Probabilistic Keys

Pieta Brown and Sebastian Link
Department of Computer Science, The University of Auckland, New Zealand.
E-mail: [p.brown|s.link]@auckland.ac.nz

Abstract—Probabilistic databases address well the requirements of an increasing number of modern applications that produce large volumes of uncertain data from a variety of sources. Probabilistic keys enforce the integrity of entities in order to facilitate data processing in probabilistic database systems. For this purpose, we establish algorithms for an agile schema- and data-driven elicitation of the marginal probability by which keys should hold in a given application domain, and for reasoning about these keys. The efficiency of our elicitation framework is demonstrated theoretically and experimentally.

Index Terms—H.2.1.d) Database models; H.2.3.d) Database semantics; F4.3.d) Decision problems; I.2.3.I Uncertainty, "fuzzy", and probabilistic reasoning; D.2.1.b) Elicitation methods

1 Introduction

Background. Keys are a core enabler for data management. They are fundamental for understanding the structure and semantics of data. Given a collection of entities, a key is a set of attributes whose values uniquely identify an entity in the collection. For example, a key for a relational table is a set of columns such that no two different rows have matching values in each of the key columns. For relational databases, keys were already introduced in Codd's seminal paper [1]. They form the primary mechanism to enforce entity integrity within database systems. Keys are essential for many other data models, including semantic models, object models, XML and RDF. They are fundamental in many classical areas of data management, including data modeling, database design, indexing, transaction processing, and query optimization. Knowledge about keys enables us to i) uniquely reference entities across data repositories, ii) minimize data redundancy at schema design time to process updates efficiently at run time, iii) provide better selectivity estimates in cost-based query optimization, iv) provide a query optimizer with new access paths that can lead to substantial speedups in query processing, v) allow the database administrator (DBA) to improve the efficiency of data access via physical design techniques such as data partitioning or the creation of indexes and materialized views, vi) enable access to the deep Web, and vii) provide new insights into application data. Modern applications raise the importance of keys even further. They can facilitate the data integration process and prune schema matches since attributes that form a key over one schema must be matched to attributes that form a key over the other schema. Keys can further help with the detection of duplicates and anomalies, provide guidance in repairing and cleaning data, and provide consistent answers to queries over dirty data. The discovery of keys from data is one of the core activities in data profiling.

TABLE 1
Probabilistic relation

W_1 ($p_1 = 0.2$)

rfid	*time*	*zone*
w1	2pm	z1
w1	3pm	z1
w1	3pm	z2

W_2 ($p_2 = 0.45$)

rfid	*time*	*zone*
w1	2pm	z1
w1	3pm	z1
w2	3pm	z2

W_3 ($p_3 = 0.3$)

rfid	*time*	*zone*
w1	2pm	z1
w1	3pm	z2
w2	3pm	z2

W_4 ($p_4 = .05$)

rfid	*time*	*zone*
w1	3pm	z1
w1	3pm	z2
w2	3pm	z2

Motivation. Relational databases target applications with certain data, such as accounting, inventory and payroll. Modern applications, such as data integration, information extraction, and financial risk assessment produce large volumes of uncertain data from a variety of sources. For instance, RFID (radio frequency identification) is used to track movements of endangered species of animals, such as wolverines. When probability distributions are affordable to acquire, it is sensible to apply probabilistic databases. Table 1 shows a probabilistic relation (p-relation), which is a probability distribution over a finite set of possible worlds, each being a relation.

In the same way keys enforce entity integrity in relational databases, we propose probabilistic keys to enforce probabilistic entity integrity in probabilistic databases. More precisely, keys enable database systems to uniquely identify entities within a relation, probabilistic keys enable probabilistic database systems to identify entities within a probabilistic relation with some probability. Extending this analogy further, knowledge about probabilistic keys may enable us to i) uniquely reference entities across probabilistic data repositories with some probability, ii) minimize data redundancy with some probability at schema design time to process update efficiently at run time, iii) rank selectivity estimates according to their probability in cost-based

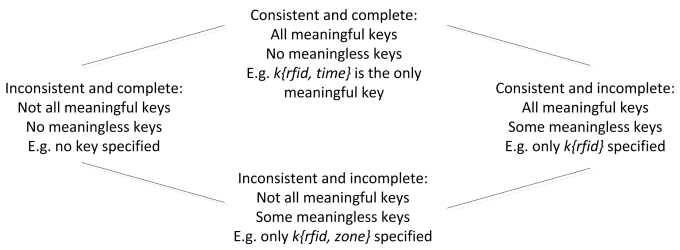


Fig. 1. Consistency and completeness dimensions as controlled by keys

query optimization, iv) use the probability of keys to rank access paths that can maximize speed ups in query processing, v) allow the DBA to improve the efficiency of data access by creating indexes based on keys that hold with sufficient probability, vi) control the efficiency of access to the deep Web, and vii) provide new insights into probabilistic application data. Before we list the contributions of our article, we illustrate some benefits of probabilistic keys on our running example.

Two important goals of a database system are to i) restrict database instances to those which are meaningful for the application domain, and ii) permit all meaningful database instances. Goal i) refers to consistency in the sense that only meaningful database instances can occur, while goal ii) refers to completeness in the sense that all meaningful database instances can occur. For goal i) we need to specify all keys that apply to the application domain, and for goal ii) we must not specify any key that does not apply to the application domain. Keys are therefore a powerful tool to address the consistency and completeness dimensions of data quality. The ultimate aim is therefore to be consistent and complete. This situation is depicted in Figure 1, where the only minimal key that applies to the given application domain is $k\{\mathit{rfid}, \mathit{time}\}$. Specifying no key would mean that we permit any database instance, in particular all those that are meaningful but also those that are meaningless. Hence, specifying no key is a case that gains completeness but not consistency. Specifying only the key $k\{\mathit{rfid}\}$ means that we also implicitly specify the actual meaningful key $k\{\mathit{rfid}, \mathit{time}\}$. That is, we do not permit any meaningless database instances, but we also exclude some meaningful database instances from occurring. Namely those meaningful database instances in which $k\{\mathit{rfid}, \mathit{time}\}$ is satisfied but $k\{\mathit{rfid}\}$ is not. Hence, specifying only the key $k\{\mathit{rfid}\}$ is a case that gains consistency but no completeness. Finally, specifying only the key $k\{\mathit{rfid}, \mathit{zone}\}$ means that we permit some meaningless database instances and also exclude some meaningful database instances. For example, any database instance that satisfies $k\{\mathit{rfid}, \mathit{zone}\}$ but violates $k\{\mathit{rfid}, \mathit{time}\}$ is meaningless, and any database instance that violates $k\{\mathit{rfid}, \mathit{zone}\}$ but satisfies $k\{\mathit{rfid}, \mathit{time}\}$ is meaningful but excluded. Hence, specifying only the key $k\{\mathit{rfid}, \mathit{zone}\}$ is a case that gains neither consistency nor completeness.

Due to the veracity inherent to probabilistic databases as well as the variety of sources the data originates from, the traditional concept of a key requires revision in this context. In our example, for instance, there is no non-trivial key that is satisfied by all possible worlds: the key $k1 = k\{\mathit{time}, \mathit{zone}\}$ holds in the worlds W_1 and W_2, $k2 = k\{\mathit{rfid}, \mathit{time}\}$ holds in W_2 and W_3, and $k3 = k\{\mathit{rfid}, \mathit{zone}\}$ holds in W_3 and W_4. One may argue to remove possible worlds that violate a key but this would neither address the completeness dimension of data quality nor would it make sensible use of probabilistic databases. Instead, we propose the concept of a *probabilistic key*, or p-key for short, which stipulates a lower bound on the marginal probability by which a traditional key holds in a probabilistic database. In our example, $k1$, $k2$, and $k3$ have marginal probability 0.65, 0.75, and 0.35, respectively, which is the sum of the probabilities of those possible worlds which satisfy the key. Indeed, the marginal probability of a key provides a control mechanism to balance consistency and completeness targets for the quality of data. Larger marginal probabilities represent stricter consistency and more liberal completeness targets, while smaller marginal probabilities represent more liberal consistency and stricter completeness targets. Having fixed these targets in the form of a lower bound on the marginal probability, p-keys can be utilized to control these data quality dimensions during updates or validate them for static analysis purposes. For instance, p-keys can help detect anomalous patterns of data in the form of p-key violations, either on a given database or when new data arrives. Such alerts can be sent out automatically when a data set does not meet a desired lower bound on the marginal probability of a key. In a different showcase, p-keys can also be used to infer probabilities that query answers are unique. In our example, we may wonder about the chance that different wolverines are in the same zone at the same time, indicating potential mating behavior. We may ask

```
SELECT DISTINCT  rfid
FROM             TRACKING
WHERE            zone='z2' AND time='2pm'
```

and using our p-keys enables us to derive a minimum probability of 0.65 that a unique answer is returned, that is, different wolverines are in zone z2 at 2pm at most with probability 0.35. These bounds can be inferred without accessing any portion of a potentially big data source at all, only requiring that the key $k1$ has at least marginal probability 0.65 on the given data set. As a final showcase, the keys $k2$, $k1$, and $k3$ should be chosen in this order to rank indexes that may be created for the speedup of data access or query evaluation.

Contributions. We propose probabilistic keys to equip the traditional notion of a key with probabilities, that is, with the main quantitative tool to stipulate uncertainty in data. While it is already challenging to identify traditional keys which are semantically meaningful in a given application domain, it is an even harder problem to identify the probabilities by which keys should hold

on quality probabilistic data. As a realistic compromise, we stipulate lower bounds on the marginal probability by which keys should hold. Our contributions can be summarized as follows.

Modeling. We propose p-keys as a natural class of semantic integrity constraints over uncertain data. P-keys are expressions of the form $kX_{\geq p}$ and state that the key kX must hold at least with marginal probability p. The special case of p-keys $kX_{\geq 1}$ on p-relations with just one possible world captures the notion of a traditional key over traditional relations. The intention of p-keys is to provide benefits to the management of probabilistic data in the same way traditional key benefit the management of traditional data. One main application is to help organizations balance consistency and completeness targets for the quality of their data. P-keys can distinguish semantically meaningful from meaningless patterns in large volumes of uncertain data from a variety of sources, help quantify the probability for unique query answers, and provide rankings of access structures that speedup data processing.

