$\mathcal{D}'_h + V$	300	5m	yellow
$\mathcal{D}_g$	300	5m	red
$\mathcal{D}_g + V$	300	5m	red

TABLE X: Predicting Query Performance at Design Time

(a) Rank Prediction of q_1

$\mathcal{D}$	ℓ_E^{J,α_1}	ℓ_E^{J,α_2}	color
$\mathcal{D}_2$	3k	5k	blue
$\mathcal{D}_2 + V$	3k	5k	blue
HAP	3k	5k	blue
$\mathcal{D}_1 + V$	3k	5k	green
$\mathcal{D}_g + V$	3k	5k	green
$\mathcal{D}_h + V$	3k	5k	green
$\mathcal{D}_1$	1	1	yellow
$\mathcal{D}_g$	1	1	yellow
$\mathcal{D}'_g$	1	1	yellow
$\mathcal{D}'_g + V$	1	1	yellow
$\mathcal{D}_h$	1	1	yellow
$\mathcal{D}'_h$	1	1	yellow
$\mathcal{D}'_h + V$	1	1	yellow

(b) Rank Prediction of q_2

$\mathcal{D}$	ℓ_V^{J,α_1}	ℓ_V^{J,α_2}	color
$\mathcal{D}_1$	300	5m	blue
$\mathcal{D}_1 + V$	300	5m	blue
HAP	300	5m	blue
$\mathcal{D}_2 + V$	300	5m	green
$\mathcal{D}'_g + V$	300	5m	green
$\mathcal{D}'_h + V$	300	5m	green
$\mathcal{D}_2$	1	1	yellow
$\mathcal{D}_g$	1	1	yellow
$\mathcal{D}_g + V$	1	1	yellow
$\mathcal{D}'_g$	1	1	yellow
$\mathcal{D}_h$	1	1	yellow
$\mathcal{D}_h + V$	1	1	yellow
$\mathcal{D}'_h$	1	1	yellow

Operations u_1 and q_1 are affected by the FD $E \to C$, and operations u_2 and q_2 by FD $V \to C$. In Tables IX and X, the second and third columns list the level of α-redundancy caused by the FDs on the various schema designs. For updates, the lower the levels, the higher the rank. For queries, the higher the levels, the higher the rank. However, more can be predicted for designs where the level is different from 1.

For updates, we can distinguish between those that can be processed i) based on using keys (blue color), ii) locally on some schema of the design based on the non-key FD (green color), iii) on the view based on the non-key FD (yellow color), and iv) on the join of some schemata in the design due to a non-key FD (red). Using this grouping from Table IX at schema design time, provides a good prediction of the physical update performance illustrated in Figure 2.

For queries, we can distinguish between those that can be processed i) based on a non-key FD on the schema (blue color), ii) based on a non-key FD on the view (green color), and iii) based on the join of some schemata in the design (yellow). Using this grouping from Table X at schema design time, provides a good prediction of the physical join performance illustrated in Figure 3.

Of course, there would be other factors to consider, such as the projection onto attribute subsets which may require duplicate elimination. However, these details provide little insight into the aims of our normalization framework.

Hence, our framework can quantify update and join efficiency, even when tailored to records that meet requirements on their data quality. We can predict, at schema design time, which designs will best support which operations once the database is operational. This transcends into the definition and selection of materialized views.

IX. Related Work

Our work unifies different strands of previous work. The work in [4] introduced the Bounded Cardinality Normal Form without considering data quality dimensions, while [9] introduced a possibilistic approach to logical schema design. The work in [7] has established an axiomatization for qCCs and qFDs. Our work provides a clear extension for these results. In particular, we show how the quantification of update and join efficiency at schema design time can work for any combination of data quality dimensions, see Figure 1. In fact, our experiments confirm the ability to predict at schema design time how well different designs will support updates and joins once the database is operational.

