PURPOSE-DRIVEN APPROACH FOR FLEXIBLE STRUCTURE-INDEPENDENT DATABASE DESIGN

Youri I. Rogozov, Alexander S. Sviridov, Sergey A. Kutcherov
Taganrog Institute of Technology, Southern Federal University
22 Chekhov Street, 347928, Taganrog, Russia

Wladimir Bodrow
Department Business Computing, University of Applied Sciences Berlin
8 Tresckowalle Street, 10318, Berlin, Germany

Keywords: Semi structured data, Relational database model, Database construction method, Structure-independent database.

Abstract: This paper presents a purpose-driven approach for development of flexible databases outgoing from the relational database concept. Based on carried out analysis of both relational and entity-attribute-value database models the aspects essential for the described purpose-driven approach are defined. The necessary requirements to be satisfied by structure-independent databases are derived and discussed in detail. Several implementations of structure-independent databases using the suggested approach have been realized and presented as well. The improvement of relational database model based on proposed structure-independent database approach is formulated.

1 INTRODUCTION

The problem of storing changeable relational data structures appears for instance as concomitant in the research and development projects focused on creating of CASE-tools for automated development of inquiry and communication systems. In such systems a database is to be designed under conditions of full or partial uncertainty regarding users' data that will be stored in it. Due to this vagueness the application of the relational data model turns out to be difficult. The crucial point is that the reflection of the stored data structure at logical level to the relational database structure at physical level results in the significant growing of the labour intensity to maintain this physical structure. Furthermore it leads to data surplus and normalization rules violation. The complicated daunting task which can be completed only by a group of highly qualified database administrators helps to face with such difficulties appearing in the databases that are able to reach up to several hundreds of tables. This contradicts the idea of automated development of a flexible information system presupposing the opportunity to enter changes into the system by users themselves as many times as necessary. Another defined ambition is the reducing of the number of software engineering and database specialists involved in the particular project.

Among known approaches to solve the problem it is possible to single out two main ones, i.e.:

1. Applying the techniques of storing semi structured data, such as native XML databases, Object-Exchange Model (Bourret, 2010, Agrawal et al., 1995);
2. Creating and using database static physical relational structures independent from the stored data logical structure (Paley, 2002, Tenzer, 2001, Bannikov, 2010, Polishchuk and Thcernykh, 2009).

The initial purpose of similar technologies for dealing with changeable data is in accordance with the first approach. The list of existing concepts may tend to an independent research of their efficiency for solution of the mentioned task. It's worth being noted that the basic technologies of this field, e.g.

XML, are a modified hierarchical data model. Thus the basic technologies contain the disadvantages of this model (Date, 2008). The relational database model eliminates these disadvantages.

The second approach is based on the differentiation concept of database at physical and logical levels. It helps to use all the optimization method diversity worked out for years of the relational tables application. The used set of static tables at the physical level allows keeping the relational technologies. The consequent database differentiation at physical and logical levels allows simultaneously working with different database models; as relational model, object-oriented model, hierarchical model etc. The evident shortcomings of this approach are the insignificant efficiency decrease and the complication of data access mechanisms.

Presented research describes the purpose-driven approach for the storage of data structures with high degree of ability to be changed. It is based on generation and use of static physical relational database structures that are independent from the stored data at logical level. The approach allows the realization of different structurally independent databases using the relational database models according to the particular application declared by developer. It is based on the implementation analysis of the relational model at physical level and entity-attribute-value model which allows the specifications revealing of the structure-independent database. The following steps lead to compliance with identified requirements and describe the implementation of the physical database structure.

2 DATA MODELS ANALISYS

The start point of the proposed approach is the implementation analysis of the independence of physical database structure from the logical structure of the stored data. To make the peculiarities obvious, one can consider every model implementation through the three-dimensional matrix images in the basis "entity – attribute – instance".

2.1 Relational Model

The basis of the relational model (Codd, 1970) implementation consists in storing data in the form of tables (entities) with a finite set of fields (attributes) both at the logical and physical levels. The set of tables and their interconnections are specified at the design stage. The continuous line represents the growth of data amount in the database (instances), entailing changes in the physical structure (entities), and the dotted line signs the absence of any change.

The relational model implementation can be spatially represented applying the taken symbol and the basis "entity-attribute-instance" as a three-dimensional random figure (Fig. 1). This figure cells contain attribute values for the entity instances as follows.

