Evaluation and Improvement of Database Schemas: A transformation-based framework

Jonathan Lemaitre

Laboratory of Database Application Engineering - PReCISE research Center
Faculty of Computer Science, University of Namur
Rue Grandgagnage 21 - B-5000 Namur, Belgium
jle@info.fundp.ac.be
http://www.fundp.ac.be/precise

Abstract. Data schemas are primary artefacts for the development and maintenance of data intensive software systems. As for the application code, one way to improve the quality of the models is to ensure that they comply with best design practices. In this paper, we present a research on data schemas quality, in which the quality evaluation process is based on the identification of specific schema constructs and their comparison with best practices. We provide an overview of a framework based on the use of semantics-preserving transformations as a way to compare and suggest improvements for the most significant best design practices. The paper also summarize the contribution of the approach and of the current achievements.

Keywords: Database schema, schema evaluation, schema improvement, schema transformation.

1 Introduction

Quality has become one of the main topics in software engineering. Research and industrial communities acknowledge that behind maintainability, efficiency, etc., lies the satisfaction of the users and the financial impacts. The question has been studied through many works during the last three decades. In the 90's authors have already assessed the impact of poor quality and errors made during the modeling phase [1]. During the last few years quality of schemas became more and more important owing to the MDE approach that relies mainly on modeling and schema transformations.

Looking at data schema, one can observe different approaches to deal with quality. A first approach regroups the quality model and framework. Quality models are mainly composed of definitions of quality characteristics [2, 3] while in addition frameworks define high level views of the quality and methodologies [4–6]. Using quality models and frameworks, some authors propose metrics [7, 8]. Metrics give a numerical evaluation of (a specific aspect of) the quality. The metric approach deals with the evaluation of quality, most of the time, in a

affordable way, but it does not allow one to precisely identify the source of a poor quality score. Finally, the last approach regroups proposals, that study very specific problems such as, for example, the normalization [9] and the impact of particular construct [10, 11]. These proposals are generally based on validated and intuitive concepts. They also offer direct means to improve the quality of models. Combined with general modeling conventions, they can define a set of commonly agreed *best practices* that are to ensure specific requirements such as expressiveness, maintainability, evolutivity, performance, etc.

Both the metric approach and the study of specific problems have their advantages and limitations. In this paper, we will describe an ongoing research in which defects and best practices are considered as a possible basis for the schema quality evaluation and improvement. Our approach relies on the identification of schema structures that have been defined to express specific types of facts of the application domain. This analysis is used for evaluating the schema by comparing its structural content to a reference frame and the requirements of the schema context. The improvement activity will modify the schema structure, using semantics-preserving transformations, in order to increase its compliance with the context while preserving its semantics. Our work address the following questions:

- What can be the contribution of semantics-preserving transformations to the quality improvement of data models?
- What kind of problems/defects could be addressed this way?
- How to provide quality evaluation methods in order to assess the impact of transformations?

In section 2, we summarize the general concepts used in our research. Those are about schema abstraction level and paradigm and schema transformation. In section 3, we present our motivations and illustrate the type of quality problems we are studying. Section 4 presents the basis of a framework for the evaluation and improvement of schema quality. In the section 5, we compare our approach to related existing proposals. Finally, the section 6 describes the results achieved so far and gives directions about future works.

2 Background

In this section we briefly describe the main basic concepts used in our work. They are the abstraction levels and paradigms and the transformational approach. The interested reader is referred to reference [12] for a more detailed description of these.

A schema is a formal description of the information/data structures of a database, be it in construction or in use. It is positioned at a certain level of abstraction. Database engineering processes generally rely on a hierarchy of 3 abstraction levels, namely the conceptual, logical and physical levels. Such multi-level approach is currently called *Model-Driven Engineering*. A schema is

also expressed in a specification language, based on a definite paradigm. Entity-relationship (ER) with its many variants, UML class diagrams, relational, object-relational, XML, IMS, standard files and even *schema-less*, are some of them. The database community calls them *models* (e.g., the *relational model*), a term we will use in this paper. There is an agreement on which abstraction level a given paradigm best fits. For instance, the Entity-relationship model is as its best at the conceptual level while the object-relational model should be used at the logical level. Abstraction levels and paradigms define a two-dimension space in which an arbitrary schema can be located and evaluated.

