A Formal Taxonomy to Improve Data Defect Description

João Marcelo Borovina Josko[1](✉), Marcio Katsumi Oikawa[2], and João Eduardo Ferreira[1]

[1] Institute of Mathematics and Statistics, University of São Paulo, Sao Paulo, Brazil
{jmbj,jef}@ime.usp.br

[2] Center of Mathematics, Computing and Cognition, Federal University of ABC, Santo Andre, Brazil
marcio.oikawa@ufabc.edu.br

Abstract. Data quality assessment outcomes are essential for analytical processes, especially for big data environment. Its efficiency and efficacy depends on automated solutions, which are determined by understanding the problem associated with each data defect. Despite the considerable number of works that describe data defects regarding to accuracy, completeness and consistency, there is a significant heterogeneity of terminology, nomenclature, description depth and number of examined defects. To cover this gap, this work reports a taxonomy that organizes data defects according to a three-step methodology. The proposed taxonomy enhances the descriptions and coverage of defects with regard to the related works, and also supports certain requirements of data quality assessment, including the design of semi-supervised solutions to data defect detection.

Keywords: Data defects · Dirty data · Formal taxonomy · Data quality assessment · Relational database · Big data

1 Introduction

The effects of poor data quality on the reliability of the outcomes of analytical processes are notorious, especially for big data environment. Improving data quality requires alternatives that combine procedures, methods and techniques. The Data Quality Assessment process (DQAp) provides practical inputs for choosing the most suitable alternative through its mapping of data defects. To provide a reliable outcome, this process requires *know about* data defect structures to *know how* to assess them.

Data defects descriptions provide structural understanding of the problem associated with each defect. Descriptions are relevant for different DQAp issues such as the rule definition of certain computational approaches of data assessment and the establishment of measurement-based data quality program.

© Springer International Publishing Switzerland 2016
H. Gao et al. (Eds.): DASFAA 2016 Workshops, LNCS 9645, pp. 307–320, 2016.
DOI: 10.1007/978-3-319-32055-7_25

Much literature has described data defects through hierarchical [14], formal [13] or formal-textual-example [5] models. However, the analysis of this literature shows remarkable differences in terminology, nomenclature, coverage, granularity of description, and the description model used (as also mentioned by [8]). This poor organization and description cause uncertainties on which data defects should be assessed, which their structures are, and it also hampers the ability to determine the corresponding detection approaches.

To address this situation, this work reports a taxonomy of data defects related to the *accuracy*, *completeness* and *consistency* quality dimensions. The taxonomy is characterized and categorized according to a three-step methodology and its main contribution is a major coverage of data defects and enhanced descriptions in terms of terminology, examples and mathematical formalism.

The work reported here is organized as follows: Sect. 2 reviews all related works. Section 3 presents the methodology applied to the development of the proposed taxonomy, while Sect. 4 describes the taxonomy and its basic concepts. Section 5 presents the conclusions of this work.

2 Related Works

In literature, works that describe data defects are common in certain research areas, including Information and Data Quality Management, Data Mining and Statistics. Here, these works are analysed according to how the following questions are answered: *What is the representative set of defects related to the quality dimensions of accuracy, completeness and consistency? What is the problem structure behind each defect?*

Certain works describe data defects as a complementary topic for their main subject of interest. This approach by [3,4,15] leads to ambiguous structure descriptions of the data defects. Moreover, the data defects representativeness is assigned by common sense within a context.

In contrast, the data defects issues are relevant for Data Profiling [12], Statistics [16] and Data Cleaning [5] works. Such works expose the data defect structure through the combination of textual, instance-based examples and formal resources. However, they cover few defects.

Lastly, taxonomies intend to provide reasonable descriptions aligned to a broad coverage and classification of data defects. However, a review of state-of-art taxonomies [1,2,7,9,10,13,14] reveals heterogeneous descriptions and coverage. This situation is caused by the *degree of accuracy afforded by the defect definition, terminology* and *absence of theoretical support on defect selection.*

The *defect description model* determines the degree of accuracy afforded by the defect definition. Except by [13], the taxonomies use an informal or example-driven description model which require considerable interpretations when considered from a technical perspective.

Regarding *terminology* and *nomenclature*, the taxonomies use distinct terms to well known database jargon defects. For instance, "Domain format errors" by [10] and "Wrong data type" by [7,9] are the different terms applied to the same database jargon of "Domain Constraint Violation".

The *shared absence of theoretical support* to identify the set of data defects and the lack of concern with extending the descriptions contribute to an incomplete coverage of data defects. For instance, despite the sequence of citation between [13,14] and [9], data defects as "Embedded values" by [14] do not appear in [9,13]. Moreover, defects regarding data modelling rules (e.g., Cardinality Ratio) and data life cycle failures (e.g., Missing References, False Tuple) are not addressed by any taxonomy. Further examples of data defect heterogeneities are also mentioned by [8].