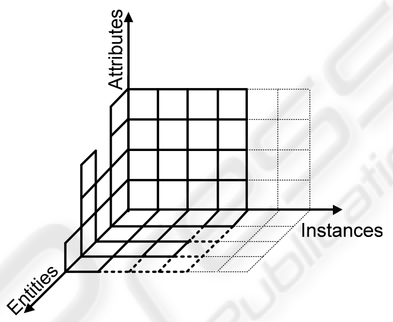

Figure 1. Spatial representation of the relational data model implementation in the "entity-attribute-instance" basis.

Figure 1 allows the conclusion that in relational model physical database structure is independent from the quantity of entity instances. It is justifies by the obvious storage of metadata denoting the ones stored earlier in table with field titles and tables ties. The relational model rejects the metadata change at any moment of the database use without physical interference. This provides its being inapplicable for storing changeable data structures.

2.2 Entity-Attribute-Value Model

Unlike the relational model, the entity-attribute-value (EAV) data model (Stead, 1982, Nadkarni, 2010) allows to change metadata partially (entity attributes) during the process of database utilization. Decisive database growth directions according to this model can be visualized based on extended image used before in the "entity – attribute – instance" basis (Figure 2).

The additional cells presented by dotted lines visualize the possibility of dynamical extension of attributes as for entities as for instances. In the conclusion Figure 2 underlines that the physical database structure doesn't depend on quantity of attributes or entities and can be changed based on instance quality. This implementation ability is anchored direct within the changeable metadata

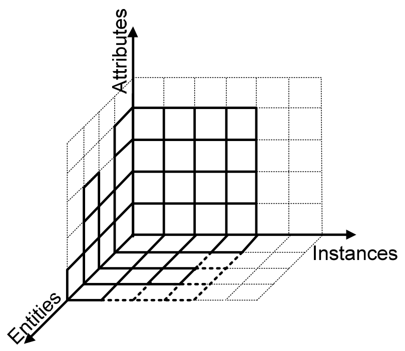

Figure 2. Spatial representation of the EAV data model implementation in the "entity-attribute-instance" basis.

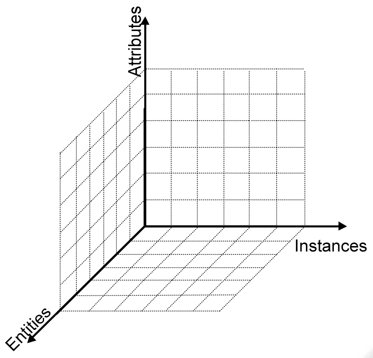

Figure 3. Spatial representation of structure-independent database in "entity-attribute-instance" basis.

inside the database. A special table is generated for storing attributes and their descriptions, the attributes values are kept apart (Dinu and Nadkarni, 2007, Brandt et al., 2002). In addition to that different instances of the same entity can contain variable quantity of attributes. In other words, "instance – attribute" basis is a sparse matrix containing only really needed attribute values.

At the same time the EAV database is positioned as one of stringent specification and so is often used in fields with predetermined unchangeable number of entities and parts of their attributes, e.g. in medical information systems or e-commerce systems. The fitting of the physical database structure to the stored data structure justifies the reasonability of the EAV model characteristics its applicability for structure-independent database development focused on purposes in particular project.

2.3 Requirements to Structure-Independent Databases

The data structure representation as a sparse matrix provides its dynamic change. Direct storage of structural metadata in the database permits its independence at both physical and logical levels of implementation. The discussion above allows the development concept for structure-independent database using the three-dimensional representation in the "entity-attribute-instance" basis (Figure 3).

According to the Figure 3 the structure-independent database do not need the description of the subject domain beforehand, it is ready for being used even without precise definition of all entities or attributes. This way the full differentiation of the physical and logical implementation levels is achieved:

- logic structure characterizing user data is described not by the physical database structure, but by the stored metadata;
- the physical database structure is fixed and relational technologies can be applied for its development.

The data models analysis lets form the **requirements** for the structure-independent database, implemented in accordance with the relational technologies:

1. *Metadata that define the logical database structure are stored directly in the database as data sets in the relational tables and build a metadata substructure.* Independence between the logic and physical structure is provided by refuse from metadata explicit determine as table titles, fields and table links;
2. *Data are grouped in tables with respect to types. Tables contain attributes indicating membership in a reference to the metadata tables and compose a certain subsystem.* The field title where values are stored are not attribute title;
3. *The structure of the database is represented by a sparse matrix.* The logical structure of data is generated afterwards according to the project purposes. It contains only entities and attributes, really used in the application. The logical data structure can be added or changed anytime;
4. *The total number of database tables at the physical level is fixed and depends just on metadata tables number and used data types, but it doesn't depend on the logical structure of the stored data.* The restricted number of tables on the physical level is the key to implementation productivity. It decreases the complexity requirements for the database,

which is important for the maintenance during the exploitation of the solution.