For probabilistic databases [10], one may use sharp constraints that every possible world needs to satisfy, or probabilistic constraints that hold when their marginal probability meets a given threshold. Sharp constraints eliminate possible worlds in which the constraints do not hold, and thus condition probabilistic databases. Normalization theory may carry over from classical to probabilistic databases when sharp constraints are used [11]. For probabilistic constraints, normalization has not been investigated yet. However, the class of probabilistic FDs is not even finitely axiomatizable [7].

Schema design was studied in data models like SQL [12], nested relations [13], data warehouses [14], object-orientation [15], semantics [16], temporality [17], the Web [18], uncertainty [9], and graphs [19]. As for classical design, our results may be extended to richer data models and higher normal forms such as 4NF [20] or Inclusion-Dependency Normal Form [21].

Unlike BCNF [22], [23] and 3NF [24] which only use FDs, CCs measure the effort to keep data consistent. We compute designs that quantify worst-case update inefficiency and best-case join efficiency, including maintenance and join support by materialized views.

Numerical dependencies (NDs) apply classical normalization after horizontal decomposition into blocks where FDs hold [25]. An ND is an expression of the form $X \rightarrow_\ell Y$ expressing that every X-value in an underlying relation r can co-occur with at most ℓ different Y-values in r. That is, every X-value is associated with at most ℓ different Y-values. More formally, for all $t_1, \ldots, t_{\ell+1} \in r$ where $t_1(X) = \ldots = t_{\ell+1}(X)$, there are some $1 \leq i, j \leq \ell + 1$ such that $t_i(Y) = t_j(Y)$. Evidently, an FD $X \rightarrow Y$ is captured as the special case of an ND $X \rightarrow_\ell Y$ where $\ell = 1$, and a CC $\mathit{card}(X) \leq \ell$ is captured as the special case of an ND $X \rightarrow_\ell (R - X)$. Consequently, our condition of an ℓ-Bounded Cardinality Normal Form reads that for every $X \rightarrow_1 Y \in \Sigma^+$ where $Y \not\subseteq X$, we must have $X \rightarrow_\ell (R - X) \in \Sigma^+$. Unfortunately, however, the class of NDs is not finitely axiomatizable [26], so a more general approach via general NDs is not possible, at least not directly by using finite axiomatizations. Indeed, our framework only relies on the combined class of the two special cases of NDs which capture FDs and CCs, respectively. This class is axiomatized by $\mathfrak{L}$ [8].

Another attempt to quantify data redundancy in logical database design is based on information theory [27]. Here, the authors define the *guaranteed information content* of a schema (R, Σ) for an attribute $A \in R$ as the least amount of information content that may be found in A-columns of instances of R that satisfy Σ. This measure represents the worst case of redundancy in column A over all possible instances. The authors demonstrate that Third Normal Form incurs the least possible redundancy among all dependency-preserving decompositions of a schema. Hence, the guaranteed information content quantifies the sources of data redundancy based on functional dependencies alone, while our work quantifies the level of data redundancy that these sources cause using functional dependencies and cardinality constraints. As an illustration for the difference, the guaranteed information content for the attribute C is $1/2$ in both $\mathcal{D}_1$ and $\mathcal{D}_2$ and for all other attributes it is 1 in both decompositions, while the smallest level of α_1/α_2-data redundancy prevented by $\mathcal{D}_1$ is $300/5m$ and that prevented by $\mathcal{D}_2$ is $3k/5k$. Hence, based on the information-theoretic characterization of 3NF [27], $\mathcal{D}_1$ and $\mathcal{D}_2$ are indistinguishable. An interesting direction of future work is to develop information-theoretic characterizations of the Bounded Cardinality Normal Forms.

Schema design is not expected to optimize the layout for all instances. Instead, physical tuning kicks in after deploying the logical schema and once reliable data retrieval patterns emerge. These include virtual de-normalization in main memory databases [28], migration to NoSQL [29], and data warehouse designs [14]. Guidelines have been devised to recover 3NF from de-normalised schema [30], and de-normalizing normalised schemata [31]. Information-theoretic justifications exist for fact tables in snowflake schemata [32] and for 3NF [27]. Update inefficiency and join efficiency inform the difficult problem of selecting materialized views [33].