Reasoning. We characterize the implication problem of p-keys axiomatically by a simple finite set of Horn rules, as well as a linear time decision algorithm. This enables organizations to reduce the overhead of managing p-keys to a minimal level necessary. For example, enforcing $k\{rfid\}_{\geq 0.3}$, $k\{rfid,time\}_{\geq 0.25}$, and $k\{rfid,zone\}_{\geq 0.35}$, would be redundant as the enforcement of $k\{rfid,time\}_{\geq 0.25}$ is already implicitly done by enforcing $k\{rfid\}_{\geq 0.3}$.

Visualization. A main inhibitor to the uptake of p-keys is the difficulty of determining the right lower bound on the marginal probabilities by which keys should hold. For the schema-driven elicitation of the lower bounds, we show how to visualize concisely any given system of p-keys in the form of Armstrong PC-tables. These Armstrong PC-tables are perfect semantic summaries of all p-keys currently perceived meaningful by the analysts. That is, the Armstrong PC-table satisfies every key with the exact marginal probability that is perceived to best represent the application domain. Data engineers can use our algorithm to compute an Armstrong PC-table which they can jointly inspect with domain experts to identify any problems with the perceptions they currently have about the application domain. For example, Figure 2 shows an Armstrong PC-table for the p-key set $\{k1_{\geq 0.65}, k2_{\geq 0.75}, k3_{\geq 0.35}\}$. In the CD table, the W column of a tuple shows the identifiers of possible worlds to which the tuple belongs. The P-table shows the probability distribution on the possible worlds. The PC-table represents the p-relation from Table 1. Every p-key that is not implied by the p-key set is violated by the p-relation, in particular the keys $k\{rfid\}$, $k\{time\}$ and $k\{zone\}$ all have marginal probability zero in the p-relation. To the best of our knowledge, our article is the first to investigate the concept of an Armstrong database in the context of probabilistic databases. Our results link Armstrong databases with the well-known

Fig. 2. Armstrong PC-table for $\{k1_{\geq 0.65}, k2_{\geq 0.75}, k3_{\geq 0.35}\}$

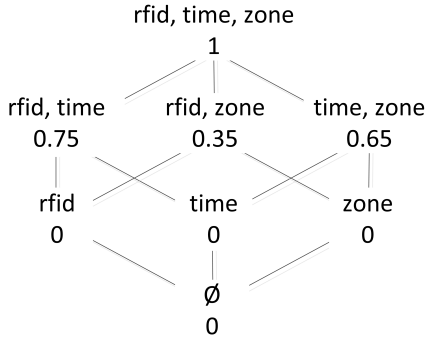
CD table

rfid	*time*	*zone*	W
w1	2:00pm	z1	$1, 2, 3$
w1	3:00pm	z1	$1, 2, 4$
w1	3:00pm	z2	$1, 3, 4$
w2	3:00pm	z2	$2, 3, 4$

P table

W	$\mathcal{P}$
1	.2
2	.45
3	.3
4	.05

Fig. 3. Profile of p-keys that hold on PC-table in Figure 2



complete representation systems of PC-tables.

Profiling. For the data-driven elicitation of p-keys we compute the marginal probability of every key from a given PC-table. This algorithm supports the elicitation process because domain experts or data engineers may want to apply some changes to the PC-table or p-relation they inspect. After such changes have been applied, they want to know which p-keys hold on the new data set. The process of discovering patterns in data is also known as data mining or data profiling. To the best of our knowledge, our paper is the first to propose probabilistic data profiling techniques. For example, if we want to know the marginal probabilities by which an attribute set forms a key in the PC-table from Figure 2, then our algorithm would return the profile $k\emptyset_{\geq 0}$, $k\{rfid\}_{\geq 0}$, $k\{time\}_{\geq 0}$, $k\{zone\}_{\geq 0}$, $k\{rfid,time\}_{\geq 0.75}$, $k\{rfid,zone\}_{\geq 0.35}$, $k\{time,zone\}_{\geq 0.65}$, and $k\{rfid,time,zone\}_{\geq 1}$, as visualized in Figure 3. We apply our profiling technique to PC-tables that results from any changes that data engineers or domain experts have made on the Armstrong PC-tables generated previously. In this context the number of the possible worlds is relatively low.

Experiments. Our experiments demonstrate that our visualization and profiling techniques work efficiently in the context of our elicitation framework. In particular, a strong point in our construction of Armstrong PC-tables is the low number of possible worlds required by the p-relation it represents. While the problem of computing an Armstrong PC-table is shown to be precisely exponential in the input, our experiments show that, for an average input set of reasonable size, it takes less than a second to compute an Armstrong PC-table with a small number of tuples. Experiments with MapReduce further indicate that the profiling time for p-keys scales linearly in the underlying number of possible worlds, which we limited

to forty within the context of our elicitation framework. **Organization.** We discuss related work in Section 2. P-keys are introduced in Section 3, and axiomatic and linear-time algorithmic characterizations of their implication problem are established in Section 4. These lay the foundation for the schema- and data-driven discovery algorithms of p-keys in Section 5. Experiments with these algorithms are presented in Section 6. We conclude and sketch future work in Section 7.

2 Related Work

Integrity constraints enforce the semantics of application domains in database systems. They form a cornerstone of database technology [2]. Entity integrity is one of the three inherent integrity rules proposed by Codd [3]. Keys and foreign keys are the only ones amongst around 100 classes of constraints [2] that enjoy built-in support by SQL database systems. In particular, entity integrity is enforced by primary keys [4]. Core problems investigate reasoning [5], Armstrong databases [6], and discovery [7], [8], [9], [10], [11]. Applications include anomaly detection [12], consistency management [13], consistent query answers [14], [15], data cleaning [16], exchange [17], fusion [18], integration [19], profiling [20], quality [21], repairs [22], and security [23], schema design [24], query optimization [25], transaction processing [26], and view maintenance [27]. Surrogate keys ('auto-increment' fields) do not help with enforcing domain semantics or supporting applications while semantic keys do. The important role of keys transcends beyond the relational model of data: They have been investigated in data models with incomplete [28], [29], [30], [31], temporal [32], object-oriented [33], [34], XML [35], [36], [37], [38], and RDF data [39], as well as description logics [40], but not in probabilistic data models.

Our contributions extend results on keys from traditional relations, covered by our framework as the special case where the p-relation consists of one possible world only. Extensions include work on the classical implication problem [41], [29], Armstrong relations [42], [43], [6], [29], [44] and the discovery of keys from relations [45], [7], [9]. In fact, our axiomatic and algorithmic characterizations of the implication problem as well as the schema- and data-driven discovery of the right probabilities of keys are novel. Specifically, Armstrong databases and data profiling have not been studied yet for probabilistic data. For traditional relations and relations with incomplete information there is empirical evidence that Armstrong databases help with the elicitation of meaningful business rules [46]. Our techniques will make it possible to conduct such empirical studies for p-keys in the future.

There is a large body of work on the discovery of "approximate" business rules, such as keys, functional and inclusion dependencies [47], [48], [9]. Approximate means here that not all tuples satisfy the given rule, but some exceptions are tolerable. Our constraints are not approximate since they are either satisfied or violated by the given p-relation or the PC-table that represents it. Again, it is future work to investigate approximate versions of probabilistic keys.

Closest to our approach is the work on possibilistic keys [49], where tuples are attributed some degree of possibility and keys some degree of certainty saying to which tuples they apply. In general, possibility theory can offer a qualitative approach, while probability theory is a quantitative approach to uncertainty. This research thereby complements the qualitative approach to keys in [49] by a quantitative approach. In the same possibilistic model, follow-up work has investigated possibilistic functional dependencies [50], [51], cardinality constraints [52], [53], normal forms and normalization [54], as well as non-invasive data cleaning [55].

Keys have also been included in description logic research [40], but we are unaware of any work concerning keys on probabilistic data.

PC-tables are well-known systems that can represent every probabilistic database [56]. Instead of computing p-relations that are Armstrong for a given set of probabilistic keys, it is natural to represent these p-relations as PC-tables. After all, Armstrong databases are meant to provide semantic summaries of a small size. It makes therefore sense to apply these representation systems to decrease the size of the summary further.

The results of this article have been announced in [57]. The current article has been extended in different directions. Firstly, we have included all the proofs, which provide the actual insight into our results. Secondly, the overall presentation of the results has been extended, including a more detailed justification for the new concept of probabilistic keys, and additional examples that illustrate our concepts and results. Thirdly, we have included a new characterization of the implication problem for probabilistic keys in terms of the implication problem for traditional keys. The characterization allows us to apply our well-developed understanding of traditional keys to the new concept of probabilistic keys. Finally, we have included additional experiments illustrating that the discovery of probabilistic keys from probabilistic relations scales linearly in the number of possible worlds required by our elicitation framework.

In follow-up work of the present article, different extensions of probabilistic keys have been investigated. These include probabilistic keys that stipulate upper bounds on the marginal probability by which keys hold on a probabilistic database as well as probabilistic keys that stipulate lower and upper bounds [58], and probabilistic cardinality constraints which stipulate upper bounds on the marginal probability by which cardinality constraints hold on a probabilistic database [59].

3 Probabilistic Keys

We introduce preliminary concepts from probabilistic databases and the new notion of a probabilistic key.

A *relation schema* is a finite set R of attributes A. Each attribute A is associated with a domain $dom(A)$ of values. A tuple t over R is a function that assigns to each attribute A of R an element $t(A)$ from the domain $dom(A)$. A *relation* over R is a finite set of tuples over R. Relations over R are also called *possible worlds* of R here. An expression kX over R with $X \subseteq R$ is called a *key*. A key kX is said to hold in a possible world W of R, denoted by $W \models kX$, if and only if there are no two tuples $t_1, t_2 \in W$ such that $t_1 \neq t_2$ and $t_1(X) = t_2(X)$. A *probabilistic relation* (p-relation) over R is a pair $r = (\mathcal{W}, P)$ of a finite non-empty set $\mathcal{W}$ of possible worlds over R and a probability distribution $P : \mathcal{W} \to (0, 1]$ such that $\sum_{W \in \mathcal{W}} P(W) = 1$ holds.