Any process that consists in deriving artefacts from other artefacts relies on such techniques as renaming, translating, restructuring, replacing, refining and abstracting, which basically are transformations. Most database engineering processes can be formalized as chains of elementary models and data transformations that preserve some of their aspects. This approach is known as the transformational approach. In this paper, we will address the multiplicity of representations of a given concept by the use of specific transformations called semantics-preserving. Considering a transformation as a function, say g(c), defined on a set of constructs C, g is semantics-preserving or reversible iff there exists an inverse g', defined on C' such that, for each valid instance c of C, g(c) is a valid instance of C' and c = g'(g(c)).

3 Motivations

The quality of a schema can be seen through various aspects. One can deal with the visual quality [13], that relates to the graphical representation of models and properties of the schema objects (e.g. object position). Quality also encompass the syntactical correctness. The semantic quality refers to the correctness and completeness of the schema in regard to the application domain. Quality address problems such as the unsatisfiable constructs. These are syntactically valid constructs that cannot be instantiated because of the constraints it contains [14, 15]. One can also talk about quality in term of representation choice. In the previous section, we introduced the notion of abstraction level and paradigm. They defined a set of requirements on the schema, implying that some constructs may be incongruous for representing some elements of the application domain. Those constructs cannot be called errors, as they are syntactically and semantically correct. Instead, we use the term *defect*.

The figure 1 provides 2 examples of defects. The schema *(a)* contains an is-a relationship with one empty subtype. The use of the is-a relationship is not necessary, not to say unwise, if the subtype `FORMER-CUSTOMER` is not intended to evolve. Indeed, as `FORMER-CUSTOMER` has no role, nor attribute, it is said to be weakly specific and could be replaced by a simple boolean attribute of `CUSTOMER`. By doing such a transformation, we simplify a non-minimal schema. In the schema (b), one can observe an unusual construct coming from another abstraction level and paradigm (considering (b) to be an ER-like schema). In-

deed, the entity types, together with their relationship types, form a complex but valid expression of a single many-to-many relationship type. Such construct is very common in legacy IMS databases. Considering schema *(b)* at the conceptual level, all objects in the schema, considered separately, belong to the good paradigm but their combination forms a construct influenced by another abstraction level and paradigm.

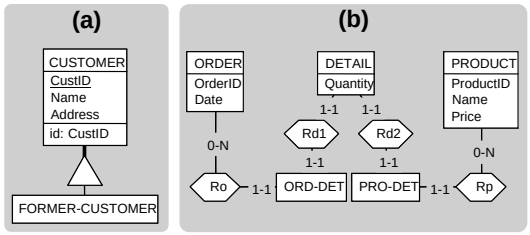

Fig. 1. Context-dependent defects

The transformational approach allow us to deal with the quality through the identification of defects, considering their possible existing alternatives and to improve schemas in order to make them compliant with best practices.

In the remaining of this section, we present defects identified for conceptual ER-like schema considering their understandability, which is one of the main requirement for conceptual schemas. The understandability refers to the efficiency with which a construct can express a type of fact of the application domain. For example, in order to represent a category of concepts A and its subcategories A_1 and A_2, the best solution will be to use an is-a relationship. It is the most expressive construct for such type of facts. However, alternatives exists and appear in schemas. A common alternative is the materialization of the is-a using one-to-one relationship types.

The identified defects have been regrouped into 5 categories. A full description can be found in [16]. Those categories are the non minimal, the insufficiently expressive, the abnormal, the irregular and the redundant constructs. For most categories, we will list the identified defects, but we can only detail some of them due to space restriction.

The **non minimal** constructs refer to the schema simplicity, which means that a type of fact should be represented as simple as possible. Attribute entity types are entity types that represent by *value* or *instance* attributes coming from another entity type. In order to simplify the schema, they can be represented in this other entity type. Empty and unique subtypes can be use to represent a simple property of the supertype, but should rather be expressed through a indicator in the supertype. Other non minimal defects are: N-ary relationship types with a [1-1] or [0-1] role; compound attributes with only one component; one-to-one relationship types with mandatory roles; weakly specified subtypes; split existence constraints.