3 Methodological Approach to Organize the Taxonomy

The taxonomy proposed to address the limitations and the questions in Sect. 2 resulted from applying three steps in sequence. The first step *re-examined a broad set of topics related to relational theory.* Among the topics, but no restricted to them, it can be mentioned conceptual data modelling, transformation decisions between conceptual and logical models, and constraints [5,6,11].

These topics revealed a broad rule set that may be applied on a relational schema to represent properties and behavior of the Universe of Discourse (UoD). Each rule was basis to identify one or more violations (data defects) that leads data to defective states. Furthermore, the review also determined the terminology, nomenclature and the description model.

The second step *classified the data defects in layers according to their shared properties.* The first layer determined whether or not data defects violate rules about the UoD, named respectively as "Data Constraint" and "Fact Representation Deviation". "Data Constraint" gathers data defects that violate static or dynamic rules. The former denotes explicit rules or inherent characteristics of relational model (implicit rules) that a valid state of a relation must satisfy, including data domains, integrity and participation in data relationships. In contrast, the latter comprises rules applied during state transitions of a relation. "Fact Representation Deviation" denotes defects related to differences between data representation and the corresponding fact about the objects of the UoD, including meaning, content and element of representation.

The third step *classified each data defect based on its place or granularity of occurrence*, which are attribute value (V), single attribute (A), single tuple (T), single relation (R) or interrelation (IR), which may involve one or more database instances. The outcome of this three-step methodology is observed in Fig. 1. This figure provides an effective arrangement to identify data defects and comprehend their properties (denoted by the class hierarchy) and interrelationships. Moreover, this arrangement is basis to incorporate additional data defects to the taxonomy, such as the time-related ones.

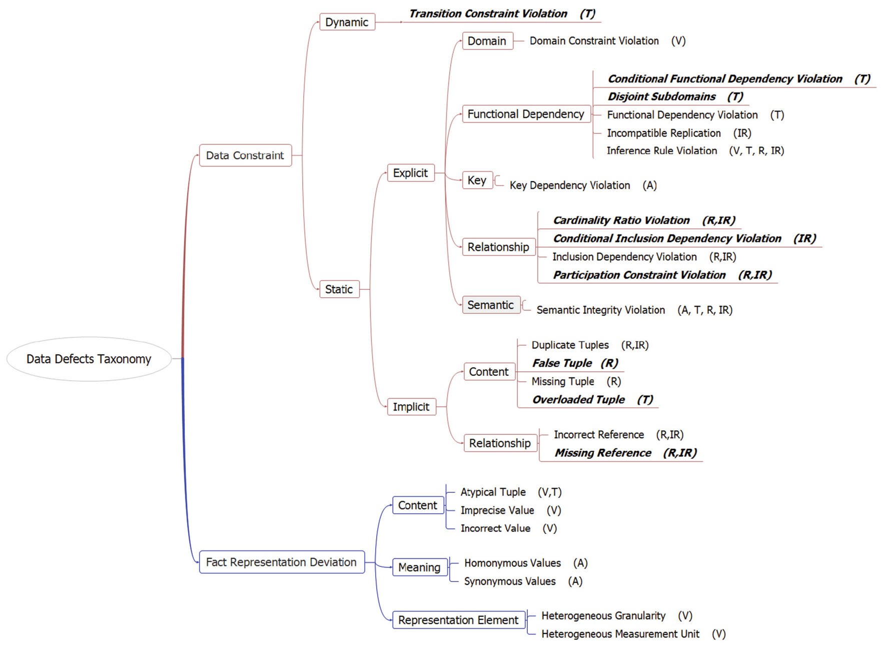
Data Defects Taxonomy
Data Constraint
Dynamic
Transition Constraint Violation (T)
Static
Explicit
Domain
Domain Constraint Violation (V)
Functional Dependency
Conditional Functional Dependency Violation (T)
Disjoint Subdomains (T)
Functional Dependency Violation (T)
Incompatible Replication (IR)
Inference Rule Violation (V, T, R, IR)
Key
Key Dependency Violation (A)
Relationship
Cardinality Ratio Violation (R,IR)
Conditional Inclusion Dependency Violation (IR)
Inclusion Dependency Violation (R,IR)
Participation Constraint Violation (R,IR)
Semantic
Semantic Integrity Violation (A, T, R, IR)
Implicit
Content
Duplicate Tuples (R,IR)
False Tuple (R)
Missing Tuple (R)
Overloaded Tuple (T)
Relationship
Incorrect Reference (R,IR)
Missing Reference (R,IR)
Fact Representation Deviation
Content
Atypical Tuple (V,T)
Imprecise Value (V)
Incorrect Value (V)
Meaning
Homonymous Values (A)
Synonymous Values (A)
Representation Element
Heterogeneous Granularity (V)
Heterogeneous Measurement Unit (V)

Fig. 1. Taxonomy of data defects (*Granularity:* V-Attribute Value, A-Single Attribute, T-Single Tuple, R-Single Relation, IR-Interrelations. *Note:* Data Defects in *italic*, **bold** and <u>underline</u> were not addressed by any of the state-of-art taxonomies.)