Example 1: Table 1 shows a probabilistic relation over relation schema WOLVERINE={*rfid*,*time*,*zone*}. World W_2, for example, satisfies the keys $k\{rfid, time\}$ and $k\{zone, time\}$, but violates the key $k\{rfid, zone\}$.

The *marginal probability* of a key kX in the p-relation $r = (\mathcal{W}, P)$ over relation schema R, denoted by $m_{kX,r}$ is the sum of the probabilities of those possible worlds in r which satisfy the key, that is, $m_{kX,r} = \sum_{W \in \mathcal{W}, W \models kX} P(W)$.

Example 2: Let r denote the p-relation from Table 1, $k1 = k\{time, zone\}$, $k2 = k\{rfid, time\}$, and $k3 = k\{rfid, zone\}$. The marginal probabilities of the keys $k1$, $k2$, and $k3$ in r are $m_{k1,r} = P(W_1) + P(W_2) = 0.65$, $m_{k2,r} = P(W_2) + P(W_3) = 0.75$, and $m_{k3,r} = P(W_3) + P(W_4) = 0.35$.

Next we define the central notion of our article.

Definition 1: A *probabilistic key*, or *p-key* for short, over relation schema R is an expression $kX_{\geq p}$ where $X \subseteq R$ and $p \in [0, 1]$. The p-key $kX_{\geq p}$ over R is *satisfied by*, or said to *hold in*, the p-relation r over R if and only if the marginal probability of kX in r is at least p, that is, $m_{kX,r} \geq p$.

Example 3: In our running example over relation schema WOLVERINE, the p-relation from Table 1 satisfies the p-keys $k\{rfid, time\}_{\geq 0.75}$ and $k\{rfid, zone\}_{\geq 0.35}$, but violates the p-keys $k\{rfid, time\}_{\geq 0.9}$ and $k\{rfid, zone\}_{\geq 0.351}$.

4 REASONING TOOLS

When using sets of p-keys to manage the integrity of entities in probabilistic databases, it is important that their overhead is reduced to a minimal level necessary. In practice, this requires us to reason about p-keys efficiently. It is the goal of this section to establish basic tools to understand the interaction of p-keys and to efficiently reason about them. This will help us compute in linear time the largest probability by which a given key is implied by a given set of p-keys. Finally, we show how to decide instances of the implication problem for p-keys by instances of the implication problem for traditional keys. This allows humans to apply their well-developed understanding about the interaction of traditional keys to the new concept of probabilistic keys. The results will also help us develop our elicitation framework in Section 5.

4.1 Implication and Inference Problems

Let $\Sigma \cup \{\varphi\}$ denote a set of constraints over relation schema R. We say Σ *implies* φ, denoted by $\Sigma \models \varphi$, if every p-relation r over R that satisfies Σ, also satisfies φ. We use $\Sigma^* = \{\varphi : \Sigma \models \varphi\}$ to denote the *semantic closure* of Σ. For a class $\mathcal{C}$ of constraints, the $\mathcal{C}$-implication problem is to decide for a given relation schema R and a given set $\Sigma \cup \{\varphi\}$ of constraints in $\mathcal{C}$ over R, whether Σ implies φ. Our goal is to characterize the $\mathcal{C}$-implication problem for the class of p-keys axiomatically by a simple finite set of Horn rules, and algorithmically by a linear time algorithm.

Problem:	Implication
Input:	Relation schema R
	Set $\Sigma \cup \{\varphi\}$ of p-keys over R
Output:	Yes, if $\Sigma \models \varphi$
	No, otherwise

As we defined possible worlds to be finite, we are strictly speaking about the finite implication problem. In theory we could also allow possible worlds to be infinite sets of tuples, and call a p-relation infinite if it contains some infinite possible world. This would lead to the unrestricted implication problem in which p-relations may also be infinite. However, for the class of p-keys, finite and unrestricted implication problems coincide.

Proposition 1: The finite and unrestricted implication problems for probabilistic keys coincide.

Proof: Let $\Sigma \cup \{\varphi\}$ denote a set of p-keys over R. If Σ implies φ in the unrestricted case, then Σ also implies φ in the finite case. Vice versa, assume that Σ does not imply $\varphi = kX_{\geq p}$ in the unrestricted case, and let $r = (\mathcal{W}, P)$ be a p-relation over R that contains at least one possible world in $\mathcal{W}$ which is not finite, and where r satisfies all elements in Σ but violates φ. Consequently, there must be a finite subset $\mathcal{W}_0 \subseteq \mathcal{W}$ of possible worlds in which kX is violated. Each of the possible worlds $W \in \mathcal{W}_0$ must thus contain two tuples t_1^W, t_2^W such that $t_1^W(X) = t_2^W(X)$ holds. Let $r' = (\mathcal{W}', P')$ result from r by i) replacing each world $W \in \mathcal{W}_0$ by the finite world $W' = \{t_1^W, t_2^W\}$, and each world $W \in \mathcal{W} - \mathcal{W}_0$ by an arbitrary singleton subset $W' \subseteq W$, and ii) defining $P(W') := P(W)$. It follows immediately that r' satisfies every p-key in Σ (subsets of possible worlds that satisfy a key also satisfy the key) but violates φ (violations of kX are maintained in the corresponding possible worlds). This shows that Σ does also not imply φ in the finite case. □

TABLE 2
Axiomatization $\mathfrak{P} = \{\mathfrak{T}, \mathfrak{Z}, \mathfrak{S}, \mathfrak{W}\}$

$\frac{}{kR_{\geq 1}}$	$\frac{}{kX_{\geq 0}}$	$\frac{kX_{\geq p}}{kXY_{\geq p}}$	$\frac{kX_{\geq p+q}}{kX_{\geq p}}$
(Trivial, $\mathfrak{T}$)	(Zero, $\mathfrak{Z}$)	(Superkey, $\mathfrak{S}$)	(Weakening, $\mathfrak{W}$)

We therefore speak of *the* implication problem. The ability to decide the implication problem efficiently has many applications in data management. For example, a p-relation that satisfies every p-key in a given set Σ also satisfies every p-key implied by the set. In particular, if a p-key $\sigma \in \Sigma$ is implied by $\Sigma - \{\sigma\}$, then σ is said to be *redundant* because it is redundant to check whether σ holds in the p-relation whenever it is known that every p-key in $\Sigma - \{\sigma\}$ holds in the p-relation. Consequently, by deciding the implication problem efficiently, we can also compute a set of non-redundant p-keys efficiently. This minimizes overheads when p-relations are updated. In fact, the larger p-relations are, the more time saving result from using sets of non-redundant p-keys.

A related computational problem is to infer for a given key kX and a given set Σ of p-keys, the largest probability p such that Σ implies the p-key $kX_{\geq p}$.

Problem:	Inference
Input:	Relation schema R
	Set Σ of p-keys over R
	Key kX over R
Output:	$\max\{p \mid \Sigma \models kX_{\geq p}\}$

Our goal is to establish an algorithm that solves the inference problem in time linear in the input.

4.2 Inference System

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 an inference rule in $\mathfrak{R}$ to some premises in $\{\sigma_1, \ldots, \sigma_{i-1}\}$. Let $\Sigma^+_{\mathfrak{R}} = \{\varphi : \Sigma \vdash_{\mathfrak{R}} \varphi\}$ be the *syntactic closure* of Σ under inferences by $\mathfrak{R}$. $\mathfrak{R}$ is *sound* (*complete*) if for every relation schema R, and for every set Σ over R, we have $\Sigma^+_{\mathfrak{R}} \subseteq \Sigma^*$ ($\Sigma^* \subseteq \Sigma^+_{\mathfrak{R}}$). The (finite) set $\mathfrak{R}$ is a (finite) *axiomatization* if $\mathfrak{R}$ is both sound and complete.

The set $\mathfrak{P}$ of inference rules from Table 2 forms a finite axiomatization for the implication problem of p-keys. In these rules, R denotes the underlying relation schema, X and Y form attribute subsets of R, and p, q as well as $p + q$ are probabilities.

Theorem 1: $\mathfrak{P}$ forms a finite axiomatization for the implication problem of p-keys.

Proof: The soundness is a straightforward consequence of the definitions. Indeed, $\mathfrak{T}$ is sound since every possible world over R is a relation that cannot contain two different tuples with matching values on all the attributes of R. The soundness of $\mathfrak{Z}$ is satisfied trivially. The soundness of $\mathfrak{S}$ follows from the fact that every possible world that satisfies kX also satisfies kXY. The soundness of $\mathfrak{W}$ follows immediately from the definition of a p-key with lower bounds.