Insufficiently expressive constructs relate to the expressivity of a schema. The expressivity requires that a construct express clearly and naturally its nature. Those defects regroup: relationship entity types; complex attributes; reference attributes; existence constraints containing roles and attributes; implicit is-a relationships. We talk about relationship entity type when a many-to-many relationship type is expressed with an entity type and 2 one-to-many relationships. Obviously, the many-to-many relationship is more expressive. A reference attribute is an implicit reference to another entity, that can be replaced by an explicit relationship.

Abnormal and **irregular** constructs decrease the foreseeable nature of a schema. Indeed, the user should not be surprise by the use of particular constructs. **Abnormal** constructs are: degenerated structures; entity types with no attribute, nor role; foreign constructs. A structure is degenerated if its composition is unjustified, e.g. a coexistence constraint with only one component or an is-a with a total constraint but only one subtype. Such defect should be solve by removing the unjustified element. Foreign constructs show a coloration coming from another paradigm and/or abstraction level (e.g. IMS,CODASYL), and may arise from a migration process. Those constructs will be replaced with the corresponding construct of the conceptual level.

Irregular constructs introduce a notion of uniformity between the constructs representing the same type of facts. In order to remove this type of defects, a choice of representation has to be done and applied to the whole schema.

Finally, the last category of defects concerns the **redundant** constructs. A schema should express a fact only once. Redundancies should be identified and removed in order to make the schema more understandable.

These defects can be removed by applying semantics-preserving transformations. Using the transformational approach allows us to apply similar reasoning with other abstraction levels, paradigms and quality characteristics. For example, at the logical level, the operational performance could be more interesting. The evolutivity of the schema may also be preferred at the conceptual level. To summary, the use of a construct C of a schema should be evaluated through three questions: Does C naturally belongs to this paradigm? Does C *feel comfortable* (so to speak) at this abstraction level? Does it best translate the intention of the designer?

4 Framework proposal

The goal we have chosen to reach in this research is to design a quality evaluation and improvement framework for data schemas. In particular, we expect (1) to augment global quality evaluation approaches with metrics based on semantically rich structural patterns and (2) to associate with each structural pattern correction transformations, in order to improve schema quality considering context requirements. Such transformations can be either suggested or automatically applied. Especially, we are evaluating the use of semantics-preserving transformations. The framework is based on the principle that these transformations

allow the production of alternative structures. Among the many possible structures, some of them, though correct, may not be considered best practices, while others may meet all requirements imposed by the context and therefore be considered as best practices. A more complete definition of the framework can be found in [12].

The concept of semantic equivalence class

In order to formalize our view of alternative structures, we defined the concept of equivalence class (EC). We consider K, the collection of all the constructs of the GER that are pertinent in some engineering processes and a set of transformations T. Let us also consider a construct C from K and all the equivalent constructs that can be derived through the reversible transformations of T. All these constructs, together with C, form an equivalence class called $ec(C)$. Since only reversible transformations have been applied, $\forall C' \in ec(C), ec(C') = ec(C)$. We now define the function $sec : K \rightarrow (K \times 2^K)$. $sec(C)$ associates to each construct in K its semantic equivalence class (sec), an equivalence class in which the specific element C has been tagged. C is the *intention* of this equivalence class. $sec(C)$ provides all the constructs a designer can introduce in a schema to express the semantics (the application domain fact type) of C, hence the name *semantic equivalence class* or *sec*.

Among a SEC, we consider a *best* structure, that is, the most suitable structure for expressing the modeling intention. Such structure can generally be considered as the best practice of the SEC in term of expressiveness. However, as discussed previously, depending on the context the use of this structure is not the best solution.

Context and SEC

In order to understand the use of the SEC, we need to define in more detail the concept of *context*. The *context* of a schema S is a set of requirements defined by the intended use of S. S has been designed for the abstraction level A, according to the paradigm P and to meet the design criterion D. We call *(A,P,D)* the context of S. Given a construct C that can appear in schema S, a scoring function is assigned to $sec(C)$ for a given context. As the SEC are defined independently of any model, we propose the concept of *projection* in order to take into account the model used in a particular context. The *projection* of a SEC for a model M provides the subset of all constructs of the SEC that comply with M.