4 Data Defect Taxonomy

4.1 Structural Background

This work applies a formalization language well known by database community based on [5,6,11], which the main elements are observed in Table 1. It is beyond the scope of this work to evaluate the most proper language for this goal.

Moreover, each defect is illustrated by examples selected from a simple financial domain, as shown in the logical model below.

Customer (<u>CID</u>, Name, Job, Salary, State, City, Zip, Age, Ms, SpouID)
SpouID references Customer
CreditCardAccount (<u>CAccID</u>, ActivationDate, UsageTime, IsFreeOffer)
CustomerCreditAccount (<u>CCAID</u>, CAccID, CID, Score, IsHolder, State)
CAccID references CreditCardAccount
CID references Customer

In this model, *Customer* has certain properties of Natural People and Legal Entities in regard to owners of credit cards. *CreditCardAccount* denotes the properties of acquired credit cards. *CustomerCreditAccount* represents all of the relationship roles (holder or joint holder) between the customers and the credit cards. An instance I_0 of each logical relation is observed in Tables 2, 3 and 4.

Table 1. Main elements of the formalization language

Relational	Relational Database Schema	Set of relations schemas $BD = \{R_1, R_2, R_3..., R_m\}$, $m \geq 1$
	Relation Schema	Set of attributes $A = \{a_1,..., a_k\}$ denoted by $R(A)$, $R \in DB$
	Attribute	Each a_j, $j = [1, k]$, is regulated by a domain D_j, given as $dom(a_j)$
	Subset of a Relation	Set of attributes $X, Y \subset R(A)$, where $R \in DB$ and $X \cap Y = \oslash$
	State of Relation	Set of n tuples, $r = \{t_1, t_2, t_3..., t_n\}$, denoted by $r(R)$
	Tuple	Each tuple t_p, $p \in [1, n]$, is a list of q values $t_p = \{v_1, v_2, ..., v_q\}$
	Tuple Value	Each value v_s, $s \in [1, q]$, is a domain element of the corresponding attribute a_s, denoted as $t[a_s]$
	Relationship	Referential integrity rule between relations W (refer to) and U (referred), denoted by $Rel : R_W \rightarrow R_U$
	Universal Thesaurus	Lexical definitions, relationships and similarity degrees of terms in common usage, denote by LEX.
Operational	Value Predicate Symbols	$\ominus = \{<, \leq, =, \neq, \geq, >\}$
	Set Elements and Operators	$Q = \{\in, \notin, \subseteq, \subset, \cup, \cap\}$
	Logical Connectives	$\{\wedge, \vee\}$ of type $Boolean \times Boolean \rightarrow Boolean$
	Unary Connective	$\{\neg\}$ of type $Boolean \rightarrow Boolean$
	Quantifiers	$\{\forall, \exists\}$ are the universal and existential quantifiers

Table 2. An instance of customer

	CID	Name	Job	Salary	State	City	Zip	Age	Ms	SpouID
c1:	1	John Taylor	Bassist	20k	SP	SP	08000	52	E	19
c2:	3	Joan Ripley	Tapster	320k	BHZ	BHZ	03000	20	M	40
c3:	8	John T	Bartender	20k	MG	BHZ	08200	52	W	NULL
c4:	13	Ann P. Taylor	Barkeeper	NULL	MG	BHZ	31000	44	U	1
c5:	19	Chris Taylor	NULL	8k	SP	SP	08100	39	J	28
c6:	28	Carl de la Poll	Student	21k	SP	SP	08400	34	M	13
c7:	29	James Bond	Bassist	22k	SP	SJC	08000	53	W	NULL
c8:	40	Alice Bond	Principle Manager	1k	SP	SP	08501	53	E	49
c9:	41	John N. T	Principal Manager	40k	MG	BHZ	03099	17	Y	NULL
c10:	3	Ann P. Taylor	Writer	38k	MG	BHZ	03100	44	J	1
c11:	52	Jean P. Jones	Student	33k	SP	SJC	08400	15	S	NULL
c12:	53	Dick Rhodes	Writer	35k	SP	SJC	12200	45	W	NULL

Table 3. An instance of CreditCardAccount

	CAccID	ActivationDate	UsageTime	IsFreeOffer
cr1:	100	07/30/2001	13	No
cr2:	155	01/19/2004	10	No
cr3:	199	05/12/2005	9	Yes
cr4:	200	01/19/2004	1	No
cr5:	201	04/11/2013	1	No

Table 4. An instance of CustomerCreditAccount

	CCAID	CAccID	CID	Score	IsHolder	State
cc1:	120	100	1	2.12307	Yes	SP
cc2:	312	100	13	3.00999	No	MG
cc3:	138	100	19	1.80500	No	SP
cc4:	813	100	3	3.10999	Yes	MG
cc5:	883	155	28	2.11001	Yes	SP
cc6:	901	199	44	3.89099	Yes	SP
cc7:	902	200	40	2.12320	Yes	MG
cc8:	903	201	52	1.83449	Yes	MG
cc9:	909	201	41	1.80011	No	MG
cc10:	911	100	3	19.13329	No	SP

4.2 Taxonomy Description

This section formally describes the higher granularity of each data defect, as observed in Fig. 1. For each definition, there is an example based on the financial logical model aforementioned. The taxonomy has a subset of defect that requires deep knowledge of data context to determine its occurrence, i.e., requires high human curation. Therefore, this subset's formalization applies elements that denote a specialist knowledge.