The completeness proof uses contraposition, showing that non-inferable p-keys are also non-implied. In fact, for the completeness of $\mathfrak{P}$ let R be some relation schema and $\Sigma \cup \{kX_{\geq p}\}$ be a set of p-keys over R such that $kX_{\geq p} \notin \Sigma^+_{\mathfrak{P}}$. We need to show that $kX_{\geq p} \notin \Sigma^*$. From $kX_{\geq p} \notin \Sigma^+_{\mathfrak{P}}$ we conclude that $p > 0$ and $R - X \neq \emptyset$, due to $\mathfrak{Z}$ and $\mathfrak{S}, \mathfrak{W}$, respectively. Let $p' := \sup\{p'' : kZ_{\geq p''} \in \Sigma \wedge Z \subseteq X\}$. In particular, $p' = 0$, if there is no $kZ_{\geq p''} \in \Sigma$ where $Z \subseteq X$. We conclude that $p' < p$, as otherwise the following would apply: Since $kZ_{\geq p'} \in \Sigma$ we get $kX_{\geq p'} \in \Sigma^+_{\mathfrak{P}}$ by $\mathfrak{S}$ and $kX_{\geq p} \in \Sigma^+_{\mathfrak{P}}$ by $\mathfrak{W}$. We now define the following p-relation $r = (\mathcal{W}, P)$ over R:

W_1 with $P(W_1) = 1 - p'$	
X	$R - X$
$0 \cdots 0$	$0 \cdots 0$
$0 \cdots 0$	$1 \cdots 1$

W_2 with $P(W_2) = p'$	
X	$R - X$
$0 \cdots 0$	$0 \cdots 0$

Note that $W_1 \in \mathcal{W}$ and $W_2 \in \mathcal{W}$, if $p' > 0$. As kX does not hold in world W_1, it follows that kX holds with probability p' on r. Since $p' < p$, we conclude that $kX_{\geq p}$ does not hold on r. It remains to show that every p-key $kZ_{\geq q} \in \Sigma$ holds on r. If $Z \not\subseteq X$, then kZ holds in both worlds W_1 and W_2, and the probability of kZ is 1. Consequently, $kZ_{\geq q}$ holds on r. Otherwise, $Z \subseteq X$ and the probability with which kZ holds on r is p'. Moreover, as $Z \subseteq X$ and $kZ_{\geq q} \in \Sigma$ we have $p' \geq q$. Consequently, $kZ_{\geq q}$ holds on r. □

Example 4: The set $\Sigma = \{k\{\textit{time}\}_{\geq 0.2}, k\{\textit{rfid}\}_{\geq 0.3}\}$ imply the p-key $\varphi = k\{\textit{rfid}, \textit{time}\}_{\geq 0.25}$, but not the p-key $\varphi' = k\{\textit{rfid}, \textit{time}\}_{\geq 0.35}$. Indeed, φ can be inferred from Σ by applying $\mathfrak{S}$ to $k\{\textit{rfid}\}_{\geq 0.3}$ to infer $k\{\textit{rfid}, \textit{time}\}_{\geq 0.3}$, and applying $\mathfrak{W}$ to $k\{\textit{rfid}, \textit{time}\}_{\geq 0.3}$ to infer φ.

Example 5: The set $\Sigma = \{k\{\textit{time}\}_{\geq 0.2}, k\{\textit{rfid}\}_{\geq 0.3}\}$ does not imply the p-key $k\{\textit{rfid}, \textit{time}\}_{\geq 0.35}$. We can apply the construction in the proof of Theorem 1, which leads to the following p-relation $r = (\{W_1, W_2\}, P)$:

W_1 with $P(W_1) = 0.7$		
rfid	*time*	*zone*
0	0	0
0	0	1

W_2 with $P(W_2) = 0.3$		
rfid	*time*	*zone*
0	0	0

.

Clearly, r satisfies all p-keys in Σ and violates $k\{\textit{rfid}, \textit{time}\}_{\geq 0.35}$.

Algorithm 1 Inference

Require: R, Σ, kX
Ensure: $\max\{p : \Sigma \models kX_{\geq p}\}$
1: **if** $X = R$ **then**
2: $p \leftarrow 1$;
3: **else**
4: $p \leftarrow 0$;
5: **for all** $kZ_{\geq q} \in \Sigma$ **do**
6: **if** $Z \subseteq X$ and $q > p$ **then**
7: $p \leftarrow q$;
8: **return** p;

4.3 Algorithmic Characterization

In practice, the semantic closure Σ^* of a finite set Σ is infinite and even though it can always be represented finitely, it is often unnecessary to determine all implied p-keys. In fact, the implication problem for p-keys has as input $\Sigma \cup \{\varphi\}$ and the question is whether Σ implies φ. Computing Σ^* and checking whether $\varphi \in \Sigma^*$ is not feasible. In fact, we will now establish a linear-time algorithm for computing the maximum probability p, such that $kX_{\geq p}$ is implied by Σ. The following theorem allows us to reduce the implication problem for p-keys to a single scan of the input.

Theorem 2: Let $\Sigma \cup \{kX_{\geq p}\}$ denote a set of p-keys over relation schema R. Then Σ implies $kX_{\geq p}$ if and only if $X = R$ or $p = 0$ or there is some $kZ_{\geq q} \in \Sigma$ such that $Z \subseteq X$ and $q \geq p$.

Proof: The sufficiency of the three conditions will be established using the soundness of $\mathfrak{P}$, while the necessity will be established by using the completeness of $\mathfrak{P}$.

We show the sufficiency first. If $X = R$, then the soundness of $\mathfrak{T}$ and $\mathfrak{W}$ imply that $\Sigma \models kX_{\geq p}$. If $p = 0$, then the soundness of $\mathfrak{Z}$ ensures that $\Sigma \models kX_{\geq p}$. If there is $kZ_{\geq q} \in \Sigma$ such that $Z \subseteq X$ and $q \geq p$, then the soundness of $\mathfrak{S}$ and $\mathfrak{W}$ imply that $\Sigma \models kX_{\geq p}$.

It remains to show the necessity. Let $R - X \neq \emptyset$, $p > 0$, and Σ such that for all $Z \subseteq X$ we have $q < p$. Using the terminology from the completeness proof of Theorem 1 it follows that $p' := \sup\{p'' : kZ_{\geq p''} \in \Sigma \wedge Z \subseteq X\} < p$. Consequently, the p-relation r from the completeness proof of Theorem 1 shows that Σ does not imply $kX_{\geq p}$. □

Theorem 2 enables us to design Algorithm 1, which returns the maximum probability p by which a given key kX is implied by a given set Σ of p-keys over R. If $X = R$, then we return probability 1. Otherwise, starting with $p = 0$ the algorithm scans all input keys $kZ_{\geq q}$ and sets p to q whenever q is larger than the current p and X contains Z. We use $|\Sigma|$ and $|R|$ to denote the total number of attributes that occur in Σ and R, respectively.

Theorem 3: On input (R, Σ, kX), Algorithm 1 returns in $\mathcal{O}(|\Sigma| + |R|)$ time the maximum probability p with which $kX_{\geq p}$ is implied by Σ.

Proof: The correctness of Algorithm 1 follows from Theorem 2.

Algorithm 1 returns $p = 1$, if $X = R$. Otherwise it returns the largest probability p that results from an input key $kZ_{\geq p}$ where $Z \subseteq X$. By Theorem 2 this p is the largest probability by which $kX_{\geq p}$ is implied by Σ. Thus, Algorithm 1 is correct.

The time complexity follows from having to look at each attribute occurrence in the input once. □

Given $R, \Sigma, kX_{\geq p}$ as an input to the implication problem, Algorithm 1 computes $p' := \max\{q : \Sigma \models kX_{\geq q}\}$ and returns an affirmative answer if and only if $p' \geq p$. We therefore obtain the following result.

Corollary 1: The implication problem of p-keys is decidable in linear time.

Example 6: Given $\Sigma = \{k\{time\}_{\geq 0.2}, k\{rfid\}_{\geq 0.3}\}$ and the key $k\{rfid, time\}$, Algorithm 1 returns $p = 0.3$. Consequently, the p-key $k\{rfid, time\}_{\geq 0.25}$ is implied by Σ, but $k\{rfid, time\}_{\geq 0.35}$ is not implied by Σ.

4.4 Reasoning about p-keys with traditional keys

As a final result of this section we show how any instance I of the implication problem for p-keys can be translated into an instance I' of the implication problem for traditional keys such that I is true if and only if I' is true. Considering Theorem 2, the decision about $I = (\Sigma, kX_{\geq p})$ only relies on the p-keys $kZ_{\geq q}$ in Σ where $q \geq p$. Indeed, if there is some $kZ_{\geq q}$ with $Z \subseteq X$, then I is true. A special case occurs for $X = R$, which always results in a true instance I. A traditional key kX is implied by a set Σ_p of traditional keys if and only if $X = R$ or there is some key $kZ \in \Sigma_p$ such that $Z \subseteq X$. Therefore we define

$$\Sigma_p := \{kX \mid \exists kX_{\geq q} \in \Sigma \cup \{k\emptyset_{\geq 0}\} \wedge q \geq p\}$$

and obtain the following result.

Theorem 4: Let $\Sigma \cup \{kX_{\geq p}\}$ denote a set of p-keys over relation schema R. Then Σ implies $kX_{\geq p}$ if and only if $\Sigma_p \models kX$.

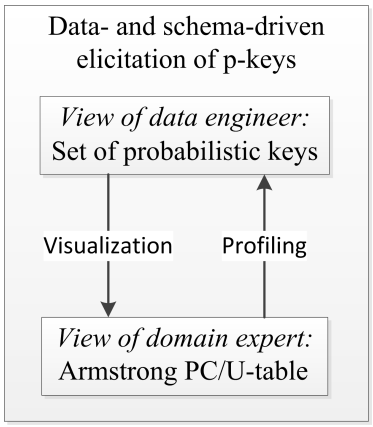
Proof: The proof uses a combination of Theorem 2 and the algorithmic characterization of traditional keys.

We use the classical result that for a set $\Sigma_p \cup \{kX\}$ of traditional keys, Σ_p implies kX if and only if $X = R$ or there is some $kZ \in \Sigma_p$ such that $Z \subseteq X$ [60].

By Theorem 2, Σ implies $kX_{\geq p}$ if and only if i) $X = R$ or ii) $p = 0$ or iii) there is some $kZ_{\geq q} \in \Sigma$ such that $Z \subseteq X$ and $q \geq p$.

For the *only if* direction assume that Σ implies $kX_{\geq p}$. If i) holds, then Σ_p implies kX. If ii) holds, then Σ_p contains $k\emptyset$ and the empty set $\emptyset$ is contained in every set, that

Fig. 4. Elicitation framework



is, Σ_p implies kX. If iii) holds, then $kZ \in \Sigma_p$ for some $Z \subseteq X$. Consequently, Σ_p implies kX. Consequently, Σ_p implies kX.

