Approximate Keys and Functional Dependencies in Incomplete Databases With Limited Domains*

Munqath Al-atar[1,2], Attila Sali[1,3]

[1] Department of Computer Science and Information Theory,
Budapest University of Technology and Economics
m.attar@cs.bme.hu
[2] ITRDC, University of Kufa
munqith.alattar@uokufa.edu.iq
[3] Alfréd Rényi Institute of Mathematics
sali.attila@renyi.hu

Abstract. A possible world of an incomplete database table is obtained by imputing values from the attributes (infinite) domain to the place of NULLs. A table satisfies a possible key or possible functional dependency constraint if there exists a possible world of the table that satisfies the given key or functional dependency constraint. A certain key or functional dependency is satisfied by a table if all of its possible worlds satisfy the constraint. Recently, an intermediate concept was introduced. A strongly possible key or functional dependency is satisfied by a table if there exists a strongly possible world that satisfies the key or functional dependency. A strongly possible world is obtained by imputing values from the active domain of the attributes, that is from the values appearing in the table. In the present paper, we study approximation measures of strongly possible keys and FDs. Measure g_3 is the ratio of the minimum number of tuples to be removed in order that the remaining table satisfies the constraint. We introduce a new measure g_5, the ratio of the minimum number of tuples to be added to the table so the result satisfies the constraint. g_5 is meaningful because the addition of tuples may extend the active domains. We prove that if g_5 can be defined for a table and a constraint, then the g_3 value is always an upper bound of the g_5 value. However, the two measures are independent of each other in the sense that for any rational number $0 \leq \frac{p}{q} < 1$ there are tables of an arbitrarily large number of rows and a constant number of columns that satisfy $g_3 - g_5 = \frac{p}{q}$. A possible world is obtained usually by adding many new values not occurring in the table before. The measure g_5 measures the smallest possible distortion of the active domains.

Keywords: Strongly possible functional dependencies, Strongly possible keys, incomplete databases, data Imputation, Approximate functional dependencies, approximate keys.

1 Introduction

The information in many industrial and research databases may usually be incomplete due to many reasons. For example, databases related to instrument maintenance, medical

* Research of the second author was partially supported by the National Research, Development and Innovation Office (NKFIH) grants K–116769 and SNN-135643. This work was also supported by the BME- Artificial Intelligence FIKP grant of EMMI (BME FIKP-MI/SC) and by the Ministry of Innovation and Technology and the National Research, Development and Innovation Office within the Artificial Intelligence National Laboratory of Hungary.

applications, and surveys [10]. This makes it necessary to handle the cases when some information missing from a database and are required by the user. Imputation (filling in) is one of the common ways to handle the missing values [20].

A new approach for imputing values in place of the missing information was introduced in [2], to achieve complete data tables, using only information already contained in the SQL table attributes (which are called the active domain of an attribute). Any total table obtained in this way is called a strongly possible world. We use only the data shown on the table to replace the missing information because in many cases there is no proper reason to consider any other attribute values than the ones that already exist in the table. Using this concept, new key and functional dependency constraints called strongly possible keys (spKeys) and strongly possible functional dependencies (spFDs) were defined in [5, 3] that are satisfied after replacing any missing value (`NULL`) with a value that is already shown in the corresponding attribute. In section 2, we provide the formal definitions of spKeys and spFDs.

The present paper continues the work started in [5], where an approximation notion was introduced to calculate how close any given set of attributes can be considered as a key, even when it does not satisfy the conditions of spKeys. This is done by calculating the minimum number of tuples that need to be removed from the table so that the spKey constraint holds.

Tuple removal may be necessary because the active domains do not contain enough values to be able to replace the `NULL` values so that the tuples are pairwise distinct on a candidate key set of attributes K. In the present paper, we introduce approximation measures of spKeys and spFDs by adding tuples. Adding a tuple with new unique values will add more values to the attributes' active domains, thus some unsatisfied constraints may get satisfied. For example, Car_Model and $DoorNo$ is designed to form a key in the Cars Types table shown in Table 1 but the table does not satisfy the spKey $sp\langle Car\_Model, DoorNo\rangle$. Two tuples would need to be removed, but adding a new tuple with distinct door number value to satisfy $sp\langle Car\_Model, DoorNo\rangle$ is better than removing two tuples. In addition to that, we know that the car model and door number determines the engine type, then the added tuple can also have a new value in the $DoorNo$ attribute so that the table satisfy $(Car\_Model, DoorNo) \rightarrow_{sp} Engine\_Type$ rather than removing other two tuples.

Car_Model	**Door No**	**Engine_Type**
BMW I3	4 doors	$\perp$
BMW I3	$\perp$	electric
Ford explorer	$\perp$	V8
Ford explorer	$\perp$	V6

Table 1: Cars Types Incomplete Table

Adding tuples with new values provides more values in the active domains used to satisfy the spKey. But if the total part of the table does not satisfy the key, then it is useless to add more values to the active domain. Thus, we assume throughout this paper that the K-total part of the table satisfies the spKey $sp\langle K\rangle$ constraint, and that the X-total part satisfies the spFD constraint $X \rightarrow_{sp} Y$ (for exact definitions see Section 2). The interaction between spFDs and spKeys is studied in [1]. We also assume that every attribute has at least one non-null value (so that the active domain is not the empty set) and we have at least 2 attributes in the key set K since it was observed in [5] that a single attribute can only be an spKey if the table does not contain `NULL` in it.

The main objectives of this paper are:

- Extend the g_3 measure defined for spKeys in [5] to spFDs.
- Propose a new approximation measure for spKeys and spFDs called g_5, that adopt adding tuples with new values to the tables that violate the constraints.
- Compare the newly proposed measure g_5 with the earlier introduced measure g_3 and show that adding new tuples is more effective than removing violating ones.
- Nevertheless, g_3 and g_5 are independent of each other.

It is important to observe the difference between possible worlds and strongly possible worlds. The former one was defined and studied by several sets of authors, for example in [18, 9, 28]. In possible worlds, any value from the usually countably infinite domain of the attribute can be imputed in place of `NULL`s. This allows an infinite number of worlds to be considered. By taking the newly introduced active domain values given by the added tuples and minimizing the number of the tuples added, we sort of determine a minimum world that satisfies the constraints and contains an spWorld allowed by the original table given.

The paper is organized as follows. Section 2 gives the basic definitions and notations. Some related work and research results are discussed in section 3. The approximation measures for spKeys and spFDs are provided in Sections 4 and 5 respectively. And finally, the conclusions and the future directions are explained in Section 6.

2 Basic Definitions

Let $R = \{A_1, A_2, \ldots A_n\}$ be a relation schema. The set of all the possible values for each attribute $A_i \in R$ is called the domain of A_i and denoted as $D_i = dom(A_i)$ for i = 1,2,... n. Then, for $X \subseteq R$, then $D_X = \prod_{\forall A_i \in K} D_i$.