Definition 1 (Atypical Tuple). Let $outl : R(A) \rightarrow \{true, false\}$ be a function that maps an attribute from relation R to a statistical result of outlier detection methods. An atypical value occurs *iff* $\exists t \in R, \exists a \in R(A)$ such that $outl(t[a])$ is *true*.

An atypical tuple deviates from the common behavior of most tuples within a relation. The unusual composition of attributes values or the unusual value of an isolated attribute are instances of atypical tuple.

Example: A Customer tuple $c2$ reveals an uncommon situation due to the composition of its Salary, Job and Age values.

Definition 2 (Cardinality Ratio Violation). Let $cr : R_1 \times R_2 \rightarrow \mathbb{N}$ be a function that maps the cardinality association of $Rel : R_1 \rightarrow R_2$. Let $s : r(R_1) \times r(R_2) \rightarrow \mathbb{N}$ be a function defined as follows:

$$s(a,b) = \begin{cases} 1 & \text{if } a = b, \\ 0 & \text{if } a \neq b. \end{cases}$$

The cardinality violation occurs *iff* $\exists t_i \in R_1,\ t_j \in R_2$ such that $\sum s(t_i[W], t_j[U]) > cr(R_1, R_2)$.

The cardinality ratio establishes the maximum number of relationship instances so that each tuple from a relation can participate within a binary relationship. A violation occurs when any tuple does not comply with the maximum as the *referred* or *refer to* role.

Example: In regard to the self-relationship of the "marriage" role (SpouID), each customer must be referred to at most once. However, the customer on tuple $c1$ is referred twice (tuples $c4, c10$).

Definition 3 (Conditional Functional Dependency Violation). Let Tp be a pattern tableau with attributes in X_1 and Y_1, where for each attribute $a \in X_1 \cup Y_1$ and for each pattern tuple $tp \in Tp$, $tp[a]$ is either a particular value in dom(a) or a "wildcard" symbol "_" that draws values from dom(a). Let Y_1 be conditionally dependent on X_1 defined as $(X_1 \rightarrow Y_1, Tp)$ on relation R. This conditional dependency is violated *iff* $\exists t_i, t_j \in r(R)$, $i \neq j$, such that $t_i[X_1] = t_j[X_1] = tp[X_1]$ and $t_i[Y_1] = t_j[Y_1] \neq tp[Y_1]$ or $t_i[Y_1] \neq t_j[Y_1]$.

A conditional functional dependency $(A \rightarrow B, Tp)$ on a relation R denotes that B values depend functionally on A values only for the tuples of R that satisfies a data pattern specified at Tp. A violation arises when this constraint is not obeyed by a some B value.

Example: The Customer relation has a conditional functional dependency between State, City and Zipcode, denoted by $[State, City \rightarrow ZipCode, Tp]$. Pattern Tableau Tp specifies that state "SP" and city "SP" uses the zipcode range between 08000 and 08499, while the same state and city "SJC" has a zipcode range from 12200 to 12248. However, the tuples $c7$ and $c11$ violate these patterns.

Definition 4 (Conditional Inclusion Dependency Violation). Let Tp be a pattern tableau with attributes in X_p and Y_p, where for each attribute $a \in X_p \cup Y_p$ and for each pattern tuple $tp \in Tp$, $tp[a]$ is a particular value in dom(a). Let R_1 be conditionally dependent on R_2, as represented by $(R_1[X; X_p] \subseteq R_2[Y; Y_p], Tp)$. This dependency is violated *iff* $\exists t_i \in r(R_1), \exists t_j \in r(R_2)$ such that $t_i[X_p] = tp[X_p]$ and $t_j[Y_p] = tp[Y_p]$ and $t_i[X] \neq t_j[Y]$.

The OR exclusive constraint specifies that relationships set to a root relation must be disjunct. A violation of this constraint arises when there are root relation tuples that participate in two or more mutually exclusive relationships.

Example: An disjoint constraint prohibits that holder and joint holder roles be exerted by the same customer. However, there are a customer ($c2$) with both roles on the same card ($cc4$ and $cc10$).

Definition 5 (Disjoint Subdomains). The problem of disjoint subdomains holds when exists subdomains $S_1, S_2, ..., S_n$ for an attribute a_j such that $dom(a_j) = \bigcup_{i=1}^{n} S_i$, and exists a function $f : a_i \rightarrow \{S_1, S_2, \ldots, S_n\}$ that maps a_i to one subdomain of a_j, $i \neq j$ and $a_i, a_j \in R(A)$. It establishes that values of a_j depend on values of a_i.

An attribute has disjoint subdomains (or multiple uses) when its values represent different facts about the objects of the UoD, according to some assignment predicate.