For the *if* direction assume that Σ_p implies kX. If $X = R$ holds, then i) holds and Σ implies $kX_{\geq p}$. If there is some $kZ \in \Sigma_p$ such that $Z \subseteq X$, then for some $q \geq p$ there is some $kZ_{\geq q} \in \Sigma$ and hence iii) holds. Consequently, Σ implies $kX_{\geq p}$. □

Example 7: We have seen in Example 6 that the p-key set $\Sigma = \{k\{time\}_{\geq 0.2}, k\{rfid\}_{\geq 0.3}\}$ implies the p-key $k\{rfid, time\}_{\geq 0.25}$. This can be confirmed by Theorem 4 since $\Sigma_{0.25} = \{k\{rfid\}\}$ and $\{rfid\}$ is a subset of $\{rfid, time\}$. We have also seen in Example 6 that the p-key set $\Sigma =$ implies the p-key $k\{rfid, time\}_{\geq 0.35}$. This can be confirmed by Theorem 4 since $\Sigma_{0.35} = \emptyset$ and thus there is no key in $\Sigma_{0.35}$ that is contained in $\{rfid, time\}$.

5 Elicitation of Probabilistic Keys

Applications will benefit from the ability of data engineers to acquire a good lower bound for the marginal probability by which keys hold in the domain of the application. For that purpose, analysts should communicate with domain experts. We establish two major tools that help engineers communicate effectively with domain experts. We follow the framework in Figure 4. Here, engineers use our algorithm to visualize abstract sets Σ of p-keys in the form of some Armstrong PC-table, which is then inspected jointly with domain experts. In particular, the PC-table represents simultaneously for every key kX the marginal probability that quality data sets in the target domain should exhibit. Domain experts may change the PC-table or supply new PC-tables to the engineers. For that case we establish an algorithm that profiles p-keys. That is, the algorithm computes the marginal probability of each key in the given PC-table. Such profiles are also useful for query optimization, for example.

5.1 Armstrong relations for keys

As Armstrong p-relations generalize the concept of Armstrong relations, it is worth to summarize the basics about Armstrong relations for the class of traditional keys. The general notion of an Armstrong relation is as follows. For a class $\mathcal{C}$ of constraints, a relation schema R, a set Σ of constraints from $\mathcal{C}$ over R, a relation r over R is said to be $\mathcal{C}$-Armstrong for Σ if and only if for every constraint φ from $\mathcal{C}$ over R the following holds: r satisfies φ if and only if Σ implies φ [61]. It is well-known that traditional keys *enjoy* Armstrong relations [60]. That is, for every relation schema R, and for every set Σ of traditional keys over R, there is some relation r over R that is Armstrong for Σ.

One general construction of an Armstrong relation is to compute from the input set Σ the set Σ^{-1} of anti-keys. An attribute subset $X \subseteq R$ is an anti-key for Σ if and only if X is maximal with the property that kX is not implied by Σ. Having computed Σ^{-1}, one starts with a base tuple t_0 and adds for every anti-key $X \in \Sigma^{-1}$ a tuple that agrees with t_0 on every attribute in X and has unique values on all the other attributes.

Example 8: Let Σ consist of the two keys $k\{time, zone\}$ and $k\{rfid, time\}$. Then the set Σ^{-1} of anti-keys for Σ consists of $\{time\}$ and $\{rfid, zone\}$. Applying the construction above, the following relation

rfid	*time*	*zone*
w1	2pm	z1
w3	2pm	z3
w1	5pm	z1

is indeed Armstrong for Σ.

5.2 Construction of Armstrong p-relations

Our goal for now is to show constructively that p-keys enjoy Armstrong p-relations. First, we state the definition of Armstrong databases for the class of p-keys.

Definition 2: Let Σ denote a set of p-keys over a given relation schema R. A p-relation $r = (\mathcal{W}, P)$ over R is *Armstrong* for Σ if and only if for all p-keys φ over R, r satisfies φ if and only if Σ implies φ.

In particular, Definition 2 captures that of traditional Armstrong relations [61] as the special case where only one possible world exists.

Example 9: The p-relation in Table 1 is Armstrong for the set Σ of the following p-keys: $k\{time, zone\}_{\geq 0.65}$, $k\{rfid, time\}_{\geq 0.75}$, and $k\{rfid, zone\}_{\geq 0.35}$.

As a consequence of Definition 2, every Armstrong p-relation r for Σ has the property that for every key kX, the marginal probability $m_{kX,r}$ of kX in r is the largest probability p such that $kX_{\geq p}$ is implied by Σ. In

other words, the availability of an Armstrong p-relation r reduces, for all kX, the inference problem with input (Σ, kX) to the computation of the marginal probability $m_{kX,r}$ of kX in r. Given such strong semantic data summarization properties, the existence of Armstrong p-relations cannot be taken for granted.

Example 10: Continuing Example 9, the marginal probabilities of all keys over WOLVERINE in the p-relation r of Table 1 are showing in Figure 3.

The following theorem shows that every distribution of probabilities to keys, that follows the inference rules from Table 2, can be represented by a single p-relation which exhibits this distribution in the form of marginal probabilities.

Theorem 5: Let $l : R \to [0,1]$ be a function such that $l(R) = 1$ and for all $X, Y \subseteq R$, $l(XY) \geq l(X)$ holds. Then there is some p-relation r over R such that r satisfies $kX_{\geq l(X)}$, and for all $X \subseteq R$ and for all $p \in [0,1]$ such that $p > l(X)$, r violates $kX_{\geq p}$.

Proof: The proof uses a reduction to the existence of Armstrong relations for traditional keys.

Let $\{l_1, \ldots, l_n\} = \{l(X) : X \subseteq R\}$ such that $l_1 < l_2 < \ldots < l_n$, and let $l_0 = 0$. Define a probabilistic relation $r = (\{W_1, \ldots, W_n\}, P)$ as follows. For all $i = 1, \ldots, n$, the world W_i is an Armstrong relation for the key set $\Sigma_i = \{kY : l(Y) \geq l_i\}$, and $P(W_i) = l_i - l_{i-1}$. For all $X \subseteq R$, let $l(X) = l_j$ for $j \in \{1, \ldots, n\}$. Then, kX holds on W_i if and only if $i \leq j$. Consequently, kX has marginal probability $l(X)$ with respect to r, and $kX_{\geq l(X)}$ is satisfied. However, r violates $kX_{\geq p}$ for every $p > l(X)$. □

We can use the construction in the proof of Theorem 5 to provide a constructive proof that p-keys enjoy Armstrong p-relations.

Theorem 6: Probabilistic keys enjoy Armstrong p-relations.

Proof: Let R be some relation schema, and let Σ be a set of p-keys over R. For all $X \subseteq R$, let $p_X := \sup\{p : \exists Y \subseteq X(kY_{\geq p} \in \Sigma \cup \{kR_{\geq 1}\})\}$. Then for all $Z \subseteq R$, Σ implies $kZ_{\geq p}$ if and only if $p \leq p_Z$. Now, let $l(X) := p_X$. Then $l(R) = p_R = 1$ and $l(XY) = p_{XY} \geq p_X = l(X)$. By Theorem 5 it follows that there is some Armstrong p-relation r, since for all $Z \subseteq R$ and all $p \in [0,1]$, Σ implies $kZ_{\geq p}$ if and only if r satisfies $kZ_{\geq p}$. □

5.3 Computation of Armstrong PC-tables

Instead of computing Armstrong p-relations as described in the proofs above, we compute PC-tables that are concise representations of Armstrong p-relations. We call these *Armstrong* PC-tables.

Recall the following standard definition from probabilistic databases [56]. A *conditional table* or *c-table*, is a tuple $CD = \langle r, W \rangle$, where r is a relation, and W assigns to each tuple t in r a finite set W_t of positive integers. The set of *world identifiers* of CD is the union of the sets W_t for all tuples t of r. Given a world identifier i of CD, the possible world associated with i is $W_i = \{t | t \in r \text{ and } i \in W_t\}$. The semantics of a c-table $CD = \langle r, W \rangle$, called *representation*, is the set $\mathcal{W}$ of possible worlds W_i where i denotes some world identifier of CD. A *probabilistic conditional database* or *PC-table*, is a pair $\langle CD, P \rangle$ where CD is a c-table, and P is a probability distribution over the set of world identifiers of CD. The semantics of a PC-table $\langle CD, P \rangle$, called the representation, is the p-relation whose set of possible worlds is the representation of CD, and the probability of each possible world W_i is defined as the probability of its world identifier. A PC-table is said to be *Armstrong* for a set Σ of p-keys, if the representation of the PC-table is an Armstrong p-relation for Σ. For example, Figure 2 shows a PC-table $\langle CD, P \rangle$ that is Armstrong for the following set of p-keys: $k\{time, zone\}_{\geq 0.65}$, $k\{rfid, time\}_{\geq 0.75}$, and $k\{rfid, zone\}_{\geq 0.35}$.

We will now describe an algorithm that computes an Armstrong PC-table for every given set Σ of p-keys. In our construction, the number of possible worlds is determined by the number of distinct probabilities that occur in Σ. For that purpose, for every given set Σ of p-keys over R and every probability $p \in [0,1]$, let

$$\Sigma_p = \{kX : \exists kX_{\geq q} \in \Sigma \wedge q \geq p\}$$

denote the *p-cut* of Σ, i.e., the set of keys over R which

Algorithm 2 Armstrong PC-table

Require: R, Σ
Ensure: Armstrong PC-table $\langle CD, P \rangle$ for Σ

```
Let p_1, ..., p_n denote the i-th smallest probabilities p_i occurring in Σ;  ▷ If p_n < 1, n ← n + 1 and p_n ← 1
p_0 ← 0;
P ← ∅;
for i = 1, ..., n do
    P ← P ∪ {(i, p_i − p_{i−1})};
    Σ_i^{-1} ← Set of anti-keys for Σ_{p_i};
Σ^{-1} ← ∅;
for all X ∈ Σ_1^{-1} ∪ ··· ∪ Σ_n^{-1} do
    Σ^{-1} ← Σ^{-1} ∪ {(X, {i : X ∈ Σ_i^{-1}})};
for all A ∈ R do
    t_0(A) ← 0;
CD ← {(t_0, {1, ..., n})};
j ← 0;
for all (X, W) ∈ Σ^{-1} do
    j ← j + 1;
    for all A ∈ R do
        t_j(A) ← { 0   , if A ∈ X
                 { j   , otherwise ;
    CD ← CD ∪ {(t_j, W)};
return ⟨CD, P⟩;
```

Fig. 5. An Armstrong PC-table

CD table

rfid	*time*	*zone*	W
w1	2pm	z1	$1, 2, 3, 4$
w1	3pm	z2	1
w2	4pm	z1	1
w3	2pm	z3	$1, 2$
w1	5pm	z1	$2, 3, 4$
w4	2pm	z1	$3, 4$
w1	2pm	z4	4

P table

W	$\mathcal{P}$
1	.35
2	.3
3	.1
4	.25

have at least marginal probability p. It is possible that Σ does not contain any p-key $kX_{\geq p}$ where $p = 1$. In this case, Algorithm 2 computes an Armstrong PC-table for Σ that contains one more possible world than the number of distinct probabilities occurring in Σ. Processing the probabilities Σ from smallest p_1 to largest p_n, the algorithm computes as possible world with probability $p_i - p_{i-1}$ (line 5) a traditional Armstrong relation for the p_i-cut Σ_{p_i}. For this purpose, the anti-keys are computed for each p_i-cut (line 6), and the set W of those worlds i is recorded for which X is an anti-key with respect to Σ_{p_i} (line 9). The CD-table contains one tuple t_0 which occurs in all possible worlds (line 12), and for each anti-key X another tuple t_j that occurs in all worlds for which X is an anti-key and that has matching values with t_0 in exactly the columns of X (lines 14-18).