Generation and representation of the structures

The application of transformations for generating the SEC should be considered carefully. The equivalence class of a construct C can be obtained by recursively applying the transformations of T until no new construct can be produced.

However, this naive approach can lead to a very large (and, depending on T, possibly infinite) set of constructs of which only a small subset would be of interest. Appropriate meta-rules are necessary to keep the process into reasonable limits. Considering the *is-a* pattern, one can adopt a *regularity of treatment* meta-rule according to which each sub-category of a given category must be expressed in the same way. For example, a construct obtained by applying the upward inheritance transformation to one sub-category and the materialization transformation to another one would be rejected. Another example: when an entity type EA results from the transformation of an attribute A, the attribute(s) of the latter cannot be further transformed through the same transformation (figure 2).

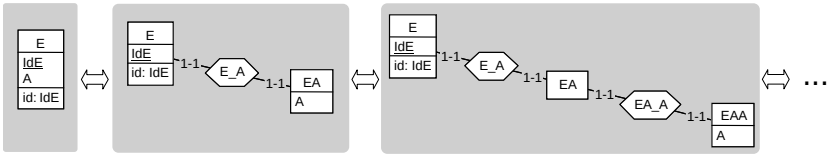

Fig. 2. Infinite transformation of an attribute.

Another important aspect of our framework is the use of generic definitions, or patterns, for representing the studied constructs. While the use of concrete examples, such as in the figure 1, for illustrating the considered defects is natural, it is completely unrealistic to attempt to list all of them without introducing a certain level of genericity.

5 Related works

The normalization process proposed by Codd [9] relies on the use of transformations in order to eliminate problematic functional dependencies. Compared with our framework, it deals with a *no redundancies* quality criterion. Burton and Weber [11] and Gemino and Wand [10] have studied particular constructs and their influence on specific qualities. Even though they did not explicitly refer to reversible transformations, our work seek to address similar problems. An important basis of our work is the proposal of Assenova and Johanesson [17]. They have considered the use of reversible transformations for enhancing the quality of conceptual data schemas. However, they associated quality *scores* directly to transformations, while we consider it to be related to the construct itself. Finally, Kurtev [18] used the concept of *transformation space* for dealing with schema quality. Such space represents a transformation by its initial and resulting structures and allows to link it with quality indicators. However, studied objects are atomic, while we consider semantically richer constructs.

6 Achievements and future works

So far, we have identified about 20 SEC, through a schema review process. All SEC represent some of the most common modeling intentions, that can be regrouped into 4 main categories: concepts, properties, relationships and constraints. Those categories are generally related to specific types of objects, e.g. entity types and tables for the concepts and attributes and columns for properties. The SEC regroups different constructs richer, in term of semantics, than simple objects. Besides the SEC, specific types of defects have also been identified and classified (e.g. non-minimal constructs, unexpressive constructs). The relation between SEC and those specific defects is currently studied.

Using SEC for enhancing the quality of schemas requires the ability to compare the quality between constructs. We have considered different approaches: the standard empirical studies, the use of metrics and the evaluation by experts. Among them, we choose to evaluate the quality of structures through the opinions of experts, which seemed to us to be a good compromise between the cost and the validity of the evaluation. Experts will be asked to assess each construct, independently of the schemas in which it appears.

The next step of this research is the complete quality evaluation of the SEC and their constructs. We should gather experts in order to obtain their quality rating. Such rating will allow us to produce *construct-based* metrics and will be the basis of the improvement process. The improvement method has to be defined in detail and carefully considering limits of our approach. For example, constructs belonging to different SEC could have common objects in a schema. Consequently, conflicts may appear between possible improvements.

An important goal of our research is to ensure the usability of the framework. This cannot be realized without a tool support for the identification of the constructs in a schema and the application of an improvement process. However, such tool has to be semi-automatic, due to limits of the approach. As the definitions of the SEC constructs is generic and represent structural properties, the same patterns may appear in different SEC. In such case, the identification of the modeling intention between the different possibilities has to be done manually by the analyst.

