A Hybrid EAV-Relational Model for Consistent and Scalable Capture of Clinical Research Data

Omar KHAN [a,1], Sarah N. LIM CHOI KEUNG [a], Lei ZHAO [a] and
Theodoros N. ARVANITIS [a]
[a] *Institute of Digital Healthcare, WMG, University of Warwick, Coventry, UK*

Abstract. Many clinical research databases are built for specific purposes and their design is often guided by the requirements of their particular setting. Not only does this lead to issues of interoperability and reusability between research groups in the wider community but, within the project itself, changes and additions to the system could be implemented using an ad hoc approach, which may make the system difficult to maintain and even more difficult to share. In this paper, we outline a hybrid Entity-Attribute-Value and relational model approach for modelling data, in light of frequently changing requirements, which enables the back-end database schema to remain static, improving the extensibility and scalability of an application. The model also facilitates data reuse. The methods used build on the modular architecture previously introduced in the CURe project.

Keywords. Databases, Entity-Attribute-Value, Electronic Health Records, Clinical Research Informatics

1. Introduction

Electronic Health Records (EHRs) have seen increased use within medicine in the last decade and significant resources have been invested into standardizing the terminologies and classifications used within these systems[1]. However, less effort has been devoted to standardizing the structure of the database system used to store the clinical data[2] and many large EHR vendors have used separate approaches[3].

The issue of non-standardization of storage structures is more apparent in clinical research systems. The development of many of these systems are dictated by the requirements of the setting, for which the system is produced, with no systems put in place to handle requirement changes in a structured and methodological way[4]. This is often justified on the basis that the application is only built to meet internal needs or due to the time and resource limitations[5]. This is illustrated by the early development of the SENSELAB project[4], where its application expanded and its database schema grew to become more complex, making application maintenance and extension difficult.

Within the CURe clinical research database framework[6], to respond to system evolution and frequent requirement changes in a structured and scalable fashion, we propose a hybrid Entity-Attribute-Value (EAV)-relational model approach, where EAV tables containing clinical data are implemented alongside a relational database structure, while keeping a modular configuration, separating specialty and generic information.

[1] Corresponding Author. E-mail: m.o.khan@warwick.ac.uk

1.1. CURe

The Comprehensive Unified Research (CURe) framework was set up to support clinical researchers in collecting research data for patients in their care. Clinical researchers have been using a number of stand-alone systems to record research data, and these cannot be shared with other researchers and across studies. This results in redundant data entry and non-standardised ways of representing data. CURe includes a framework for the development of software systems for research. It has been developed to create extensible web-based applications for various medical conditions.

CURe uses a relational schema for the database structure. While this is suitable for most data requirements, it is less so for representing some data elements, such as questionnaires as their standards may change, or the questionnaires themselves may be added or deleted, during the course of an ongoing clinical study, requiring refactoring of the database schema when using a relational database. These evolving requirements require the review of the current CURe structure to address the limitations and add flexibility in developing and maintaining data structures.

1.2. EAV

The Entity-Attribute-Value model is a knowledge representation model where arbitrary information on any object is recorded as a set of attribute-value pairs, e.g. the HTML tag attribute for the English language: “lang=en”. In a simple EAV design, each occurrence of a value is expressed as a row within one EAV table, with entity and attribute IDs used as a key. Metadata tables also need to be present which define the entities and attributes occurring within the EAV table as well as their relationships.

In relational database design however, entities are normally modelled as individual tables with each attribute listed as a column. For an application, where the number of attributes is expected to remain fixed, and nearly all fields are expected to have values, this design is the most appropriate. In this scheme, each instance of the entity is fully encompassed by a single row, meaning that most programming and querying languages only have to make one call to the database to retrieve all information.

Dinu and Nadkarni[7] identify 3 circumstances in which using an EAV table is preferred: (1) large number of attributes but few have values (sparse attribute instances); (2) large number of entities, with few attributes, but few instances of each

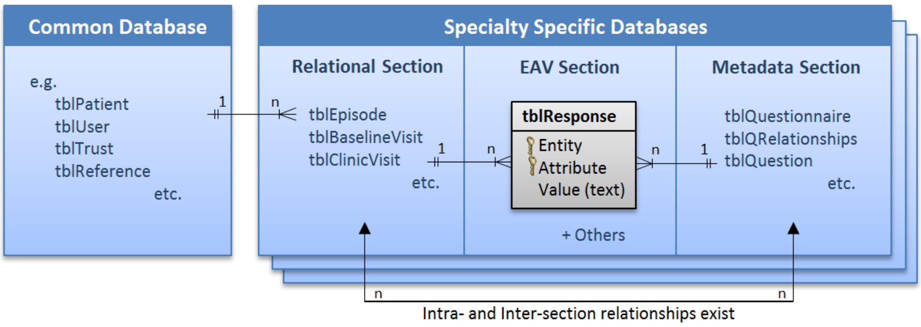

Figure 1. The simplified hybrid EAV-Relational database model for CURe. Common and specialty-specific data elements are still separated as outlined in [5], keeping the application modular. The regularly obtained and stable data items are still represented using a relational structure. Other information is captured using EAV table(s), depending on the type and context of information to be recorded. Information recorded within the EAV table will need to be fully defined using associated metadata tables which can describe type of response expected, relationships between attributes, as well as relationships between attributes and entities modelled in the relational section of the database, such as the questions displayed for certain visit types.

(sparse entity instances); (3) attributes evolve continuously. Using an EAV model in these situations allows all the disparate tables filled with sparse data to be modelled using a single table with only three columns with associated metadata tables resulting in a simplified, static schema which is unaffected by most changes to data requirements.