Theorem 7: Let Σ denote a set of p-keys over relation schema R, and let n denote the number of distinct probabilities that occur in Σ. Algorithm 2 computes an Armstrong PC-table for Σ in which the number of possible worlds is n, if there is some p-key in Σ with lower bound 1, and $n + 1$ otherwise.

Proof: Algorithm 2 follows the proofs of Theorem 5 and Theorem 6, which construct an Armstrong PC-table of the stated size. □

We illustrate the construction on our running example.

Example 11: Recall that the p-key set Σ contains $k\{\textit{rfid}, \textit{time}\}_{\geq 0.75}$, $k\{\textit{time}, \textit{zone}\}_{\geq 0.65}$, and $k\{\textit{rfid}, \textit{zone}\}_{\geq 0.35}$. Applying Algorithm 2 to WOLVERINE and Σ may result in the Armstrong PC-table of Figure 5.

Finally, we derive some bounds on the time complexity of finding Armstrong PC-tables. Additional insight is given by our experiments in Section 6.

Let the size of an Armstrong PC-table be defined as the number of tuples that it contains. In practice, the most appealing Armstrong PC-tables for a p-key set Σ should be of minimum size. The reason is that a small number of tuples is easier to comprehend for humans. Therefore, it is a practical question to ask how many tuples a minimum-sized Armstrong PC-table requires. An Armstrong PC-table for Σ is said to be *minimum-sized* if there is no Armstrong PC-table for Σ with fewer tuples. We recall what we mean by *precisely exponential* [**?**]. Firstly, it means that there is an algorithm for computing an Armstrong PC-table table, given a set Σ of p-keys, where the running time of the algorithm is exponential in the size of $|\Sigma|$, that is, the total number of attribute occurrences in Σ. Secondly, it means that there is a set Σ of p-keys in which the number of tuples in each minimum-sized Armstrong PC-table for Σ is exponential — thus, an exponential amount of time is required in this case simply to write down the table.

Theorem 8: The time complexity to find an Armstrong PC-table for a given set Σ of p-keys over relation schema R is precisely exponential in $|\Sigma|$.

Proof: Given R and Σ as input, Algorithm 2 computes an Armstrong PC-table for Σ in time at most exponential in $|\Sigma|$. Indeed, an Armstrong relation for Σ_{p_i} can be computed in time at most exponential in $|\Sigma_{p_i}| \leq |\Sigma|$, and we require no more than $|\Sigma|$ computations of such relations.

There are cases where the number of tuples in any Armstrong PC-table for Σ over R is exponential in $|\Sigma|$. Such a case is given by $R_n = \{A_1, \ldots, A_{2n}\}$ and $\Sigma_n = \{\{A_i, A_j\}_{\geq 1} \mid 1 \leq i \leq 2 \cdot n - 1, i \text{ odd}, j = i + 1\}$ with $|\Sigma_n| = 2 \cdot n$. Every Armstrong PC-table requires $2^n + 1$ tuples, and there is only one possible world. □

It is important to note that there are also other extreme cases, in which the size of Armstrong PC-tables is logarithmic in that of the given constraint set.

Theorem 9: There are sets Σ of p-keys for which Armstrong PC-tables exist that require a number of rows that is logarithmic in $|\Sigma|$.

Proof: Let $R_n = \{A_1, \ldots, A_{2n}\}$ and $\Sigma_n = \{(X_1 \cdots X_n)_{\geq 1} : X_i \in \{A_{2i-1}, A_{2i}\} \text{ for } i = 1, \ldots, n\}$ with $|\Sigma_n| = n \cdot 2^n$. One Armstrong PC-table for Σ represents a single possible world which has $n+1$ tuples that realize the n anti-keys $R - \{A_{2i-1}, A_{2i}\}$. □

Theorem 8 and Theorem 9 show that the representation in the form of an Armstrong PC-table does not dominate the representation in the form of a constraint set, or vice versa. Our recommendation is to use both representation systems.

The computation of Armstrong PC-tables by Algorithm 2 also provides us with a computation of Armstrong instances over any other complete representation system of probabilistic relations, such as U-relations [56]. In fact, we may first apply Algorithm 2 to compute an Armstrong PC-table, and then apply a standard transformation into the other representation system. The choice of representation system may depend on its properties, such as the closure under unions of conjunctive queries for U-relations.

5.4 Profiling PC-tables with P-keys

We are now turning to the other problem illustrated in Figure 4, which is to compute the marginal probability of every key in a p-relation that a given PC-table represents.

Problem:	Profiling PC-tables with p-keys
Input:	Relation schema R PC-table that represents p-relation p over R
Output:	$\{(X, m_{kX,r}) \mid X \subset R\}$

The profiling problem of PC-tables with p-keys subsumes the profiling problem of relations with traditional keys as the special case where the PC-table represents a relation with just one possible world: In this case, the marginal probability of each p-key kX is either 0 or 1, that is, a key kX is either violated, or satisfied. Indeed, the profiling problem of relations with traditional keys has received much interest in the 1980s, e.g. [10], and recently again in the context of big data [45], [7], [11], motivated by modern applications such as data integration. However, as p-keys have only been introduced in [57], the profiling problem of PC-tables with p-keys has not been studied.

Our proposed solution to the profiling problem of PC-tables with p-keys is targeted at the elicitation of p-keys, as illustrated in Figure 4. As the computation of Armstrong PC-tables has shown, Armstrong PC-tables represent p-relations with a modest number of possible worlds. The algorithm we will present now, as well as its experimental evaluation in Section 6, show that the profiling problem of PC-tables with p-keys can be solved efficiently in this context, and also scales linearly in the number of possible worlds.

Algorithm 3 assumes that the p-relation $r = (\mathcal{W}, P)$ that is represented by the given PC-table is explicitly given. If that is not the case in our context, then the time spent on computing the p-relation is negligible in comparison to the overall complexity of the algorithm. For ease of notation we let p_X denote $m_{kX,r}$, for all $X \subseteq R$. The profiling problem can then be solved as follows: For each $X \subset R$, initialize $p_X \leftarrow 0$ (lines 1-2), and for all worlds $W \in \mathcal{W}$, add the probability $p_W = P(W)$ of W to p_X, if X contains some minimal key of W (lines 3-7). Here, a traditional key kX is a *minimal* key of world W, if kX is satisfied by W and kY is violated by W for every proper subset Y of X. The computation of the set of minimal keys of a world W is just the well-studied profiling problem of a relation W with traditional keys, and can be solved by any known solution. For example, the elegant solution in [62] (line 4) computes the minimal keys of W as the hypergraph transversals of the disagree sets in W.

Note that the exact complexity of the hypergraph transversal problem is still open, in particular the existence of an algorithm that is polynomial in the output. The exact complexity of the profiling problem of relations with traditional keys (and that of PC-tables with p-keys) remains therefore also open.

Algorithm 3 Profiling

Require: P-relation $r = (\mathcal{W}, P)$ over relation schema R
Ensure: $\{(X, m_{kX,r}) \mid X \subset R\}$
1: **for all** $X \subset R$ **do**
2: $\quad p_X \leftarrow 0$;
3: **for all** $W \in \mathcal{W}$ **do**
4: $\quad \mathcal{M}(W) \leftarrow$ Set of minimal keys on W; ▷ e.g., [62]
5: $\quad$ **for all** $X \subset R$ **do**
6: $\quad\quad$ **if** X contains some $M \in \mathcal{M}(W)$ **then**
7: $\quad\quad\quad p_X \leftarrow p_X + P(W)$;
8: **return** $\{(X, p_X) : X \subseteq R\}$;

Theorem 10: Given a p-relation $r = (\mathcal{W}, P)$ over relation schema R, Algorithm 3 computes the marginal probability of all keys over R in r.

Proof: The correctness of Algorithm 3 follows straight from the correctness of computing the set of minimal keys from a traditional relation. □

Let us illustrate Algorithm 3 on our running example.

Example 12: We apply Algorithm 3 to the p-relation from Table 1, which is represented by the PC-table in Figure 2 for example. The minimal disagree sets and the minimal keys (hypergraph transversals of the set of minimal disagree sets) of the worlds are:

W	$P(W)$	min disagree sets	min keys
W_1	0.2	$\{t\}$, $\{z\}$	$\{t, z\}$
W_2	0.45	$\{t\}$, $\{r, z\}$	$\{r, t\}$, $\{t, z\}$
W_3	0.3	$\{r\}$, $\{t, z\}$	$\{r, t\}$, $\{r, z\}$
W_4	0.05	$\{r\}$, $\{z\}$	$\{r, z\}$

The marginal probability of a given key kX is now the sum of the probabilities of those possible worlds W in which X contains some minimal key of W. For example, $k\{time, zone\} = P(W_1) + P(W_2) = 0.65$. Figure 3 illustrates the marginal probabilities of all keys.