An instance $T = (t_1, t_2, \ldots t_s)$ over R is a list of tuples such that each tuple is a function $t : R \rightarrow \bigcup_{A_i \in R} dom(A_i)$ and $t[A_i] \in dom(A_i)$ for all A_i in R. By taking a list of tuples we use the bag semantics that allows several occurrences of the same tuple. Usage of the bag semantics is justified by that SQL allows multiple occurrences of tuples. Of course, the order of the tuples in an instance is irrelevant, so mathematically speaking we consider a multiset of tuples as an instance. For a tuple $t_r \in T$ and $X \subset R$, let $t_r[X]$ be the restriction of t_r to X.

It is assumed that $\perp$ is an element of each attribute's domain that denotes missing information. t_r is called V-total for a set V of attributes if $\forall A \in V$, $t_r[A] \neq \perp$. Also, t_r is a total tuple if it is R-total. t_1 and t_2 are weakly similar on $X \subseteq R$ denoted as $t_1[X] \sim_w t_2[X]$ defined by Köhler et.al. [17] if

$$\forall A \in X \quad (t_1[A] = t_2[A] \text{ or } t_1[A] = \perp \text{ or } t_2[A] = \perp).$$

Furthermore, t_1 and t_2 are strongly similar on $X \subseteq R$ denoted by $t_1[X] \sim_s t_2[X]$ if

$$\forall A \in X \quad (t_1[A] = t_2[A] \neq \perp).$$

For the sake of convenience we write $t_1 \sim_w t_2$ if t_1 and t_2 are weakly similar on R and use the same convenience for strong similarity. Let $T = (t_1, t_2, \ldots t_s)$ be a table instance over

R. Then, $T' = (t'_1, t'_2, \dots t'_s)$ is a possible world of T, if $t_i \sim_w t'_i$ for all $i = 1, 2, \dots s$ and T' is completely `NULL`-free. That is, we replace the occurrences of $\perp$ with a value from the domain D_i different from $\perp$ for all tuples and all attributes. A active domain of an attribute is the set of all the distinct values shown under the attribute except the `NULL`. Note that this was called the visible domain of the attribute in papers [2, 3, 5, 1].

Definition 2.1. *The active domain of an attribute A_i (VD_i^T) is the set of all distinct values except $\perp$ that are already used by tuples in T:*

$$VD_i^T = \{t[A_i] : t \in T\} \setminus \{\perp\} \text{ for } A_i \in R.$$

To simplify notation, we omit the upper index T if it is clear from the context what instance is considered.

Then the VD_1 in Table 2 is {Mathematics, Datamining}. The term active domain refers to the data that already exist in a given dataset. For example, if we have a dataset with no information about the definitions of the attributes' domains, then we use the data itself to define their own structure and domains. This may provide more realistic results when extracting the relationship between data so it is more reliable to consider only what information we have in a given dataset.

While a possible world is obtained by using the domain values instead of the occurrence of `NULL`, a strongly possible world is obtained by using the active domain values.

Definition 2.2. *A possible world T' of T is called a strongly possible world (spWorld) if $t'[A_i] \in VD_i^T$ for all $t' \in T'$ and $A_i \in R$.*

The concept of strongly possible world was introduced in [2]. A strongly possible worlds allow us to define strongly possible keys (spKeys) and strongly possible functional dependencies (spFDs).

Definition 2.3. *A strongly possible functional dependency, in notation $X \to_{sp} Y$, holds in table T over schema R if there exists a strongly possible world T' of T such that $T' \models X \to Y$. That is, for any $t'_1, t'_2 \in T'$ $t'_1[X] = t'_2[X]$ implies $t'_1[Y] = t'_2[Y]$. The set of attributes X is a strongly possible key, if there exists a strongly possible world T' of T such that X is a key in T', in notation $sp\langle X \rangle$. That is, for any $t'_1, t'_2 \in T'$ $t'_1[X] = t'_2[X]$ implies $t'_1 = t'_2$.*

Note that this is not equivalent with spFD $X \to_{sp} R$, since we use the bag semantics. For example, {Course Name, Year} is a strongly possible key of Table 2 as the strongly possible world in Table 3 shows it.

Course Name	**Year**	**Lecturer**	**Credits**	**Semester**
Mathematics	2019	$\perp$	5	1
Datamining	2018	Sarah	7	$\perp$
$\perp$	2019	Sarah	$\perp$	2

Table 2: Incomplete Dataset

If $T = \{t_1, t_2, \dots, t_p\}$ and $T' = \{t'_1, t'_2, \dots, t'_p\}$ is an spWorld of it with $t_i \sim_w t'_i$, then t'_i is called an sp-extension or in short an extension of t_i. Let $X \subseteq R$ be a set of attributes and let $t_i \sim_w t'_i$ such that for each $A \in R$: $t'_i[A] \in VD(A)$, then $t'_i[X]$ is an strongly possible extension of t_i on X (sp-extension)

Course Name	Year	Lecturer	Credits	Semester
Mathematics	2019	Sarah	5	1
Datamining	2018	Sarah	7	2
Datamining	2019	Sarah	7	2

Table 3: Complete Dataset

3 Related Work

Giannella et al. [11] measure the approximate degree of functional dependencies. They developed the IFD approximation measure and compared it with the other two measures: g_3 (minimum number of tuples need to be removed so that the dependency holds) and τ (the probability of a correct guess of an FD satisfaction) introduced in [16] and [12] respectively. They developed analytical bounds on the measure differences and compared these measures analysis on five datasets. The authors show that when measures are meant to define the knowledge degree of X determines Y (prediction or classification), then IFD and τ measures are more appropriate than g_3. On the other hand, when measures are meant to define the number of "violating" tuples in an FD, then, g_3 measure is more appropriate than IFD and τ. This paper extends the earlier work of [5] that utilized the g_3 measure for spKeys by calculating the minimum number of tuples to be removed from a table so that an spKey holds if it is not. The same paper proposed the g_4 measure that is derived from g_3 by emphasizing the effect of each connected component in the table's corresponding bipartite graph (where vertices of the first class of the graph represent the table's tuples and the second class represent all the possible combinations of the attributes' active domains). In this paper, we propose a new measure g_5 to approximate FDs by adding new tuples with unique values rather than deleting tuples as in g_3.

Several other researchers worked on approximating FDs in the literature. King et al. [15] provided an algorithmic method to discover functional and approximate functional dependencies in relational databases. The method provided is based upon the mathematical theory of partitions of row identification numbers from the relation, then determining non-trivial minimal dependencies from the partitions. They showed that the operations that need to be done on partitions are both simple and fast.

In [26], Varkonyi et al. introduced a structure called Sequential Indexing Tables (SIT) to detect an FD regarding the last attribute in their sequence. SIT is a fast approach so it can process large data quickly. The structure they used does not scale efficiently with the number of the attributes and the sizes of their domains, however. Other methods, such as TANE and FastFD face the same problem [23]. TANE was introduced by Huhtala [13] to discover functional and approximate dependencies by taking into consideration partitions and deriving valid dependencies from these partitions in a breadth-first or level-wise manner.