Example: The Salary attribute may represent an adjusted remuneration for providing services (e.g., tuple $c1$) or an estimated family income (e.g., tuples $c6, c11$). The latter occurs when the customer does not have an income source, i.e., Job equal to "Student" or Age lower than sixteen years old.

Definition 6 (Domain Constraint Violation). The domain violation occurs *iff* $\exists t \in R, \exists a \in R(A)$ such that $t[a] \notin dom(a)$.

A domain constraint regulates the allowed values for an attribute domain. In this work, a domain constraint denotes a set of values (such as enumerations), a interval or semi-interval constraint, a mandatory (Not-Null) constraint or a regular expression. A domain constraint violation arises when a value does not match the permissible values of the attribute.

Example 1: The Ms attribute (abbreviation of "Marital Status") of the customer relation has a domain defined as: "M", "E", "J", "U", "D", "W", or "S". However, the tuple $c9$ contains "Y" for this attribute.

Example 2: The Score attribute of the CustomerCreditAccount relation has a domain constraint between −2.99999 and 9.9999. However, the tuple $cc10$ has a score of 19.13329.

Definition 7 (Duplicate Tuples). Let X_1 and X_2 be attribute subsets, where $X_1 \subset R_1$ and $X_2 \subset R_2$. Let X_1 and X_2 be pairwise compatible, where for all $a_1^i \in X_1$ and $a_2^i \in X_2$, $i \in [1, k]$, $k \geq 1$, $dom(a_1^i)$ and $dom(a_2^i)$ are identical. Let $\simeq_i$ be the record matching similarity predicate on attributes $a_1^1 \simeq_1 a_2^1 \wedge ... \wedge a_1^k \simeq_k a_2^k$, denote as $X_1 \simeq X_2$. There are duplicate tuples *iff* $\exists t_1 \in r(R_1), \exists t_2 \in r(R_2)$ such that $t_1[X_1] \simeq t_2[X_2]$.

This defect denotes multiple tuples from one or more relations that refer to the same object in the UoD. The content of these tuples may have identical values, have a certain similarity degree or be mostly divergent.

Example: The Customer tuples $c4$ and $c10$ represent the same object in the UoD with equal values in almost all the attributes.

Definition 8 (False Tuple). Let $ftup : r(R) \rightarrow \{true, false\}$ be a function which returns if a tuple from R complies with the rules that define its usefulness for the UoD. A tuple is false *iff* $\exists t \in r(R)$ such that $ftup(t)$ is *false*.

A database schema must represent only the required objects of an UoD. A tuple is named false when it represents an object *beyond* the UoD interest.

Example: The tuple $c7$ represent a customer who have never had a credit card.

Definition 9 (Functional Dependency Violation). Let Y be functionally dependent on X, denoted by $X \rightarrow Y$, $X, Y \subseteq R(A)$. This dependency is violated *iff* $\exists t_i, t_j \in R$, $i \neq j$, such that $t_i[X] = t_j[X]$ and $t_i[Y] \neq t_j[Y]$.

A functional dependency $A \rightarrow B$ denotes that each B values is associated with precisely one A value. A violation arises when this constraint is not obeyed by some B value.

Example: The Customer relation has the functional dependency $State \rightarrow City$. However, this dependency is violated by the tuple $c2$.

Definition 10 (Heterogeneous Granularity). Let G be the granularity defined for the attribute a from relation R, $a \in R(A)$. Let $grain : a \rightarrow \{true, false\}$ be a function which returns if an value of attribute a complies with the granularity G. This defect occurs *iff* $\exists t_i \in r(R)$ such that $grain(t_i[a])$ is *false.*

Granularity denotes the abstraction level of value representation. There is a heterogeneous granularity attribute when some of its values represent facts about objects of the UoD using different abstraction levels. These abstraction levels may have distinct degrees of disparity and also expose a random or attribute-driven pattern.

Example: Salary attribute of Customer relation must represent monthly pay. However, the tuple $c2$ has the annual payment.

Definition 11 (Heterogeneous Measurement Unit). Let Ω be an equivalence relation on the values of attribute a, $a \in R(A)$, such that $v_i \Omega v_j$ iff v_i and v_j have the same measurement unity. Let MU_Ω be an equivalence class on Ω which contains all the values of attribute a which has the measurement unity required by the UoD. Heterogeneous measurement unit holds *iff* $\exists t_i \in r(R)$ such that $t_i[a] \notin MU_\Omega$.

A measurement unit denotes the magnitude of a given physical quantity, which provides useful basis for comparison. This defect occurs when certain values of an attribute represent facts about objects of the UoD using different measurement units. These units denote dissimilar magnitudes and also expose a random or attribute-driven pattern.

Example: Customer relation must represent salaries in American dollars. However, the customers whose job is "Writer" (tuples $c10, c12$) have their salaries represented in Euros.