6 Experiments

In this section we report on some experiments regarding the computational complexity of our algorithms for the visualization and discovery of probabilistic keys.

6.1 Visualization

The Armstrong construction takes as input a set Σ of randomly generated p-keys, and outputs an Armstrong PC-table for Σ. For the random generation of Σ we firstly sample n probabilities p_n from $[0, 1]$ and for each $X \subset R$, we assign a probability randomly sampled from $\{0\} \cup \{p_1, p_2, \ldots, p_n\}$. For our experiments, n was at most 15.

Figure 6 shows the number of tuples in the Armstrong PC-table as a function of applying Algorithm 2 to the exponential case from the proof of Theorem 8 (black

Fig. 6. Size of Armstrong PC-tables

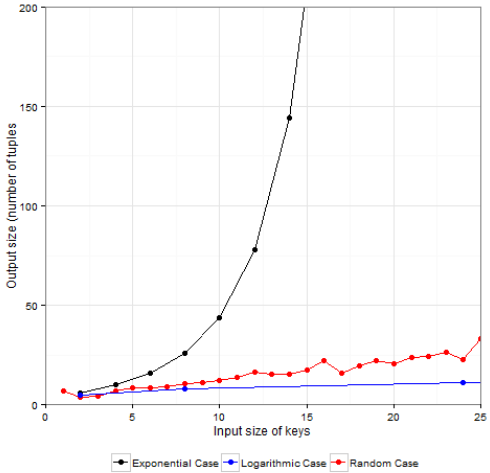
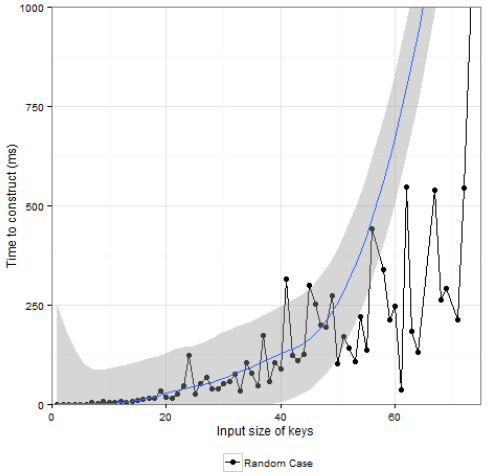


Fig. 7. Time to compute Armstrong PC-table



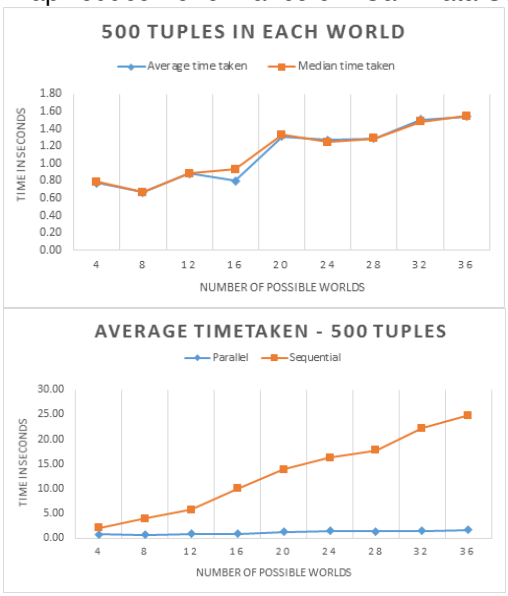
line), the logarithmic case from the proof of Theorem 9 (blue line), and the random generation (red line). The figure illustrates that the average size of an Armstrong PC-table grows linearly in the input key size. The worst-case exponential growth occurs rarely on average. This demonstrates that Armstrong PC-tables exhibit small sizes on average, making them a practical tool to acquire meaningful p-keys jointly with domain experts.

Figure 7 shows the time for computing Armstrong PC-tables from the given sets of randomly created p-keys. It shows that Armstrong PC-tables can be computed efficiently for the input sizes considered. In fact, their computation hardly ever exceeded 1 second. Figure 10 shows the graphical user interface of our visualization tool, developed in *R*. The input interface is shown on the left, and the output PC-table on the right.

6.2 Profiling

Figure 11 shows the time for profiling p-keys from the given Armstrong PC-tables we randomly created

Fig. 8. MapReduce Performance on "Car" Data Set



previously. It illustrates that the profiling problem can be solved efficiently for input sizes typical for our elicitation framework, see Figure 4. Large input sizes will require more sophisticated techniques.

6.3 MapReduce

We also applied a MapReduce implementation on a single node machine with 40 processors to the "Car" data set[1] of the UCI Machine Learning Repository. Details like buying price, maintenance price, number of doors, capacity in terms of people, luggage boot size, and estimated safety of the car are captured in six attributes. In total, this data set has 1,728 tuples with duplicates or missing data. We converted "Car" into a p-relation with rising numbers of possible worlds and 500 tuples in each world. Figure 8 shows that our algorithm for the discovery of p-keys scales linearly in the number of possible worlds, considering this number is relatively low in our elicitation framework.

7 Conclusion and Future Work

We have introduced probabilistic keys that stipulate lower bounds on the marginal probability by which keys shall hold on large volumes of uncertain data. The marginal probability of keys provides a principled mechanism to control the consistency and completeness targets for the quality of data, as shown in Figure 9.

We have established axiomatic and algorithmic tools to reason about probabilistic keys. This can minimize the overhead in using them for data quality management and query processing. These applications are effectively unlocked by developing support for identifying the right marginal probabilities by which keys should hold in a

1. http://archive.ics.uci.edu/ml/datasets/Car+Evaluation

Fig. 9. Control mechanism p

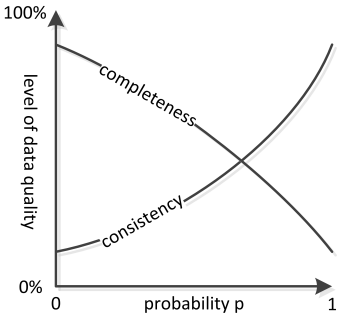
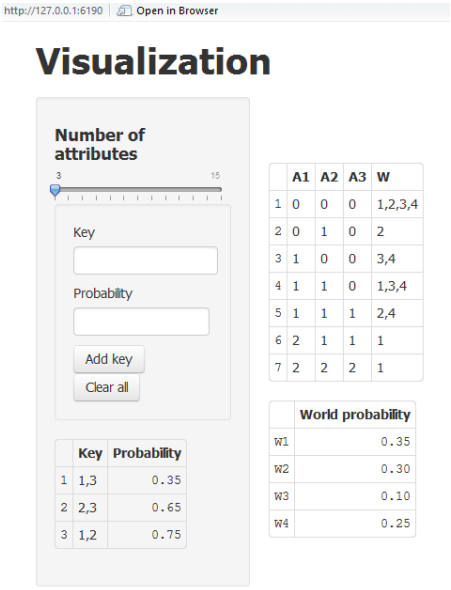


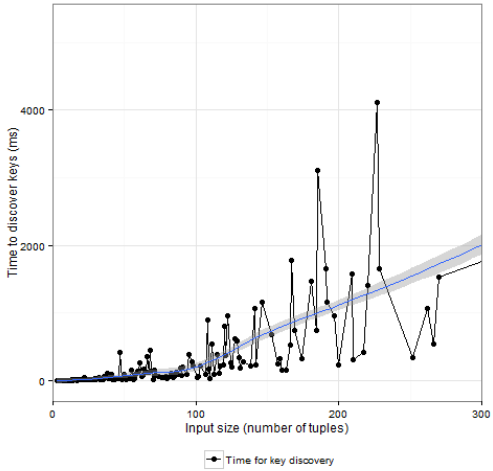
Fig. 10. GUI for Visualization



given application domain. For this challenging problem, we have developed schema- and data-driven algorithms for the use by analysts to communicate more effectively with domain experts. The schema-driven algorithm converts any set of probabilistic keys into an Armstrong PC-table that satisfies the set and violates all probabilistic keys not implied by the set. Analysts and domain experts can jointly inspect the Armstrong PC-table which points out any flaws in the current perception of marginal probabilities. The data-driven algorithm computes a profile of the probabilistic keys that a given PC-table satisfies. Such PC-tables may represent some exemplary data or result from changes to a given Armstrong PC-table in response to identifying some flaws during their inspection. Experiments confirm that the computation of Armstrong PC-tables is typically efficient, their size is small, and profiles of probabilistic keys can be efficiently computed from PC-tables of reasonable size.

In future research we will apply our algorithms to investigate empirically the usefulness of our framework for acquiring the right marginal probabilities of keys in a given application domain. This will require us to extend empirical measures from certain [46] to probabilistic data sets. Intriguing is the question whether PC-tables

Fig. 11. Times for Profiling P-keys



or p-relations are more useful. It is also interesting to raise the expressivity of probabilistic keys by allowing the stipulation of sets of marginal probabilities by which keys should hold on a probabilistic database, or other features.

Acknowledgement. This research is supported by the Marsden fund council from Government funding, administered by the Royal Society of New Zealand.