Bra, P. De, and Jan Paredaens gave a new decomposition theory for functional dependencies in [8]. They break up a relation into two subrelations whose union is the given relation and a functional dependency that holds in one subrelation is not in the other.

In [25], Tusor et al. presented the Parallelized Sequential Indexing Tables method that is memory-efficient for large datasets to find exact and approximate functional dependencies. Their method uses the same principle of Sequential Indexing Tables in storing data, but their training approach and operation are different.

Pyro is an algorithm to discover all approximate FDs in a dataset presented by Kruse [19]. Pyro verifies samples of agree sets and prunes the search spaces with the discovered FDs. On the other hand, based on the concept of "agree sets", Lopes et al. [22] developed an algorithm to find a minimum cover of a set of FDs for a given table by applying the so-called "Luxenburger basis" to develop a basis of the set of approximate FDs in the table.

Simovici et al. [24] provide an algorithm to find purity dependencies such that, for a fixed right-hand side (Y), the algorithm applies a level-wise search on the left-hand sides (X) so that $X \to Y$ has a purity measure below a user-defined threshold. Other algorithms were proposed in [14, 21] to discover all FDs that hold in a given table by searching through the lattice of subsets of attributes.

In [27], Jef Wijsen summarizes and discusses some theoretical developments and concepts in Consistent query answering CQA (when a user queries a database that is inconsistent with respect to a set of constraints). Database repairing was modeled by an acyclic binary relation $\leq_{db}$ on the set of consistent database instances, where $r_1 \leq_{db} r_2$ means that r_1 is at least as close to db as r_2. One possible distance is the number of tuples to be added and/or removed. In addition to that, Bertossi studied the main concepts of database repairs and CQA in [6], and emphasis on tracing back the origin, motivation, and early developments. J. Biskup and L. Wiese present and analyze an algorithm called preCQE that is able to correctly compute a solution instance, for a given original database instance, that obeys the formal properties of inference-proofness and distortion minimality of a set of appropriately formed constraints in [7].

4 SPKey Approximation

In [5], the authors studied strongly possible keys, and the main motivation is to uniquely identify tuples in incomplete tables, if it is possible, by using the already shown values only to fill up the occurrences of `NULL`s. Consider the relational schema $R =$ and $K \subseteq R$. Furthermore, let T be an instance over R with `NULL`s. Let T' be the set of total tuples $T' = \{t' \in \Pi_{i=1}^{b} VD_i^T : \exists t \in T \text{ such that } t[K] \sim_w t'[K]\}$, furthermore let $G = (T, T'; E)$ be the bipartite graph, called the <u>K-extension graph of T</u>, defined by $\{t, t'\} \in E \iff t[K] \sim_w t'[K]$. Finding a matching of G that covers all the tuples in T (if exists) provides the set of tuples in T' to replace the incomplete tuples in T with, to verify that K is an spKey. A polynomial-time algorithm was given in [3] to find such matching. It is a non-trivial application of the well-known matching algorithms, as $|T'|$ is usually an exponential function of the size of the input table T.

The Approximate Strongly Possible Key (ASP Key) was defined in [5] as follows.

Definition 4.1. *Attribute set K is an approximate strongly possible key of ratio a in table T, in notation $asp^-_a \langle K \rangle$, if there exists a subset S of the tuples T such that $T \setminus S$ satisfies $sp \langle K \rangle$, and $|S|/|T| \leq a$. The minimum a such that $asp^-_a \langle K \rangle$ holds is denoted by $g_3(K)$.*

The measure $g_3(K)$ represents the approximation which is the ratio of the number of tuples needed to be removed over the total number of tuples so that $sp \langle K \rangle$ holds. The measure $g_3(K)$ has a value between 0 and 1, and it is exactly 0 when $sp \langle K \rangle$ holds in T, which means we don't need to remove any tuples. For this, we used the g_3 measure introduced in [16], to determine the degree to which ASP key is approximate. For example, the g_3 measure of $sp\langle X \rangle$ on Table 4 is 0.5, as we are required to remove two out of four tuples to satisfy the key constraint as shown in Table 5.

It was shown in [5] that the g_3 approximation measure for strongly possible keys satisfies

$$g_3(K) = \frac{|T| - \nu(G)}{|T|}.$$

where $\nu(G)$ denotes the maximum matching size in the K-extension graph G. The smaller value of $g_3(K)$, the closer K is to being an spKey.

For the bipartite graph G defined above, let $\mathscr{C}$ be the collection of all the connected components in G that satisfy the spKey, i.e. for which there exists a matching that covers all tuples in the set ($\forall_{C\in\mathscr{C}}\ \nexists X \subseteq C \cap T$ such that $|X| > N(X)$ by Hall's Theorem). Let $D \subseteq G$ be defined as $D = G \setminus \bigcup_{\forall C\in\mathscr{C}} C$, and let $\mathscr{C}'$ be the set of connected components of D. Let V_C denote the set of vertices in a connected component C. The approximation measure of strongly possible keys may be more appropriate by considering the effect of each connected component in the bipartite graph on the matching. We consider the effect of the components of $\mathscr{C}$ to get doubled in the approximation measure, as these components represent that part of the data that do not require tuple removal. So a derived version of the g_3 measure was proposed and named g_4 considering these components' effects,

$$g_4(K) = \frac{|T| - (\sum_{C\in\mathscr{C}}(|V_C|) + \sum_{C'\in\mathscr{C}'} \nu(C'))}{|T| + \sum_{C\in\mathscr{C}} |V_C|}.$$

Furthermore, it was proved that for a set of attributes K in any table, we have either $g_3(K) = g_4(K)$ or $1 < g_3(K)/g_4(K) < 2$. Moreover, there exist tables of an arbitrarily large number of tuples with $g_3(K)/g_4(K) = \frac{p}{q}$ for any rational number $1 \leq \frac{p}{q} < 2$.

In this paper, we extend our investigation on approximating spKeys by considering adding new tuples instead of removing them to satisfy an spKey if possible. Removing a non-total tuple t_1 means that there exist another total and/or non-total tuple(s) that share the same strongly possible extension with t_2. The following proposition shows that we can always remove only non-total tuples if the total part of the table satisfies the key.

Proposition 4.1. *Let T be an instance over schema R and let $K \subseteq R$. If the K-total part of the table T satisfies the key $sp\langle K\rangle$, then there exists a minimum set of tuples U to be removed that are all non-K-total so that $T \setminus U$ satisfies $sp\langle K\rangle$.*

Proof. Observe that a minimum set of tuples to be removed is $T \setminus X$ for a subset X of the set of vertices (tuples) covered by a particular maximum matching of the K-extension graph. Let M be a maximum matching, and assume that t_1 is total and not covered by M. Then, the unique neighbour t_1' of t_1 in T' is covered by an edge (t_2, t_1') of $\mathcal{M}$. Then t_2 is non-total since the K-total part satisfies $sp\langle K\rangle$, so we replace the edge (t_2, t') by the edge (t_1, t') to get matching M_1 of size $|M_1| = |M|$, and M_1 covers one more total tuple. Repeat this until all total tuples are covered.

