Missing Semantic Annotation in Databases

The Root Cause for Data Integration and Migration Problems in Information Systems

M. Dugas

Summary

Data integration is a well-known grand challenge in information systems. It is highly relevant in medicine because of the multitude of patient data sources. Semantic annotations of data items regarding concept and value domain, based on comprehensive terminologies can facilitate data integration and migration. Therefore it should be implemented in databases from the very beginning.

Correspondence to:
Prof. Dr. Martin Dugas
University of Muenster
Institute of Medical Informatics
Albert-Schweitzer-Campus 1, Gebäude A11
48149 Münster
Germany
E-mail: dugas@uni-muenster.de

Methods Inf Med 2014; 53: 516–517
doi: 10.3414/ME14-04-0002
received: September 29, 2014
accepted: October 14, 2014
epub ahead of print: November 7, 2014

Dear Editor,

data integration and migration in medicine and other domains is a resource-intensive and complicated process. “Data integration has been referred to as a problem as hard as Artificial Intelligence, maybe even harder” [1]. In particular, semantic heterogeneity of data is a major problem: “After major technical problems of connecting different data sources on different machines are solved, the biggest challenge remains: overcoming semantic heterogeneity, that is, overcoming the effect of the same information being stored in different ways” [2]. Integration of heterogeneous data is particularly relevant in medicine [3, 4]. Most current information systems apply relational Structured Query Language (SQL) databases like Oracle [5], DB2 [6] or SQL-Server [7].

Why is it so complicated to transfer data from one information system into another? A key step in data integration and migration between two systems is matching of attributes (synonyms: data elements, data items). Schema matching is done on the concept level (has the attribute the same meaning?) and on the value level (what values of each attribute correspond to each other?).

In the current version of SQL the attribute definition consists of the name of an attribute and a data type, for example:

CREATE TABLE data (size FLOAT, gender TEXT CHECK (gender IN ('male', 'female'))) 

Now, here is the problem: Attribute names can be ambiguous and are therefore inappropriate to capture the meaning of a data element: In this simplistic example, “size” could refer to body height of a person, size of a tumor or size of a car. It is evident that for data integration and migration purposes much more detail about the meaning of this attribute is required than just a name “size”.

Semantic codes can be used to annotate attributes to provide such details about the concept domain. For example, in medicine there are sophisticated terminologies available like SNOMED [8] (>800.000 terms) or the Unified Medical Language System (UMLS) [9] (>2 Mio. terms). In UMLS, an attribute “size” can be annotated with concept code C0005890 to specify body height and code C0475440 to refer to a tumor size.

Currently it is common practice to add free-text explanations to attributes, for instance in the database documentation or in reference tables. However, such free-text descriptions need to be processed manually for matching in the context of data integration and migration. This manual matching is resource-intensive and error-prone, given the high number (typically >1.000) of attributes for information systems and the need for deep domain knowledge to provide appropriate matches.

Semantic annotations are needed not only for the concept domain, but also for the value domain: in the previous example, size has data type “FLOAT”. But again, this specification does not provide an appropriate level of detail for data integration and migration: What is the measurement unit? Is size measured in m, cm or mm? In analogy to the concept domain, semantic annotations could provide this level of detail. Unified Code for Units of Measure (UCUM) [10] can be applied as a terminology for measurement units.

For codelists (such as ‘male’, ‘female’ in the example) SQL provides mechanisms to ensure that only certain values can be entered into a table. Amongst others, the ENUM type or foreign key constraints provide such a mechanism. However, regarding data integration and migration a similar problem arises like with the attribute “size”: Has “male” the same meaning like “m” – in some situations probably yes, in others no. In analogy to semantic annotation of attributes, codelists of attributes can be annotated with semantic codes to...
specify the meaning of all coded values, for example “male” could be annotated with UMLS code C0024554 to specify male gender and “female” with UMLS code C0015780.

Of note, the meaning of attribute names and coded values is so complex, that sometimes a suitable code is not available even in large terminologies like UMLS with millions of codes. In this situation postcoordination can be applied, i.e. several semantic codes are combined to describe the meaning. A medical example would be “father’s disease”, which could be assigned UMLS concepts C0015671 (father) and C0012634 (disease). For this reason it is advisable that lists of semantic codes can be assigned to an attribute or coded value.

Another important aspect of semantic annotations is versioning of terminologies. Most terminologies evolve over time and are regularly updated, which can also cause ambiguities. For this reason it is preferable to provide the version identifier of a terminology and not only the name, for example UMLS2014AA instead of UMLS.

If attributes in information systems would be annotated with semantic codes, matching of these attributes for data integration and migration tasks could be automated [11]: At least data from attributes with identical codes in the concept and the value domain could be directly integrated or migrated between two information systems. This could substantially reduce manual, error-prone attribute matching work.

Taking into account these considerations regarding semantic codes, it would be very helpful for data integration and migration if future versions of SQL would support semantic annotation of attributes, data types and codelists. This would help to overcome the root cause for data integration and migration problems, because these tasks require data semantics, which is now missing in most information systems.

To illustrate this idea, the previous example of a create table statement could be rewritten as:

```sql
CREATE TABLE data (  
  size (UMLS2014AA:C0005890) FLOAT(UCLM1.9:m),  
  gender TEXT CHECK  
  (gender IN  
    ('male' (UMLS2014AA:C0024554),  
    'female' (UMLS2014AA:C0015780))))
```

Such a table would not only provide data, it would contain data plus semantics, i.e. information. It is evident that the topic of semantic annotation is going well beyond medicine to many domains of information systems. Medicine can play a leading role regarding semantic annotation due to the availability of sophisticated terminologies. From my perspective, 44 years after Codd’s seminal publication on the relational model [12] it is time to upgrade SQL data definition language (DDL) to meet the requirements of modern, interoperable, semantically-aware information systems.

References