It remains to check the validity of the framework. Here, we wish to rely on teachers and students. (Last year) students form a realistic sample of designers of various skills, ranging from desperately inapt to experienced and ingenious. On the other hand, teachers are expected to be expert in evaluating the quality of medium size schemas. Therefore, comparing and aligning academic and automated evaluations allow the tuning of the evaluation framework. These validation and alignment processes are still under investigation.

References

1. Standish Group: Chaos: A recipe for success. Technical report, Standish Group International (1999)

2. ISO/IEC: ISO 9126-1:2001, Software engineering - Product quality, Part 1: Quality model. ISO/IEC (2001)
3. Davis, A., Overmyer, S., Jordan, K., Caruso, J., Dandashi, F., Dinh, A., Kincaid, G., Ledeboer, G., Reynolds, P., Sitaram, P., Ta, A., Theofanos, M.: Identifying and measuring quality in a software requirements specification. In: Proceedings of the First International Software Metrics Symposium. (1993) 141–152
4. Moody, D.L., Shanks, G.G.: Improving the quality of data models: empirical validation of a quality management framework. Inf. Syst. **28**(6) (2003) 619–650
5. Maes, A., Poels, G.: Evaluating quality of conceptual models based on user perceptions. In: International Conference on Conceptual Modeling - ER 2006, Tucson, Arizona (November 2006) 54–67
6. Krogstie, J.: Integrating the understanding of quality in requirements specification and conceptual modeling. SIGSOFT Softw. Eng. Notes **23**(1) (1998) 86–91
7. Manso, M.E., Genero, M., Piattini, M.: No-redundant metrics for uml class diagram structural complexity. In: CAiSE. Volume 2681 of Lecture Notes in Computer Science., Springer (2003) 127–142
8. Si-Said Cherfi, S., Akoka, J., Comyn-Wattiau, I.: Perceived vs. measured quality of conceptual schemas: An experimental comparison. In: ER (Tutorials, Posters, Panels & Industrial Contributions), Australian Computer Society (2007) 185–190
9. Codd, E.F.: Normalized data structure: A brief tutorial. In: SIGFIDET Workshop, ACM (1971) 1–17
10. Gemino, A., Wand, Y.: Complexity and clarity in conceptual modeling: comparison of mandatory and optional properties. Data Knowl. Eng. **55**(3) (2005) 301–326
11. Burton-Jones, A., Weber, R.: Understanding relationships with attributes in entity-relationship diagrams. In: ICIS '99: Proc. of the 20th international conference on Information Systems, Atlanta, GA, USA (1999) 214–228
12. Lemaitre, J., Hainaut, J.L.: Transformation-based framework for the evaluation and improvement of database schemas. In: Proc. of the 22st International Conference on Advanced Information Systems (CAISE'10, to appear). (2010)
13. Moody, D.: What makes a good diagram? improving the cognitive effectiveness of diagrams in is development. In Knapp, Magyar, eds.: Intl Conf on Information Systems Development, Budapest, Hungary, Springer (August 31-2 2006)
14. Boufares, F., Bennaceur, H.: Consistency problems in er-schemas for database systems. Inf. Sci. **163**(4) (2004) 263–274
15. Damm, F.M., Hansen, B.S., Bruun, H.: On type checking in vdm and related consistency issues. In: VDM '91: Proceedings of the 4th International Symposium of VDM Europe on Formal Software Development-Volume I, London, UK, Springer-Verlag (1991) 45–62
16. Hainaut, J.L.: Bases de donnees. Dunod (2009)
17. Assenova, P., Johannesson, P.: Improving quality in conceptual modelling by the use of schema transformations. In: ER '96: Proc. of the 15th International Conference on Conceptual Modeling, London, UK, Springer-Verlag (1996) 277–291
18. Kurtev, I.: Adaptability of model transformations. PhD thesis, University of Twente, Enschede (2005)

Acknowledgments This doctoral research is being led under the supervision of Pr. Jean-Luc Hainaut, Laboratory of Database Application Engineering - PReCISE Research Center, Faculty of Computer Science, University of Namur, Belgium.