4.1 Measure g_5 for spKeys

The g_3 approximation measure for spKeys was introduced in [5]. In this section, we introduce a new approximation measure for spKeys. As we consider the active domain to be the source of the values to replace each null with, adding a new tuple to the table may increase the number of the values in the active domain of an attribute. for example, consider Table 4, the active domain of the attribute X_1 is $\{2\}$ and it changed to $\{2, 3\}$ after adding a tuple with new values as shown in Table 6.

X	
X_1	X_2
$\perp$	1
2	$\perp$
2	$\perp$
2	2

Table 4: Incomplete Table to measure $sp\langle X\rangle$

X	
X_1	X_2
$\perp$	1
2	2

Table 5: The table after removing $(asp^{-}_{a}\langle X\rangle)$

X	
X_1	X_2
$\perp$	1
2	$\perp$
2	$\perp$
2	2
3	3

Table 6: The table after adding $(asp^{+}_{b}\langle X\rangle)$

In the following definition, we define the g_5 measure as the ratio of the minimum number of tuples that need to be added over the total number of tuples to have the spKey satisfied.

Definition 4.2. *Attribute set K is an add-approximate strongly possible key of ratio b in table T, in notation $asp^{+}_{b}\langle K\rangle$, if there exists a set of tuples S such that the table TS satisfies $sp\langle K\rangle$, and $|S|/|T| \leq b$. The minimum b such that $asp^{+}_{b}\langle K\rangle$ holds is denoted by $g_5(K)$.*

The measure $g_5(K)$ represents the approximation which is the ratio of the number of tuples needed to be added over the total number of tuples so that $sp\langle K\rangle$ holds. The value of the measure $g_3(K)$ ranges between 0 and 1, and it is exactly 0 when $sp\langle K\rangle$ holds in T, which means we do not have to add any tuple. For example, the g_5 measure of $sp\langle X\rangle$ on Table 4 is 0.25, as it is enough to add one tuple to satisfy the key constraint as shown in Table 6.

Let T be a table and $U \subseteq T$ be the set of the tuples that we need to remove so that the spKey holds in T, i.e, we need to remove $|U|$ tuples, while by adding a tuple with new values, we may make more than one of the tuples in U satisfy the spKey using the new added values for their `NULL`s. In other words, we may need to add a fewer number of tuples than the number of tuples we need to remove to satisfy an spKey in the same given table. For example, Table 4 requires removing two tuples to satisfy $sp\langle X\rangle$, while adding one tuple is enough.

On the other hand, one may think about mixed modification of both adding and deleting tuples for Keys approximation, by finding the minimum number of tuples needs to be either added or removed. If first the additions are performed, then after that by Proposition 4.1, it is always true that we can remove only non-total tuples; then, instead of any tuple removal, we may add a new tuple with distinct values. Therefore, mixed modification in that way would not change the approximation measure, as it is always equivalent to tuples addition only. However, if the order of removals and additions count, then it is a topic of further research whether the removals can be substituted by additions.

The values of the two measures, g_3 and g_5, range between 0 and 1, and they are both equal to 0 if the spKey holds (we do not have to add or remove any tuples). Proposition 4.2 proves that the value of g_3 measure is always larger than or equal to the value of g_5 measure.

Proposition 4.2. *For any $K \subseteq R$ with $|K| \geq 2$, we have $g_3(K) \geq g_5(K)$.*

Proof. Indeed, we can always remove non-total tuples for g_3 by Proposition 4.1. Let the tuples to be removed be $U = \{t_1, t_2, \ldots t_u\}$. Assume that T^* is an spWorld of $T \setminus U$, which certifies that $T \setminus U \models sp\langle K\rangle$ For each tuple $t_i \in U$, we add tuple $t'_i = (z_i, z_i, \ldots, z_i)$ where z_i is a value that does not occur in any other tuple originally of T or added. The purpose of adding t'_i is twofold. First it is intended to introduce a completely new active domain value for each attribute. Second, their special structure ensures that they will never agree with any other tuple in the spWorld constructed below for the extended instance. Let t_i" be a tuple such that exactly one `NULL` in K of t_i is replaced by z_i, any other `NULL`s of t_i are imputed by values from the original active domain of the attributes. It is not hard to see that tuples in $T^* \cup \{t'_1, t'_2 \ldots, t'_u\} \cup \{t_1", t_2" \ldots, t_u"\}$ are pairwise distinct on K.

According to Proposition 4.2 we have $0 \leq g_3(K) - g_5(K) < 1$ and the difference is a rational number. What is not immediate is that for any rational number $0 \leq \frac{p}{q} < 1$ there exist a table T and $K \subseteq R$ such that $g_3(K) - g_5(K) = \frac{p}{q}$ in table T.

Proposition 4.3. *Let $0 \leq \frac{p}{q} < 1$ be a rational number. Then there exists a table T with an arbitrarily large number of rows and $K \subseteq R$ such that $g_3(K) - g_5(K) = \frac{p}{q}$ in table T.*

Proof. We may assume without loss of generality that $K = R$, since $T' \models sp\langle K\rangle$ if and only if we can make the tuples pairwise distinct on K by imputing values from the active domains, that is values in $R \setminus K$ are irrelevant. Let T be the following $q \times (p+2)$ table (with $x = q - p - 1$).

$$T = \begin{array}{c} \left.\begin{array}{c} 1\ 1\ 1\ \ldots\ 1 \\ 1\ 1\ 1\ \ldots\ 2 \\ \vdots \\ 1\ 1\ 1\ \ldots\ x \end{array}\right\} q-p-1 \\ \left.\begin{array}{c} \perp\ 1\ \ldots\ 1\ 1 \\ 1\ \perp\ \ldots\ 1\ 1 \\ \ddots \\ 1\ 1\ \ldots \perp 1 \end{array}\right\} p+1 \end{array} \tag{1}$$

Since the active domain of the first $p + 1$ attributes is only $\{1\}$, we have to remove $p + 1$ rows so $g_3(K) = \frac{p+1}{q}$. On the other hand it is enough to add one new row $(2, 2, \ldots, 2, q - p)$ so $g_5(K) = \frac{1}{q}$. Since $\frac{p}{q} = \frac{cp}{cq}$ for any positive integer c, the number of rows in the table could be arbitrarily large.