Definition 12 (Homonymous Values). Let $sp : r(R) \times r(R) \rightarrow \{true, false\}$ be a function which returns if the graphy and pronunciation of attributes values are equal, according to LEX. Let $me : r(R) \times r(R) \rightarrow \{true, false\}$ be a function which returns if the meaning of attributes values are equal or nearly the same, according to LEX. An attribute has homonymous values *iff* $\exists a \in R(A), \exists t_i, t_j \in r(R), i \neq j$, such that $sp(t_i[a], t_j[a])$ is *true* and $me(t_i[a], t_j[a])$ is *false.*

The homonym denotes terms pronounced in the same way but with distinct meanings. This defect arises when homonymous terms are applied interchangeably and indicate the same fact about objects of the UoD.

Example: The customer tuples $c8$ and $c9$ has different jobs ("Main Manager" and "Principal", respectively) represented with homonymous terms.

Definition 13 (Imprecise Value). Let σ be the imprecise degree allowed to each value v of the attribute a, $a \in R(A)$, such that $v - \sigma \leqslant v \leqslant v + \sigma$. Let $rv : a \rightarrow dom(a)$ be a function on relation R which returns the value of attribute a in the UoD. Imprecise value occurs *iff* $\exists t \in r(R)$ such that $|t[a] - rv(a)| > \sigma$.

The term “imprecise” denotes a value that is close to the fact about of an object of the UoD. This closeness denotes an certain accuracy level determined by a range of values within which the accurate value is asserted to be.

Example: The Score attribute of CustomerCreditAccount allows a imprecision degree between ± 0.00049. The tuple $cc3$ has the score of 1.80500 for an accurate value of 1.80538.

Definition 14 (Inclusion Dependency Violation). Let $Rel : R_1 \rightarrow R_2$ be a relationship set between relations R_1, R_2. There is an inclusion dependency violation *iff* $\exists t_i \in r(R_1)$, $\forall t_j \in r(R_2)$ such that $t_i[W] \neq t_j[U]$.

Inclusion dependency imposes acceptance conditions on actions over instances of relationships to ensure referential integrity consistency. A violation is created when a tuple t_1 refers to tuple t_2, which is not available for the referred relation.

Example: The CustomerCreditAccount tuple $cc6$ refers to a customer who is absent within the Customer relation.

Definition 15 (Incompatible Replication). Let $X \subset R_1(A)$, $Y \subset R_2(A)$ be two subsets of relations R_1, R_2. A replication $X \rightrightarrows Y$ occurs if $\forall t_i \in R_1$, $t_j \in R_2$, $t_i[W] = t_j[U] \Rightarrow t_i[X] = t_j[Y]$. A replication defect occurs *iff* $\exists t_i \in r(R_1)$, $t_j \in r(R_2)$, such that $t_i[W] = t_j[U]$ and $t_i[X] \neq t_j[Y]$.

Due to certain reasons (including performance and poor data modelling), a base attribute may have its content replicated into multiple attribute copies. There is a contradictory situation when these attributes have different values.

Example: The State attribute of the Customer relation should have been replicated to the State attribute of the CustomerCreditAccount relation. However, the two hold different values for the same client (tuples $cc4, cc7, cc8$).

Definition 16 (Incorrect Reference). Let $Rel : R_1 \rightarrow R_2$ be a relationship set between relations R_1, R_2. Let $rrel : Rel \rightarrow \{true, false\}$ be a function which returns if an instance of relationship Rel holds in the UoD. Incorrect reference occurs *iff* $\exists (t_i, t_j) \in Rel$ such that $rrel(t_i, t_j)$ is *false*.

This occasion refers to a relationship instance that does not represent a fact about an object of the UoD, although it obeys all of the other rules.

Example: Customer tuple $c8$ is owner of credit card number 199 (tuple $cc6$). However, this customer is related to the credit card number 200 (tuple $cc7$).

Definition 17 (Incorrect Value). Let $rv : a \rightarrow dom(a)$ be a function on relation R which returns the value of attribute a in the UoD. Incorrect values occurs *iff* $\exists t \in r(R)$ such that $t[a] \neq rv(a)$.

An incorrect value is an unfaithful or contradictory representation of a fact about an object of the UoD. In other words, such a defect denotes a large discrepancy between the represented value and the real value of the object.

Example: A customer's salary is 73.4k, but it was represented as 8k on tuple $c5$.

Definition 18 (Inference Rule Violation). Let $Rel : R_1 \rightarrow R_2$ be a relationship set between relations R_1, R_2. Let $ir : Rel \rightarrow \{true, false\}$ be a function which returns if an instance of relationship Rel complies with its inference rule. There is an inference rule violation *iff* $\exists(t_i, t_j) \in Rel$ such that $ir(t_i, t_j)$ is *false.*

An inference rule is a procedure that generates new facts based on the ones available in a database. A violation arises when an inferred attribute value, tuple or relationship instance is not represented, or when it is different from the one that was determined by the rule.

Example: The Score attribute of the CustomerCreditAccount relation is inferred by a complex analysis of credit card usage for the last six months. This relation has a tuple ($cc8$) where Score is 1.83449 instead of 1.01553, which was inferred.