REFERENCES

[1] E. F. Codd, "A relational model of data for large shared data banks," *Commun. ACM*, vol. 13, no. 6, pp. 377–387, 1970.
[2] S. Abiteboul, R. Hull, and V. Vianu, *Foundations of Databases*. Addison-Wesley, 1995.
[3] E. F. Codd, *The Relational Model for Database Management, Version 2*. Addison-Wesley, 1990.
[4] J. Melton, "ISO/IEC 9075-2: 2003 (SQL/foundation)," ISO standard, 2003.
[5] S. Hartmann and S. Link, "The implication problem of data dependencies over SQL table definitions," *ACM Trans. Database Syst.*, vol. 37, no. 2, p. 13, 2012.
[6] S. Hartmann, M. Kirchberg, and S. Link, "Design by example for SQL table definitions with functional dependencies," *VLDB J.*, vol. 21, no. 1, pp. 121–144, 2012.
[7] A. Heise, Jorge-Arnulfo, Quiane-Ruiz, Z. Abedjan, A. Jentzsch, and F. Naumann, "Scalable discovery of unique column combinations," *PVLDB*, vol. 7, no. 4, pp. 301–312, 2013.
[8] H. Köhler, S. Link, and X. Zhou, "Discovering meaningful certain keys from incomplete and inconsistent relations," *IEEE Data Eng. Bull.*, vol. 39, no. 2, pp. 21–37, 2016.
[9] J. Liu, J. Li, C. Liu, and Y. Chen, "Discover dependencies from data - A review," *IEEE Trans. Knowl. Data Eng.*, vol. 24, no. 2, pp. 251–264, 2012.
[10] H. Mannila and K.-J. Räihä, *Design of Relational Databases*. Addison-Wesley, 1992.
[11] Y. Sismanis, P. Brown, P. J. Haas, and B. Reinwald, "GORDIAN: Efficient and scalable discovery of composite keys," in *VLDB*, 2006, pp. 691–702.
[12] K. Zellag and B. Kemme, "Consad: a real-time consistency anomalies detector," in *SIGMOD*, 2012, pp. 641–644.
[13] P. Bailis, A. Ghodsi, J. M. Hellerstein, and I. Stoica, "Bolt-on causal consistency," in *SIGMOD*, 2013, pp. 761–772.
[14] M. Arenas, L. E. Bertossi, and J. Chomicki, "Consistent query answers in inconsistent databases," in *SIGMOD*, 1999, pp. 68–79.
[15] P. Koutris and J. Wijsen, "The data complexity of consistent query answering for self-join-free conjunctive queries under primary key constraints," in *PODS*, 2015, pp. 17–29.

[16] E. Rahm and H. H. Do, "Data cleaning: Problems and current approaches," *IEEE Data Eng. Bull.*, vol. 23, no. 4, pp. 3–13, 2000.
[17] R. Fagin, P. G. Kolaitis, R. J. Miller, and L. Popa, "Data exchange: semantics and query answering," *Theor. Comput. Sci.*, vol. 336, no. 1, pp. 89–124, 2005.
[18] R. Pochampally, A. D. Sarma, X. L. Dong, A. Meliou, and D. Srivastava, "Fusing data with correlations," in *SIGMOD*, 2014, pp. 433–444.
[19] A. Calì, D. Calvanese, and M. Lenzerini, "Data integration under integrity constraints," in *Seminal Contributions to Information Systems Engineering*, 2013, pp. 335–352.
[20] F. Naumann, "Data profiling revisited," *SIGMOD Record*, vol. 42, no. 4, pp. 40–49, 2013.
[21] B. Saha and D. Srivastava, "Data quality: The other face of big data," in *ICDE*, 2014, pp. 1294–1297.
[22] L. E. Bertossi, *Database Repairing and Consistent Query Answering*. Morgan & Claypool Publishers, 2011.
[23] J. Biskup, *Security in Computing Systems - Challenges, Approaches and Solutions*. Springer, 2009.
[24] R. Fagin, "A normal form for relational databases that is based on domains and keys," *ACM Trans. Database Syst.*, vol. 6, no. 3, pp. 387–415, 1981.
[25] I. Ileana, B. Cautis, A. Deutsch, and Y. Katsis, "Complete yet practical search for minimal query reformulations under constraints," in *SIGMOD*, 2014, pp. 1015–1026.
[26] S. Abiteboul and V. Vianu, "Transactions and integrity constraints," in *PODS*, 1985, pp. 193–204.
[27] K. A. Ross, D. Srivastava, and S. Sudarshan, "Materialized view maintenance and integrity constraint checking: Trading space for time," in *SIGMOD*, 1996, pp. 447–458.
[28] H. Köhler, U. Leck, S. Link, and X. Zhou, "Possible and certain keys for SQL," *VLDB J.*, vol. 25, no. 4, pp. 571–596, 2016.
[29] H. Köhler, S. Link, and X. Zhou, "Possible and certain SQL keys," *PVLDB*, vol. 8, no. 11, pp. 1118–1129, 2015.
[30] H. Köhler and S. Link, "SQL schema design: Foundations, normal forms, and normalization," in *SIGMOD*, 2016, pp. 267–279.
[31] B. Thalheim, "On semantic issues connected with keys in relational databases permitting null values," *Elektr. Informationsverarb. Kybern.*, vol. 25, no. 1/2, pp. 11–20, 1989.
[32] J. Wijsen, "Temporal FDs on complex objects," *ACM Trans. Database Syst.*, vol. 24, no. 1, pp. 127–176, 1999.
[33] S. Abiteboul and P. C. Kanellakis, "Object identity as a query language primitive," *J. ACM*, vol. 45, no. 5, pp. 798–842, 1998.
[34] R. Wieringa and W. de Jonge, "Object identifiers, keys, and surrogates: Object identifiers revisited," *Theory and Practice of Object Systems*, vol. 1, no. 2, pp. 101–114, 1995.
[35] M. Arenas, J. Daenen, F. Neven, M. Ugarte, J. V. den Bussche, and S. Vansummeren, "Discovering XSD keys from XML data," *ACM Trans. Database Syst.*, vol. 39, no. 4, pp. 28:1–28:49, 2014.
[36] S. Hartmann and S. Link, "Efficient reasoning about a robust XML key fragment," *ACM Trans. Database Syst.*, vol. 34, no. 2, 2009.
[37] ——, "Unlocking keys for XML trees," in *ICDT*, 2007, pp. 104–118.
[38] ——, "Expressive, yet tractable XML keys," in *EDBT*, 2009, pp. 357–367.
[39] G. Lausen, "Relational databases in RDF: Keys and foreign keys," in *SWDB-ODBIS*, 2007, pp. 43–56.
[40] D. Toman and G. E. Weddell, "On keys and functional dependencies as first-class citizens in description logics," *J. Autom. Reasoning*, vol. 40, no. 2-3, pp. 117–132, 2008.
[41] W. W. Armstrong, "Dependency structures of data base relationships," in *IFIP Congress*, 1974, pp. 580–583.
[42] C. Beeri, M. Dowd, R. Fagin, and R. Statman, "On the structure of Armstrong relations for functional dependencies," *J. ACM*, vol. 31, no. 1, pp. 30–46, 1984.
[43] D. Geiger and J. Pearl, "Logical and algorithmic properties of conditional independence and graphical models," *The Annals of Statistics*, vol. 21, no. 4, pp. 2001–2021, 1993.
[44] H. Mannila and K.-J. Räihä, "Design by example: An application of armstrong relations," *J. Comput. Syst. Sci.*, vol. 33, no. 2, pp. 126–141, 1986.
[45] Z. Abedjan, L. Golab, and F. Naumann, "Profiling relational data: a survey," *VLDB J.*, vol. 24, no. 4, pp. 557–581, 2015.
[46] W.-D. Langeveldt and S. Link, "Empirical evidence for the usefulness of Armstrong relations in the acquisition of meaningful functional dependencies," *Inf. Syst.*, vol. 35, no. 3, pp. 352–374, 2010.
[47] C. Giannella and E. L. Robertson, "On approximation measures for functional dependencies," *Inf. Syst.*, vol. 29, no. 6, pp. 483–507, 2004.
[48] 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.
[49] H. Köhler, U. Leck, S. Link, and H. Prade, "Logical foundations of possibilistic keys," in *JELIA*. Springer, 2014, pp. 181–195.
[50] S. Link and H. Prade, "Possibilistic functional dependencies and their relationship to possibility theory," *IEEE Trans. Fuzzy Systems*, vol. 24, no. 3, pp. 757–763, 2016.
[51] T. K. Roblot and S. Link, "Possibilistic cardinality constraints and functional dependencies," in *ER*, 2016, pp. 133–148.
[52] 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.
[53] H. Köhler, S. Link, H. Prade, and X. Zhou, "Cardinality constraints for uncertain data," in *ER*, 2014, pp. 108–121.
[54] S. Link and H. Prade, "Relational database schema design for uncertain data," in *CIKM*, 2016, pp. 1211–1220.
[55] H. Köhler and S. Link, "Qualitative cleaning of uncertain data," in *CIKM*, 2016, pp. 2269–2274.
[56] D. Suciu, D. Olteanu, C. Ré, and C. Koch, *Probabilistic Databases*, ser. Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2011.
[57] P. Brown and S. Link, "Probabilistic keys for data quality management," in *CAiSE*. Springer, 2015, pp. 118–132.
[58] P. Brown, J. Ganesan, H. Köhler, and S. Link, "Keys with probabilistic intervals," in *ER*, 2016, pp. 164–179.
[59] T. Roblot and S. Link, "Probabilistic cardinality constraints," in *ER*, 2015, pp. 214–228.
[60] B. Thalheim, *Dependencies on relational databases*. Teubner, 1991.
[61] R. Fagin, "Horn clauses and database dependencies," *J. ACM*, vol. 29, no. 4, pp. 952–985, 1982.
[62] H. Mannila and K.-J. Räihä, "Algorithms for inferring functional dependencies from relations," *Data Knowl. Eng.*, vol. 12, no. 1, pp. 83–99, 1994.

Pieta Brown is the Manager of Data & Analytics at PwC Digital NZ. She holds a Master degree in Data Science, and a BSc in Statistics and Mathematics from the University of Auckland. Pieta has analytics experience across the Telecommunications, FMCG, Retail, Utilities and Financial Services sectors and a strong interest in applications of data science to health care.

Sebastian Link received a DSc from the University of Auckland in 2015, and a PhD in Information Systems from Massey University in 2005. He is an Associate Professor at the Department of Computer Science at the University of Auckland. His research interests include conceptual data modeling, semantics in databases, foundations of mark-up languages, and applications of discrete mathematics to computer science. Sebastian received the Chris Wallace Award for Outstanding Research Contributions in recognition of his work on the semantics of SQL and XML data. Sebastian has published more than 150 research papers, and served as a reviewer for numerous conferences and journals. He is a member of the editorial board of the journal Information Systems.