The tables constructed in the proof of Proposition 4.3 have an arbitrarily large number of rows, however, the price for this is that the number of columns is also not bounded. The question arises naturally whether there are tables with a fixed number of attributes but with an arbitrarily large number of rows that satisfy $g_3(K) - g_5(K) = \frac{p}{q}$ for any rational number $0 \leq \frac{p}{q} < 1$? The following theorem answers this problem.

Theorem 4.1. *Let $0 \leq \frac{p}{q} < 1$ be a rational number. Then there exist tables over schema $\{A_1, A_2\}$ with arbitrarily large number of rows, such that $g_3(\{A_1, A_2\}) - g_5(\{A_1, A_2\}) = \frac{p}{q}$.*

Proof. The proof is divided into three cases according to whether $\frac{p}{q} < \frac{1}{2}$, $\frac{p}{q} = \frac{1}{2}$ or $\frac{p}{q} > \frac{1}{2}$. In each case, the number of rows of the table will be an increasing function of q and one just has to note that q can be chosen arbitrarily large without changing the value of the fraction $\frac{p}{q}$.

Case $\frac{p}{q} < \frac{1}{2}$ Let $T_{<.5}$ be defined as

$$T_{<.5} = \begin{matrix} q-p-1 \begin{cases} 1 & 1 \\ 1 & 2 \\ \vdots & \vdots \\ 1 & q-p-1 \end{cases} \\ p+1 \begin{cases} \perp & \perp \\ \perp & \perp \\ \vdots & \vdots \\ \perp & \perp \end{cases} \end{matrix}$$

Clearly, $g_3(K) = \frac{p+1}{q}$, as all the tuples with NULLs have to be removed. On the other hand, if tuple $(2, q-p)$ is added, then the total number of active domain combinations is $2 \cdot (q-p)$, out of which $q-p$ is used up in the table, so there are $q-p$ possible pairwise distinct tuples to replace the NULLs. Since $\frac{p}{q} < \frac{1}{2}$, we have that $q-p \geq p+1$ so all the tuples in the $q+1$-rowed table can be made pairwise distinct. Thus, $g_3(K) - g_5(K) = \frac{p+1}{q} - \frac{1}{q}$.

Case $\frac{p}{q} = \frac{1}{2}$ Let $T_{=.5}$ be defined as

$$T_{=.5} = \begin{matrix} q-p-2 \begin{cases} 1 & 1 \\ 1 & 2 \\ \vdots & \vdots \\ 1 & q-p-2 \end{cases} \\ p+2 \begin{cases} \perp & \perp \\ \perp & \perp \\ \vdots & \vdots \\ \perp & \perp \end{cases} \end{matrix}$$

Table $T_{=.5}$ contains all possible combinations of the active domain values, so we have to remove every tuple containing NULLs, so $g_3(K) = \frac{p+2}{q}$. On the other hand, if we add just one new tuple (say $(2, q-p-1)$), then the largest number of active domain combinations is $2 \cdot (q-p-1)$ that can be achieved. There are already $q-p-1$ pairwise distinct total tuples in the extended table, so only $q-p-1 < p+2$ would be available to replace the NULLs. On the other hand, adding two new tuples, $(2, q-p-1)$ and $(3, q-p)$ creates a pool of $3 \cdot (q-p)$ combinations of active domains, which is more than $(q-p-1)+p+2$ that is needed.

Case $\frac{p}{q} > \frac{1}{2}$ Table T is defined similarly to the previous cases, but we need more careful analysis of the numbers.

$$T = \begin{matrix} b\left\{\begin{matrix} 1 & 1 \\ 1 & 2 \\ \vdots & \vdots \\ 1 & b \end{matrix}\right. \\ x\left\{\begin{matrix} \perp & \perp \\ \perp & \perp \\ \vdots & \vdots \\ \perp & \perp \end{matrix}\right. \end{matrix} \tag{2}$$

Clearly, $g_3(K) = \frac{x}{x+b}$. Let us assume that y tuples are needed to be added. The maximum number of active domain combinations is $(y+1)(y+b)$ obtained by adding tuples $(2, b+1), (3, b+2), \ldots, (y+1, y+b)$. This is enough to replace all tuples with NULLs if

$$(y+1)(y+b) \geq x+y+b. \tag{3}$$

On the other hand, $y-1$ added tuples are not enough, so

$$y(y-1+b) < x+y-1+b. \tag{4}$$

Since the total number of active domain combinations must be less than the tuples in the extended table. We have $\frac{p}{q} = g_3(K) - g_5(K) = \frac{x-y}{x+b}$ that is for some positive integer c we must have $cp = x-y$ and $cq = x+b$ if $gcd(p,q) = 1$. This can be rewritten as

$$y = x - cp\ ; \quad y + b = c(q-p)\ ; \quad b = cq - x\ ; \quad x + y + b = y + cq. \tag{5}$$

Using (5) we obtain that (3) is equivalent with

$$y \geq \frac{cp}{c(q-p)-1}. \tag{6}$$

If c is large enough then $\lceil \frac{cp}{c(q-p)-1} \rceil = \lceil \frac{p}{q-p} \rceil$ so if $y = \lceil \frac{p}{q-p} \rceil$ is chosen then (6) and consequently (3) holds. On the other hand, (4) is equivalent to

$$y < \frac{cq-1}{c(q-p)-2}. \tag{7}$$

The right hand side of (7) tends to $\frac{q}{q-p}$ as c tends to infinity. Thus, for large enough c we have $\lfloor \frac{cq-1}{c(q-p)-2} \rfloor = \lfloor \frac{q}{q-p} \rfloor$. Thus, if

$$y = \lceil \frac{p}{q-p} \rceil \leq \lfloor \frac{q}{q-p} \rfloor \tag{8}$$

and $\frac{q}{q-p}$ is not an integer, then both (3) and (4) are satisfied for large enough c. Observe that $\frac{p}{q-p} + 1 = \frac{q}{q-p}$, thus (8) always holds. Also, if $\frac{q}{q-p}$ is indeed an integer, then we have strict inequality in (8) that implies (7) and consequently (4).

5 spFD Approximation

In this section, we measure to which extent a table satisfies a Strongly Possible Functional Dependency (spFD) $X \to_{sp} Y$ if $T \not\models X \to_{sp} Y$.

Similarly to Section 4, we assume that the X-total part of the table satisfies the FD $X \to Y$, so we can always consider adding tuples. The measures g_3 and g_5 are defined analogously to the spKey case.

Definition 5.1. *For the attribute sets X and Y, $\sigma : X \to_{sp} Y$ is a remove-approximate strongly possible functional dependency of ratio a in a table T, in notation $T \models\approx_a^- X \to_{sp} Y$, if there exists a set of tuples S such that the table $T \setminus S \models X \to_{sp} Y$, and $|S|/|T| \leq a$. Then, $g_3(\sigma)$ is the smallest a such that $T \models\approx_a^- \sigma$ holds.*

The measure $g_3(\sigma)$ represents the approximation which is the ratio of the number of tuples needed to be removed over the total number of tuples so that $T \models X \to_{sp} Y$ holds.