Kojić and Milićev de-normalize by maximizing read benefits while write costs remain under a threshold [34]. Their technique tunes a logical schema (for example, in 3NF) based on specific updates and queries. Hence, their approach is applicable once the database is operational and a mature workload model available. This additional input makes it possible to derive a schema optimized for the input workloadfor [34]. The work in [34] does not use CCs. Our approach is for logical schema normalization where a workload model in terms of frequent update and query operations is not available yet. We use CCs to explore various normal forms with different properties in terms of update inefficiency and join efficiency on the schemata. Hence, the design team can assess different designs based on their properties. We only require a set of attributes, FDs, and CCs as input. As our work brings forward a logical schema, it may provide a different starting point for applying the work in [34] than classical normalization does. For example, once frequent updates and queries have emerged, we may apply [34] to the output of our algorithms rather than the 3NF schema $(\textsc{Hap}, (\Sigma_0)_\beta)$.

Recent approaches to schema design and evolution for NoSQL databases [35] are driven by a query workload, and NoSQL schemata are thus de-normalized. We are unaware of work for logical NoSQL schema design for CCs and FDs.

"CCs are one of the most important kinds of constraint in conceptual modeling" [36]. "CCs correspond to very common semantic rules on relationships and their formal definition at the conceptual level improves significantly the completeness of data description" [37]. Surprisingly, CCs were only used recently for logical schema design [4]. CCs were introduced in Chen's seminal ER paper [16], and studied in models such as semantic [38], Web [39], spatial and temporal [40], and uncertain models [41]. They inform major languages for data modeling, including UML, EER, ORM, XSD, or OWL (such as `owl:maxCardinality`) [42]. They have been used in data cleaning [43], database testing [44], query answering [45] and reverse engineering [46].

Violations of dependencies by dirty data motivate relaxed notions such as approximate keys and FDs [47], [48]. When mining from (dirty) data, approximate notions may improve recall but worsen precision of identifying meaningful rules. The CC $\mathit{card}(X) \leq \ell$ may be seen as approximate key permitting up to ℓ duplicates. For small ℓ, we may thus view our work as extending classical schema design to approximate keys, offering some robustness for dirty data.

X. Conclusion and Future Work

Dimensions of data quality provide invaluable information for data management. We established the first framework that quantifies the support of logical schema design for updates and joins on data that meet a given threshold for data quality. In our work, quality degrees measure how well records meet requirements on data quality. Dually, degrees of compliance are applied to integrity constraints for quantifying how well such requirements need to be satisfied. For any degree of compliance required, we have established an infinite family of ℓ-Bounded Cardinality Normal Forms. Here, level ℓ represents the worst-case effort to maintain integrity for data that meets the degree, but also the best-case to join such data. Our experiments have validated the capability of predicting at schema design time how well updates and joins will perform once the database is operational.

We will extend our findings to different data models and other constraints that facilitate entity and referential integrity.

References

[1] S. W. Sadiq, T. Dasu, X. L. Dong, J. Freire, I. F. Ilyas, S. Link, R. J. Miller, F. Naumann, X. Zhou, and D. Srivastava, "Data quality: The role of empiricism," *SIGMOD Rec.*, vol. 46, no. 4, pp. 35–43, 2017.

[2] S. W. Sadiq, Ed., *Handbook of Data Quality, Research and Practice*. Springer, 2013.

[3] C. Batini and A. Maurino, "Design for data quality," in *Encyclopedia of Database Systems, Second Edition*, 2018.

[4] S. Link and Z. Wei, "Logical schema design that quantifies update inefficiency and join efficiency," in *SIGMOD*, 2021, pp. 1169–1181.

[5] C. Batini, C. Cappiello, C. Francalanci, and A. Maurino, "Methodologies for data quality assessment and improvement," *ACM Comput. Surv.*, vol. 41, no. 3, pp. 16:1–16:52, 2009.