Definition 19 (Key Dependency Violation). Let X be an attribute subset of relation R, $X \subseteq R(A)$. Let $R(A)$ be key dependent on X, as represented by $X \rightarrow R(A)$. This dependency is violated *iff* $\exists t_i, t_j \in r(R)$, $i \neq j$, such that $t_i[X] = t_j[X]$.

The purpose of the identifier attribute subset is to uniquely identify all relation tuples to enable data relationship. This situation is violated when two or more tuples share the same value for their identifiers' attributes.

Example: The tuples $c2$ and $c10$ share their CID, but they are distinct customers.

Definition 20 (Missing Reference). Let $Rel : R_1 \rightarrow R_2$ be a relationship set between relations R_1, R_2. Let $rrel : Rel \rightarrow \{true, false\}$ be a function which returns if an instance of relationship Rel holds in the UoD. A reference is absent *iff* $\exists t_i \in r(R_1), \exists t_j \in r(R_2)$ such that $(t_i, t_j) \notin Rel$ and $rrel(t_i, t_j)$ is *true.*

Relationship instances represent facts about objects of the UoD. The missing reference defect arises when a required relationship instance is not represented.

Example: The Customers tuples $c11$ and $c12$ are married. However, this marriage relationship has not been represented.

Definition 21 (Missing Tuple). Let $mist : DB \rightarrow \{true, false\}$ be a function which returns if a relation from DB represents all of the required objects of the UoD. A tuple is absent *iff* $\exists R_i \in DB$ such that $mist(R_i)$ is *false.*

A database schema must represent only the required objects of the UoD and their properties. The missing tuple defect denotes the lack of representation of certain important objects of the UoD.

Example: Certain joint holders of credit card accounts (tuples $cc5$ to $cc8$) are not represented in the Customer relation.

Definition 22 (Overloaded Tuple). Let $overt : r(R) \rightarrow \{true, false\}$ be a function which returns if a tuple from R represents a single object of the UoD. A tuple is overloaded *iff* $\exists t_i \in r(R)$ such that $overt(t_i)$ is *false.*

A single tuple represents facts about a single object of an UoD. An overload denotes an excessive representation (more than one) of objects by one tuple.

Example: "Dick Rhodes" and "Dick Rhodes" are two distinct people of the UoD. Nonetheless, only a single customer tuple $c12$ represents both.

Definition 23 (Participation Constraint Violation). Let $pc : R_1 \times R_2 \rightarrow \mathbb{N}$ be a function that maps the minimal participation of tuples from R_1 to R_2. Let $s : r(R_1) \times r(R_2) \rightarrow \mathbb{N}$ be a function defined as follows:

$$s(a,b) = \begin{cases} 1 & \text{if } a = b, \\ 0 & \text{if } a \neq b. \end{cases}$$

A minimum participation violation occurs *iff* $\exists t_i \in R_1$, $t_j \in R_2$ such that $\sum s(t_i[W], t_j[U]) < pc(R_1, R_2)$.

The participation constraint determines the minimum number of relationship instances in which each tuple from a relation must participate in a binary relationship. The violation occurs when a tuple does not comply with the minimum *referred* or *refer to* role.

Example: A credit card must be associated with at least two customers. However, there are credit cards in the CustomerCreditAccount relation with only one customer (tuples $cc5$ to $cc7$).

Definition 24 (Semantic Integrity Violation). Let $Rel : R_1 \rightarrow R_2$ be a relationship set between relations R_1 and R_2. Let $rule : Rel \rightarrow \{true, false\}$ be a function which returns if an instance of relationship Rel complies with its semantic integrity rule. Let RU^{Rel} be the set of semantic rules on relationship Rel, denoted as $RU^{Rel} = \{rule_1, ..., rule_z\}$, $z \geqslant 1$. There is a semantic integrity violation *iff* $\exists (t_i, t_j) \in Rel, \exists rule_h \in RU^{REL}$ such that $rule_h(t_i, t_j)$ is *false*.

The semantic integrity comprises a set of complex rules for an UoD that guarantees a state of data consistency. A violation arises when one of these rules is disobeyed.

Example: A Customer relation rule determines that only 9–17 years old people living at "SP" state can possess a credit card as joint holder. For the remaining states this relationship is forbidden. However, such a rule is disobeyed by a certain customer (tuple $c9$) of "MG" state that has a credit card (tuple $cc9$).

Definition 25 (Synonymous Values). Let $sp : r(R) \times r(R) \rightarrow \{true, false\}$ be a function which returns if the graphy and pronunciation of attributes values are equal, according to LEX. Let $me : r(R) \times r(R) \rightarrow \{true, false\}$ be a function which returns if the meaning of attributes values are equal or nearly the same, according to LEX. An attribute has synonymous values *iff* $\exists a \in R(A), \exists t_i, t_j \in r(R), i \neq j$, such that $sp(t_i[a], t_j[a])$ is *false* and $me(t_i[a], t_j[a])$ is *true*.

Synonyms denote distinct terms in writing that share the same or similar meanings. Such terms can be expressed as vernacular words, acronyms, abbreviations or symbols. This defect arises when synonymous terms are used interchangeably to indicate the same fact about objects of the UoD.