Definition 5.2. *For the attribute sets X and Y, $\sigma : X \to_{sp} Y$ is an add-approximate strongly possible functional dependency of ratio b in a table T, in notation $T \models\approx_b^+ X \to_{sp} Y$, if there exists a set of tuples S such that the table $T \cup S \models X \to_{sp} Y$, and $|S|/|T| \leq b$. Then, $g_5(\sigma)$ is the smallest b such that $T \models\approx_b^+ \sigma$ holds.*

The measure $g_5(\sigma)$ represents the approximation which is the ratio of the number of tuples needed to be added over the total number of tuples so that $T \models X \to_{sp} Y$ holds. For example, consider Table 7. We are required to remove at least 2 tuples so that $X \to_{sp} Y$ holds, as it is easy to check that if we remove only one tuple, then $T \not\models X \to_{sp} Y$, but on the other hand, the table obtained by removing tuples 4 and 5, shown in Table 8 satisfies $X \to_{sp} Y$. It is enough to add only one tuple to satisfy the dependency as the table in Table 9 shows.

X		**Y**
X_1	X_2	
$\perp$	1	1
2	$\perp$	1
2	$\perp$	1
2	1	2
2	1	2
2	2	2

Table 7: Incomplete Table to measure $(X \to_{sp} Y)\rangle$

X		**Y**
X_1	X_2	
$\perp$	1	1
2	$\perp$	1
2	$\perp$	1
2	2	2

Table 8: The table after removing $(_a^- X \to_{sp} Y)$

X		**Y**
X_1	X_2	
$\perp$	1	1
2	$\perp$	1
2	$\perp$	1
2	1	2
2	1	2
2	2	2
3	3	3

Table 9: The table after adding $(_b^+ X \to_{sp} Y)$

5.1 The Difference of g3 and g5 for spFDs

The same table may get different approximation measure values for g_3 and g_5. For example, the g_3 approximation measure for Table 7 is 0.334 (it requires removing at least 2 tuples out of 6), while its g_5 approximation measure is 0.167 (it requires adding at least one tuple with new values).

The following theorem proves that it is always true that the g_3 measure value of a table is greater than or equal to the g_5 for spFDs.

Theorem 5.1. *Let T be a table over schema R, $\sigma : X \to_{sp} Y$ for some $X, Y \subseteq R$. Then $g_3(\sigma) \geq g_5(\sigma)$.*

The proof is much more complicated than the one in the case of spKeys, because we cannot assume that there always exists a minimum set of non-total tuples to be removed for g_3, as the table in Table 10 shows. In this table the third tuple alone forms a minimum set of tuples to be removed to satisfy the dependency and it has no NULL.

X		**Y**
X_1	X_2	
1	$\perp$	1
1	$\perp$	1
1	1	2
1	1	$\perp$
1	2	3

Table 10: X-total tuple needs to be removed

From that table, we need to remove the third row to have $X \to_{sp} Y$ satisfied. Let us note that adding row $(3, 3, 3)$ gives the same result, so $g_3(X \to_{sp} Y) = g_5(X \to_{sp} Y) = 1$. However, there exist no spWorlds that realize the g_3 and g_5 measure values and agree on those tuples that are not removed for g_3.

Proof. of Theorem 5.1 Without loss of generality, we may assume that $X \cap Y = \emptyset$, because $T \models X \to_{sp} Y \iff T \models X \setminus Y \to_{sp} Y \setminus X$. Also, it is enough to consider attributes in $X \cup Y$. Let $U = \{t_1, t_2, \ldots, t_p\}$ be a minimum set of tuples to be removed from T. Let T' be the spWorld of $T \setminus U$ that satisfies $X \to Y$. Let us assume that $t_1, \ldots t_a$ are such that $t_i[X]$ is not total for $1 \leq i \leq a$. Furthermore, let $t_{a+1}[X] = \ldots = t_{j_1}[X]$, $t_{j_1+1}[X] = \ldots = t_{j_2}[X]$, …, $t_{j_f+1}[X] = \ldots = t_p[X]$ be the maximal sets of tuples that have the same total projection on X. We construct a collection of tuples $\{s_1, \ldots s_{a+f+1}\}$, together with an spWorld T^* of $T \cup \{s_1, \ldots, s_{a+f+1}\}$ that satisfies $X \to Y$ as follows.

Case 1. $1 \leq i \leq a$. Let z_i be a value not occurring in T neither in every tuple s_j constructed so far. Let $s_i[A] = z_i$ for $\forall A \in X$ and $s_i[B] = t_i[B]$ for $B \in R \setminus X$. The corresponding sp-extensions $s_i^*, t_i^* \in T^*$ are given by setting $s_i^*[B] = t_i^*[B] = \beta$ where $\beta \in VD_B$ arbitrarily fixed if $t_i[B] = \perp$ in case $B \in R \setminus X$, furthermore $t_i^*[A] = z_i$ if $A \in X$ and $t_i[A] = \perp$.

Case 2. X-total tuples. For each such set $t_{j_{g-1}+1}[X] = \ldots = t_{j_g}[X]$ $(g \in \{1, 2, \ldots, f+1\})$ we construct a tuple s_{a+g}. Let $v_1^g, v_2^g, \ldots v_{k_g}^g \in T \setminus U$ be the tuples whose sp-extension $v_j^{g\prime}$ in T' satisfies $v_j^{g\prime}[X] = t_{j_g}[X]$ for $1 \leq j \leq k_g$. Let $v_1^g, v_2^g, \ldots v_\ell^g$ be those that are also X-total. Since the X-total part of the table satisfies $X \to_{sp} Y$, $t_{j_{g-1}+1}, \ldots t_{j_g}, v_1^g, v_2^g, \ldots v_\ell^g$ can be sp-extended to be identical on Y. Let us take those extensions in T^*.

Let s_{a+g} be defined as $s_{a+g}[A] = z_{a+g}$ where z_{a+g} is a value not used before for $A \in X$, furthermore $s_{a+g}[B] = v^g_{\ell+1}[B]$ for $B \in R \setminus X$. The sp-extensions are given as $v^{g*}_q[A] = z_{a+g}$ if $v^{g*}_q[A] = \perp$ and $A \in X$, otherwise $v^{g*}_q[A] = v^{g\prime}_q[A]$ for $\ell + 1 \leq q \leq k_g$. Finally, let $s^*_{a+g}[B] = v^{g\prime}_1[B]$ for $B \in R \setminus X$.

For any tuple $t \in T \setminus U$ for which no sp-extension has been defined yet, let us keep its extension in T', that is let $t^* = t'$.