2. Methods

The use of a conventional structure for relatively static and routinely collected information improves performance of the application when querying the database. Large amounts of clinical information, however, may not be recorded at each clinic visit, such as responses to quality of life or disease progression questionnaires. The questionnaires may also change with time, therefore, satisfying two of the suitability criteria for inclusion in an EAV structure. Figure 1 gives a simplified model of the intended database structure. These fields, when stored using an EAV structure, allow changes and additions to questionnaires to be implemented as new or deleted rows within metadata tables instead of new or deleted columns and tables which would require refactoring of the application code increasing development time significantly. Additionally, the issue of reaching vendor limitations on number of columns per table is avoided. Storing only the actual responses within the EAV table, instead of allocating a complete, relational, row for a semi-filled form also reduces database size.

The rich metadata associated with the EAV table allows individual responses to be fully represented using just an entity ID (such as visitID) and an attribute ID (such as questionID) which, together, form a unique key. This structure then provides a uniform interface for all sets of data within the EAV table to the front-end application, as opposed to a separate interface for each entity.

3. Results and Discussion

Questions and questionnaires are defined within rich metadata tables and responses are captured within the EAV table, this sits alongside the commonly collected information represented using a relational structure (Figure 1). Specialty specific information is kept separate from common information to keep the architecture modular. To avoid overly complex SQL statements when trying to store and retrieve information, a database abstraction layer built on top of Doctrine ORM[8], is used to hide the structure of the database from the application, allowing it access as it would a conventional database through one uniform interface.

Initial investigation indicates that the effect on page load times, within the application, is negligible after optimizing the abstraction model used by the application, through techniques such as batch processing of reads from and writes to the database; however, this will need further investigation to confirm. Using the proposed structure throughout all modules in CURe could also reduce the number of tables by approximately 50%, even when including the extra metadata tables added, and the uniform interface could reduce the lines of code in the application reading and writing data from a few thousand to fewer than 50.

Discussion: Other methods to overcome the EAV performance impact include dividing SQL queries to reduce the impact on memory[9] or using a relational data warehouse by storing pivoted data in a separate database and allowing attribute-centric ad-hoc queries to run retrospectively, syncing the live data periodically. Only running entity-centric queries, e.g. routine queries by the application on the live database would have a lower impact on performance as the EAV tables are already indexed by entity[10]. A limitation of using EAV is that validity is difficult to ensure[11] as all responses are held within the same column and no restrictions are made on data type or size. This can be overcome by defining the type in the metadata and splitting the data into separate EAV tables by type, so that the database field type limits and enforces the response type. The metadata tables can also contain expressions, defining valid responses which can be dynamically invoked by the application.

A similar approach to EHR structure was taken by Tange et al.[12]: 'paragraph types' were used as attributes for data in patient notes. However, the structure used was adapted for the searching of medical narratives, rather than storage of structured data.

4. Conclusions and Future Work

An EAV structure in conjunction with the modular configuration already used within CURe would provide many benefits during the development lifecycle, such as a simplified database structure and more streamlined change management in addition to increased extensibility of the application, but these advantages come at the cost increased initial development time, performance impact and complex queries. The benefits and limitations of the hybrid EAV-relational model for the CURe framework will be evaluated, and additional methods will be investigated in our future work.

Acknowledgement: This work has been funded in part by the West Midlands Academic Health Science Network.

References

[1] B. Fernando, D. Kalra, Z. Morrison et al., Benefits and risks of structuring and/or coding the presenting patient history in the electronic health record: systematic review, *BMJ Qual Saf* **21** (2012), 337-346.

[2] A. Begoyan, An Overview of Interoperability Standards for Electronic Health Records, *IDPT* (2007).

[3] C. Friedman, G. Hripcsak, S.B. Johnson, et al., A Generalized Relational Schema for an Integrated Clinical Patient Database, *Proc Annu Symp Comput Appl Med Care* (1990), 335-339.

[4] P.M. Nadkarni et al., Organization of Heterogeneous Scientific Data Using the EAV/CR Representation, *J Am Med Inform Assoc* **6** (1999), 478-493.

[5] I. Ogunsina, S.N. Lim Choi Keung, J. Rossiter et al, An Extensible Model for Multi-Specialty Patient Record Systems in Clinical Research, *ICICHT SAMOS* (2012).

[6] S. Lim Choi Keung, I. Ogunsina, J. Rossiter, L. Zhao, T. Arvanitis, and G. Langford, Modelling Patient Medication Usage in Secondary Care Research Systems, *eTELEMED* (2013), pp. 25–28.

[7] V. Dinu and P. Nadkarni, Guidelines for the Effective Use of Entity-Attribute-Value Modelling for Biomedical Databases, *Int J Med Inform.* **76** (2007), 769-779.

[8] Doctrine Project, Object Relational Mapper, 2014 [March 20, 2014]. web site: http://www.doctrine-project.org/projects/orm.html

[9] R.S. Chen, P. Nadkarni, L. Marenco et al., Exploring Performance Issues for a Clinical Database Organised Using an Entity-Attribute-Value Representation, *J Am Med Inform Assoc* **7** (2000), 475-487.

[10] P.M. Nadkarni and C. Brandt, Data Extraction and Ad Hoc Query of an Entity-Attribute-Value Database, *J Am Med Inform Assoc* **5** (1998), 511-527.

[11] R. Lenz, T. Elstner, H. Siegele, et al., A Practical Approach to Process Support in Health Information Systems, *J Am Med Inform Assoc* **9** (2002), 571-585.

[12] H.J. Tange et al., An experimental electronic medical-record system with multiple views on medical narratives, *Computer Methods and Programs in Biomedicine* **54** (1997), 157-172.