Example 1: Customer relation tuples $c1, c4, c5, c8, c10$ have marital statuses such as "E" (espoused), "J" (joined), "U" (united) that designate married ("M") in each case.

Example 2: Customer relation tuples $c2$ and $c4$ have job titles as "Tapster" and "Barkeeper" that designate "Bartender" in each case.

Definition 26 (Transition Constraint Violation). Let $tran : R \rightarrow R$ be a transitional function that leads the original state of R to another state R', according to a inference system $R \rightarrow^{tran} R'$. A transition violation occurs *iff* $\exists t \in R$ such that $t[R'(A)] \neq t[tran(R(A))]$.

The transition or dynamic constraints represent a set of rules that enforces the valid state transitions of data. These constraints are evaluated on a pair of successive pre and post-transaction states of a database relation. A violation arises when a tuple possesses an invalid post-transaction state.

Example: There is a rule that regulates the transitions between the valid states of MS ("MaritalStatus") attribute. The tuple $c12$ violates this rule because his marital status has changed from "S" (Single) to "W" (Widower).

5 Conclusions

This work reports a taxonomy that organized a detailed description of data defects regarding to the quality dimensions of accuracy, completeness and consistency. The taxonomy applied a three-step methodology to address all the issues discussed in Sect. 2: the theoretical review enabled the systematic and broad coverage of data defects (nine defects were not addressed by the state-of-art taxonomies, as highlighted in Fig. 1), and improved the data defect descriptions; the classification steps organized data defects according to their properties and granularity of occurrence.

The taxonomy structure can support relevant issues in data quality assessment, including the training of data quality appraisers, the establishment of measurement-based process and guiding the design decisions in regard to semi-supervised approaches for detecting data defects. Nevertheless, the taxonomy does not address time-related data defects, as well as it offers high level formal descriptions of some defects since they involve human curation or complex and broad rules. In future works, this taxonomy will be the basis for classifying data defects according to time-related data and designing a supervised approach for data defect detection.

Acknowledgments. This work has been supported by CNPq (Brazilian National Research Council) grant number 141647/2011-6 and FAPESP (Sao Paulo State Research Foundation) grant number 2015/01587-0.

References

1. Almutiry, O., Wills, G., Crowder, R.: A dimension-oriented taxonomy of data quality problems in electronic health records. In: 13th IADIS International Conference on e-Society, pp. 98–114. IADIS, Portugal (2015)
2. Barateiro, J., Galhardas, H.: A survey of data quality tools. Datenbank-Spektrum **14**, 15–21 (2005)

3. Borek, A., Woodall, P., Oberhofer, M., Parlikad, A.K.: A classification of data quality assessment methods. In: 16th International Conference on Information Quality, pp. 189–203. IEEE Press, New York (2011)
4. English, L.P.: Improving Data Warehouse and Business Information Quality: Methods for Reducing Costs and Increasing Profits. Wiley, New York (1999)
5. Fan, W., Geerts, F.: Foundations of Data Quality Management. Morgan & Claypool Publishers, San Rafael (2012)
6. Grefen, P.: Combining theory and practice in integrity control: a declarative approach to the specification of a transaction modification subsystem. In: 19th International Conference on Very Large Data Bases, pp. 581–591. Morgan Kaufmann Publishers Inc., Dublin, Ireland (1993)
7. Kim, W., Choi, B.-J., Hong, E.-K., Kim, S.-K., Lee, D.: A taxonomy of dirty data. Data Min. Knowl. Discov. **7**, 81–99 (2003)
8. Laranjeiro, N., Soydemir, S.N., Bernardino, J.: A survey on data quality: classifying poor data. In: 21st Pacific Rim International Symposium on Dependable Computing, pp. 179–188. IEEE Press, Zhangjiajie, China (2015)
9. Li, L., Peng, T., Kennedy, J.: A rule based taxonomy of dirty data. GSTF Int. J. Comput. **1**, 140–148 (2011)
10. Müller, H., Freytag, J.C.: Problems, methods, and challenges in comprehensive data cleansing. Technical report, Humboldt University Berlin (2005)
11. Maier, D.: The Theory of Relational Databases. Computer Science Press, Rockville (1983)
12. Naumann, F.: Data profiling revisited. ACM SIGMOD Rec. **42**, 40–49 (2014)
13. Oliveira, P., Rodrigues, F., Henriques, P.: A formal definition of data quality problems. In: International Conference on Information Quality, pp. 181–184. IEEE Press, New York (2005)
14. Rahm, E., Do, H.H.: Data cleaning: problems and current approaches. IEEE Bull. Tech. Comm. Data Eng. **23**, 3–13 (2000)
15. Schmid, J.: The main steps to data quality. In: Perner, P. (ed.) ICDM 2004. LNCS (LNAI), vol. 3275, pp. 69–77. Springer, Heidelberg (2004)
16. Winkler, W.E.: Methods for evaluating and creating data quality. Inf. Syst. **29**, 531–550 (2004)