Claim $T^* \models X \to_{sp} Y$. Indeed, let $t^1, t^2 \in T \cup \{s_1, \ldots, s_{a+f+1}\}$ be two tuples such that their sp-extensions in T^* agree on X, that is $t^{1*}[X] = t^{2*}[X]$. If $t^{1*}[X]$ contains a new value z_j for some $1 \leq j \leq a + f + 1$, then by definition of the sp-extensions above, we have $t^{1*}[Y] = t^{2*}[Y]$. Otherwise, either both t^1, t^2 are X-total, so again by definition of the sp-extensions above, we have $t^{1*}[Y] = t^{2*}[Y]$, or at least one of them is not X-total, and then $t^{1*} = t^{1\prime}$ and $t^{2*} = t^{2\prime}$. But in this latter case using $T' \models X \to_{sp} Y$ we get $t^{1*}[Y] = t^{2*}[Y]$.

The values g_3 and g_5 are similarly independent of each other for spFDs as in the case of spKeys.

Theorem 5.2. *For any rational number* $0 \leq \frac{p}{q} < 1$ *there exists tables with an arbitrarily large number of rows and bounded number of columns that satisfy* $g_3(\sigma) - g_5(\sigma) = \frac{p}{q}$ *for* $\sigma\colon X \to_{sp} Y$.

Proof. Consider the following table T. We clearly have $g_3(X \to_{sp} Y) = \frac{x}{x+b}$ for T as all

$T =$

X		**Y**
X_1	X_2	
1	1	1
1	2	2
$\vdots$	$\vdots$	$\vdots$
1	b	b
$\perp$	$\perp$	$b+1$
$\perp$	$\perp$	$b+2$
$\vdots$	$\vdots$	$\vdots$
$\perp$	$\perp$	$b+x$

Table 11: $g_3 - g_5 = \frac{p}{q}$

tuples with `NULL`s must be removed. On the other hand, by adding new tuples and so extending the active domains, we need to be able to make at least $x + b$ pairwise distinct combinations of X-values. If y tuples are added, then we can extend the active domains to the sizes $|VD_1| = y + 1$ and $|VD_2| = y + b$. Also, if y is the minimum number of tuples to be added, then

$$g_3(X \to_{sp} Y) - g_5(X \to_{sp} Y) = \frac{x - y}{x + b} = \frac{p}{q} \tag{9}$$

if $cp = x - y$ and $cq = x + b$ for some positive integer c. From here $y = x - cp$ and $y + b = c(q - p)$ Thus, what we need is

$$(y + 1)(y + b) = (y + 1)c(q - p) \geq cq \tag{10}$$

and, to make sure that $y - 1$ added tuples are not enough,

$$y(y + b - 1) = y(c(q - p) - 1) \leq cq - 1. \tag{11}$$

Easy calculation shows that (10) is equivalent with $y \geq \frac{p}{q-p}$, so we take $y = \left\lceil \frac{p}{q-p} \right\rceil$. On the other hand, (11) is equivalent with $y \leq \frac{cq-1}{c(q-p)-1}$. Now, similarly to Case 3 of the proof of Theorem 4.1 observe that $\frac{cq-1}{c(q-p)-1} \to \infty$ as $c \to \infty$, so, if c is large enough, then (11) holds.

5.2 Semantic Comparison of g_3 and g_5

In this section, we compare the g_3 and g_5 measures to analyze their applicability and usability for different cases. The goal is to specify when it is semantically better to consider adding or removing rows for approximation for both spFDs and spKeys.

Considering the teaching table in Table 12, we have the two strongly possible constraints $Semester\,TeacherID \to_{sp} CourseID$ and $sp\langle Semester\,TeacherID\rangle$. It requires adding one row so that $asp_a^+ \langle Semester\,TeacherID\rangle = {}_a^+ Semester\,TeacherID \to_{sp} CourseID$. But on the other hand, it requires removing 3 out of the 6 rows. Then, it would be more convenient to add a new row rather than removing half of the table, which makes the remaining rows not useful for analysis for some cases.

Adding new tuples to satisfy some violated strongly possible constraints ensures that we make the minimum changes. In addition to that, in the case of deletion, some active domain values may be removed. There are some cases where it may be more appropriate to remove rather than add tuples, however. This is to preserve semantics of the data and to avoid using values that are out of the appropriate domain of the attributes while adding new tuples with new unseen values. For example, Table 13 represents the grade records for some students in a course that imply the key ($Name$, $Group$) and the dependency $Points\,Assignment \to Result$, while both of $sp\langle \text{NameGroup}\rangle$ and $Points\,Assignment \to_{sp} Result$ are violated by the table. Then, adding one tuple with the new values (Dummy, 3, 3, Maybe, Hopeless) is enough to satisfy the two strongly possible constraints, while they can also be satisfied by removing the last two tuples. However, it is not convenient to use these new values for the attributes, since they are probably not contained in the intended domains. Hence, removing two tuples is semantically more acceptable than adding one tuple.

If g_3 is much larger than g_5 for a table, it is better to add rows than remove them. Row removal may leave only a short version of the table which may not give a useful data analysis, as is the case in Table 11. If g_3 and g_5 are close to each other, it is mostly better to add rows, but when the attributes' domains are restricted to a short-range, then it may be better to remove rows rather than adding new rows with "noise" values that are semantically not related to the meaning of the data , as is the case in Table 13.

6 Conclusion and Future Directions

Two approximation measures for spKeys and spFDs were investigated. The first one, g_3, is the ratio of the minimum number of rows to be removed, and was introduced for functional

Semester	**TeacherID**	**CourseID**
First	1	1
⊥	1	2
First	2	3
⊥	2	4
First	3	5
⊥	3	6

Table 12: Incomplete teaching table

Name	**Group**	**Points**	**Assignment**	**Result**
Bob	1	2	Submitted	Pass
Sara	1	1	Not Submitted	Fail
Alex	1	2	Not Submitted	Fail
John	1	1	Submitted	Pass
⊥	1	1	⊥	Retake
Alex	⊥	2	⊥	Retake

Table 13: Incomplete course grading table

dependencies in tables without NULL values in [11] and for spKeys in [2]. In the present paper, we extended the definition for spFDs, as well. A new measure g_5 was also introduced here, which measures the ratio of the minimum number of tuples to be added to satisfy a strongly possible constraint. This measure is only meaningful for strongly possible constraints because ordinary functional dependencies or possible functional dependencies cannot be made valid by adding tuples. However, the new tuples may extend the active domains of the attributes and hence may make some strongly possible constraints satisfied. Note that any add-approximate spKey or spFD is a possible key, respectively possible FD. Thus, the g_5 measure measures the minimum number of "extra" attribute values one has to use in a possible world satisfying the constraint.

We proved that the value of g_5 is at most as large as the value of g_3 for both spKeys and spFDs. Otherwise, however, the two measures are independent of each other, as their difference can take any non-negative rational value less than one.