[6] M. W. Vincent, "Semantic foundations of 4NF in relational database design," *Acta Inf.*, vol. 36, no. 3, pp. 173–213, 1999.

[7] T. Roblot and S. Link, "Cardinality constraints and functional dependencies over possibilistic data," *Data Knowl. Eng.*, vol. 117, pp. 339–358, 2018.

[8] S. Hartmann, "Reasoning about participation constraints and Chen's constraints," in *ADC*, 2003, pp. 105–113.

[9] S. Link and H. Prade, "Relational database schema design for uncertain data," *Inf. Syst.*, vol. 84, pp. 88–110, 2019.

[10] D. Suciu, D. Olteanu, C. Ré, and C. Koch, *Probabilistic Databases*, ser. Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2011.

[11] S. Hartmann and S. Link, "Normal forms and normalization for probabilistic databases under sharp constraints," ser. Frontiers in Artificial Intelligence and Applications, vol. 260, 2013, pp. 1–16.

[12] H. Köhler and S. Link, "SQL schema design: Foundations, normal forms, and normalization," in *SIGMOD*, 2016, pp. 267–279.

[13] W. Y. Mok, Y. Ng, and D. W. Embley, "A normal form for precisely characterizing redundancy in nested relations," *ACM Trans. Database Syst.*, vol. 21, no. 1, pp. 77–106, 1996.

[14] J. Lechtenbörger and G. Vossen, "Multidimensional normal forms for warehouse design," *Inf. Syst.*, vol. 28, no. 5, pp. 415–434, 2003.

[15] V. L. Khizder and G. E. Weddell, "Reasoning about uniqueness constraints in object relational databases," *IEEE Trans. Knowl. Data Eng.*, vol. 15, no. 5, pp. 1295–1306, 2003.

[16] P. Chen, "The entity-relationship model-toward a unified view of data," *ACM Trans. Database Syst.*, vol. 1, no. 1, pp. 9–36, 1976.

[17] C. S. Jensen, R. T. Snodgrass, and M. D. Soo, "Extending existing dependency theory to temporal databases," *IEEE Trans. Knowl. Data Eng.*, vol. 8, no. 4, pp. 563–582, 1996.

[18] M. Arenas, "Normalization theory for XML," *SIGMOD Record*, vol. 35, no. 4, pp. 57–64, 2006.

[20] R. Fagin, "Multivalued dependencies and a new normal form for relational databases," *ACM Trans. Database Syst.*, vol. 2, no. 3, pp. 262–278, 1977.

[19] W. Fan, Y. Wu, and J. Xu, "Functional dependencies for graphs," in *SIGMOD*, 2016, pp. 1843–1857.

[21] M. Levene and M. W. Vincent, "Justification for inclusion dependency normal form," *IEEE Trans. Knowl. Data Eng.*, vol. 12, no. 2, pp. 281–291, 2000.

[22] C. Beeri, P. A. Bernstein, and N. Goodman, "A sophisticate's introduction to database normalization theory," in *VLDB*, 1978, pp. 113–124.

[23] P. A. Bernstein and N. Goodman, "What does boyce-codd normal form do?" in *VLDB*, 1980, pp. 245–259.

[24] J. Biskup, U. Dayal, and P. A. Bernstein, "Synthesizing independent database schemas," in *SIGMOD*, 1979, pp. 143–151.

[25] J. Grant and J. Minker, "Normalization and axiomatization for numerical dependencies," *Inf. Control.*, vol. 65, no. 1, pp. 1–17, 1985.

[26] ——, "Inferences for numerical dependencies," *Theor. Comput. Sci.*, vol. 41, pp. 271–287, 1985.

[27] S. Kolahi and L. Libkin, "An information-theoretic analysis of worst-case redundancy in database design," *ACM Trans. Database Syst.*, vol. 35, no. 1, pp. 5:1–5:32, 2010.