The referees suggested considering tuple removal and addition concurrently, or tuple modification. If first the additions are performed, then after that by Proposition 4.1, it is always true that we can remove only non-total tuples; then, instead of any tuple removal, we may add a new tuple with distinct values. Therefore, mixed modification in that way would not change the approximation measure, as it is always equivalent to tuples addition only. However, if the order of removals and additions count, then it is a topic of further research whether the removals can be substituted by additions. Also, Proposition 4.1 is only valid for spKeys, so mixed modifications are interesting research problem for spFDs. One tuple modification can easily be replaced by one removal and one addition. The question remains open whether one can gain more with tuple modifications than the above replacement. A future research direction is to tackle algorithmic and complexity questions. It was proven in [3] that checking whether for a given subset $K \subseteq R$ and table T, $T \models sp\langle K\rangle$ holds can be decided in polynomial time. However, the questions whether $g_3(sp\langle K\rangle) \leq q$ and $g_5(sp\langle K\rangle) \leq q$ are not known to be polynomial. The problem is that we would have to check all possible tables $T' \subset T$ with $|T'|/|T| \geq 1-q$ which could mean exponentially many tables. On the other hand, it is clear that both problems, $g_3(sp\langle K\rangle) \leq q$ and $g_5(sp\langle K\rangle) \leq q$ are in NP.

The analogous question for spFDs, that is whether $T \models X \rightarrow_{sp} Y$ for a table T and subsets $X, Y \subseteq R$, is itself NP-complete [3]. This suggests that the problem of bounding the approximation measures g_3 and g_5 for spFDs is also intractable. However, it is a topic of further study to really prove it.

We studied handling missing values for Multi-valued Dependencies (spMVDs) in [4]. An interesting future research direction can be measuring approximation ratio of spMVDs.

7 Acknowledgement

The authors are indebted to the unknown referees for their careful reading of the paper. The authors are thankful for the many suggestions of improvements and calling their attention to several related works.

References

1. Munqath Al-Atar and Attila Sali. Strongly possible functional dependencies for sql. Acta Cybernetica, 2022.
2. Munqath Alattar and Attila Sali. Keys in relational databases with nulls and bounded domains. In European Conference on Advances in Databases and Information Systems, pages 33–50. Springer, 2019.
3. Munqath Alattar and Attila Sali. Functional dependencies in incomplete databases with limited domains. In International Symposium on Foundations of Information and Knowledge Systems, pages 1–21. Springer, 2020.
4. Munqath Alattar and Attila Sali. Multivalued dependencies in incomplete databases with limited domain: Properties and rules. In 16th International Miklos Ivanyi PhD & DLA Symposium, page 226, 2020.
5. Munqath Alattar and Attila Sali. Strongly possible keys for sql. Journal on Data Semantics, 9(2):85–99, 2020.
6. Leopoldo Bertossi. Database repairs and consistent query answering: Origins and further developments. In Proceedings of the 38th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, pages 48–58, 2019.
7. Joachim Biskup and Lena Wiese. A sound and complete model-generation procedure for consistent and confidentiality-preserving databases. Theoretical Computer Science, 412(31):4044–4072, 2011.
8. P De Bra and Jan Paredaens. Conditional dependencies for horizontal decompositions. In International Colloquium on Automata, Languages, and Programming, pages 67–82. Springer, 1983.
9. Ander De Keijzer and Maurice Van Keulen. A possible world approach to uncertain relational data. In Proceedings. 15th International Workshop on Database and Expert Systems Applications, 2004., pages 922–926. IEEE, 2004.
10. Alireza Farhangfar, Lukasz A Kurgan, and Witold Pedrycz. A novel framework for imputation of missing values in databases. IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, 37(5):692–709, 2007.

11. Chris Giannella and Edward Robertson. On approximation measures for functional dependencies. Information Systems, 29(6):483–507, 2004.
12. Leo A Goodman and William H Kruskal. Measures of association for cross classifications. Measures of association for cross classifications, pages 2–34, 1979.
13. Yka Huhtala, Juha Kärkkäinen, Pasi Porkka, and Hannu Toivonen. Tane: An efficient algorithm for discovering functional and approximate dependencies. The computer journal, 42(2):100–111, 1999.
14. Martti Kantola, Heikki Mannila, Kari-Jouko Räihä, and Harri Siirtola. Discovering functional and inclusion dependencies in relational databases. International journal of intelligent systems, 7(7):591–607, 1992.
15. Ronald S King and James J Legendre. Discovery of functional and approximate functional dependencies in relational databases. Journal of Applied Mathematics and Decision Sciences, 7(1):49–59, 2003.
16. Jyrki Kivinen and Heikki Mannila. Approximate inference of functional dependencies from relations. Theoretical Computer Science, 149(1):129–149, 1995.
17. Henning Köhler, Uwe Leck, Sebastian Link, and Xiaofang Zhou. Possible and certain keys for sql. The VLDB Journal, 25(4):571–596, 2016.
18. Henning Köhler, Sebastian Link, and Xiaofang Zhou. Possible and certain sql keys. Proceedings of the VLDB Endowment, 8(11):1118–1129, 2015.
19. Sebastian Kruse and Felix Naumann. Efficient discovery of approximate dependencies. Proceedings of the VLDB Endowment, 11(7):759–772, 2018.
20. Witold Lipski Jr. On databases with incomplete information. Journal of the ACM (JACM), 28(1):41–70, 1981.
21. Stéphane Lopes, Jean-Marc Petit, and Lotfi Lakhal. Efficient discovery of functional dependencies and armstrong relations. In International Conference on Extending Database Technology, pages 350–364. Springer, 2000.
22. Stéphane Lopes, Jean-Marc Petit, and Lotfi Lakhal. Functional and approximate dependency mining: database and fca points of view. Journal of Experimental & Theoretical Artificial Intelligence, 14(2-3):93–114, 2002.
23. Thorsten Papenbrock, Jens Ehrlich, Jannik Marten, Tommy Neubert, Jan-Peer Rudolph, Martin Schönberg, Jakob Zwiener, and Felix Naumann. Functional dependency discovery: An experimental evaluation of seven algorithms. Proceedings of the VLDB Endowment, 8(10):1082–1093, 2015.
24. Dan A Simovici, Dana Cristofor, and Laurentiu Cristofor. Impurity measures in databases. Acta Informatica, 38(5):307–324, 2002.
25. Balázs Tusor and Annamária R Várkonyi-Kóczy. Memory efficient exact and approximate functional dependency extraction with parsit. In 2020 IEEE 24th International Conference on Intelligent Engineering Systems (INES), pages 133–138. IEEE, 2020.
26. AR Várkonyi-Kóczy, B Tusor, and JT Tóth. A multi-attribute classification method to solve the problem of dimensionality. In Recent Global Research and Education: Technological Challenges, pages 403–409. Springer, 2017.
27. Jef Wijsen. Foundations of query answering on inconsistent databases. ACM SIGMOD Record, 48(3):6–16, 2019.
28. Esteban Zimányi and Alain Pirotte. Imperfect information in relational databases. In Uncertainty Management in Information Systems, pages 35–87. Springer, 1997.