[28] Z. Liu and S. Idreos, "Main memory adaptive denormalization," in *SIGMOD*, 2016, pp. 2253–2254.

[29] J. Yoo, K. Lee, and Y. Jeon, "Migration from RDBMS to nosql using column-level denormalization and atomic aggregates," *J. Inf. Sci. Eng.*, vol. 34, no. 1, pp. 243–259, 2018.

[30] J. Petit, F. Toumani, J. Boulicaut, and J. Kouloumdjian, "Towards the reverse engineering of denormalized relational databases," in *ICDE*, 1996, pp. 218–227.

[31] D. B. Bock and J. F. Schrage, "Denormalization guidelines for base and transaction tables," *ACM SIGCSE Bull.*, vol. 34, no. 4, pp. 129–133, 2002.

[32] M. Levene and G. Loizou, "Why is the snowflake schema a good data warehouse design?" *Inf. Syst.*, vol. 28, no. 3, pp. 225–240, 2003.

[33] R. Chirkova and J. Yang, "Materialized views," *Found. Trends Databases*, vol. 4, no. 4, pp. 295–405, 2012.

[34] N. Kojic and D. Milicev, "Equilibrium of redundancy in relational model for optimized data retrieval," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 9, pp. 1707–1721, 2020.

[35] S. Scherzinger and S. Sidortschuck, "An empirical study on the design and evolution of nosql database schemas," in *ER*, 2020, pp. 441–455.

[36] A. Olivé, "Cardinality constraints," in *Conceptual Modeling of Information Systems*. Springer, 2007, pp. 83–102.

[37] M. Lenzerini and G. Santucci, "Cardinality constraints in the entity-relationship model," in *ER*, 1983, pp. 529–549.

[38] S. W. Liddle, D. W. Embley, and S. N. Woodfield, "Cardinality constraints in semantic data models," *Data Knowl. Eng.*, vol. 11, no. 3, pp. 235–270, 1993.

[39] F. Ferrarotti, S. Hartmann, and S. Link, "Efficiency frontiers of XML cardinality constraints," *Data Knowl. Eng.*, vol. 87, pp. 297–319, 2013.

[40] F. Currim and S. Ram, "Conceptually modeling windows and bounds for space and time in database constraints," *Commun. ACM*, vol. 51, no. 11, pp. 125–129, 2008.

[41] T. Roblot, M. Hannula, and S. Link, "Probabilistic cardinality constraints," *VLDB J.*, vol. 27, no. 6, pp. 771–795, 2018.

[42] N. Hall, H. Köhler, S. Link, H. Prade, and X. Zhou, "Cardinality constraints on qualitatively uncertain data," *Data Knowl. Eng.*, vol. 99, pp. 126–150, 2015.

[43] W. Chen, W. Fan, and S. Ma, "Incorporating cardinality constraints and synonym rules into conditional functional dependencies," *Inf. Process. Lett.*, vol. 109, no. 14, pp. 783–789, 2009.

[44] N. Bruno, S. Chaudhuri, and D. Thomas, "Generating queries with cardinality constraints for DBMS testing," *IEEE Trans. Knowl. Data Eng.*, vol. 18, no. 12, pp. 1721–1725, 2006.

[45] G. Cormode, D. Srivastava, E. Shen, and T. Yu, "Aggregate query answering on possibilistic data with cardinality constraints," in *ICDE*, 2012, pp. 258–269.

[46] C. Soutou, "Relational database reverse engineering: Algorithms to extract cardinality constraints," *Data Knowl. Eng.*, vol. 28, no. 2, pp. 161–207, 1998.

[47] Y. Huhtala, J. Kärkkäinen, P. Porkka, and H. Toivonen, "TANE: an efficient algorithm for discovering functional and approximate dependencies," *Comput. J.*, vol. 42, no. 2, pp. 100–111, 1999.

[48] S. Kruse and F. Naumann, "Efficient discovery of approximate dependencies," *Proc. VLDB Endow.*, vol. 11, no. 7, pp. 759–772, 2018.