Analysis of certain structure-independent database models with applying relational technology (Paley, 2002, Tenzer, 2001, Bannikov, 2010, Polishchuk and Thcernykh, 2009) reveals that the models meet the declared application requirements. Consequently the realized implementation depends on project and/or developer's purposes. This allows the formulation of an approach for the development of flexible structure-independent database models based on relational technologies according to the project purposes. (Provided that developer follows exact the project purposes and owns the professional skill for its adequate translations into the database to be realized.)

3 APPROACH OF STRUCTURE-INDEPENDENT DATABASE CONSTRUCTION

Irrespective of the developer's professional skill the sequence of steps leading to the reception of a necessary set of connected relational tables can be formulated. This sequence meets the requirements stated above and is described in detail below. The given set of tables is a physical level of the structure-independent database. The sequence of steps represents **an approach of structure-independent databases construction based on relational technology.**

3.1 The Approach Formulation

It's most convenient to formulate an approach, through its division into some steps with respect to defined above requirements to be complied. **To meet the both first requirements it is necessary:**

1. *To define a set of the metadata necessary for the description of logic structure of a database;*
2. *To define structure of metadata and to realize them in the form of a fixed set of the connected relational tables which make a subschema of the metadata.* The given step includes:
   - division of the figured out metadata set into the groups based on accessory (for example, describing entity and the attributes, describing communications between entities etc.);
   - creating a separate relational table for each group;
   - specification of links between tables (for example, between the table of communications and description tables of entities and attributes);

To fulfill the following two requirements it is necessary:

3. *To define types of data which will be used for storage of attributes' values*;
4. *To develop a subschema of data, in the form of a final set of the same relational tables not connected among themselves for storage of attributes' values.* The quantity of tables corresponds to the quantity of used types of data. It is expedient to store values in the form of the triplet "entity-attribute-instance", where the first two elements represent the reference links to records in tables of the metadata. It allows to present data structure in the form of sparse matrixes;

To construct a structure-independent database finally it is necessary:

5. To implement the database physical structure based on definition and specification of the communications between tables of metadata subschema and a data subschema;

Fulfillment of the **third and fourth** requirements is a consequence of fulfillment of the both first:

- Representation of the stored data structure in the form of sparse matrixes is realized due to the storage of values in the form of the triplets (the sparse matrix of instances) and storage of metadata in a database (the sparse matrixes of entities and attributes);
- Independence between physical and logic level is realized due to storage of the metadata in the same database.

This sequence of the realization steps anticipates working out of user's (logic) structures of data, executed according to the technology defined by the developer of a database.

3.2 Utilization of the Approach

The described approach supports the construction of flexible structure-independent database based on relational technologies depending on particular purpose to be achieved in the project. For the validation of the proposed approach we will consider an example with known data structure on its basis and also we will result own variant.

Example 1. *Let's imagine that project purpose is the creation of universal relational structure for processing and storage of semistructured data in terms of object-oriented technology – in the form of classes and objects.* According to the approach the reception of the given structure looks as follows:

1. We allocate a metadata set:
   - Names of classes' objects and attributes;
   - Hierarchies of classes;
   - Links between objects;
   - Relations between objects and classes.
2. We form the structure of metadata tables according to the logic accessory of metadata:
   - The attributes reference table – tbl_attdef table;
   - The object classes reference table – the tbl_classdef table;
   - The links between objects reference table – the tbl_links table;
   - The objects reference table – the tbl_obj table.
3. At the conceptual level it is enough to specify only one type of the data – line. The other types of data use will lead to increase of identical data tables quantity;
4. The data subschema is realized in the form of one table – tbl_attval;
5. Communications between subschema tables of metadata and the data will be arranged according to key values of unique id.

Thus, on the basis of the formulated approach the model of the database structure (Fig. 4), realizing the specified purpose and meeting the formulated requirements is constructed.

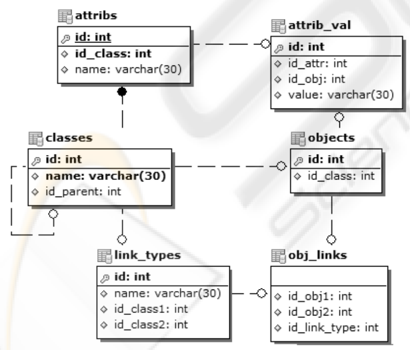

Figure 4. Model of database structure for storing objects.

A similar structure named the expanded relational scheme for processing of quasi-structured data, is come across in references (Paley, 2002). Remaining within the limits of purpose of data storage in terms of object-oriented technology, it is possible to obtain a set of various structures – each developer chooses the metadata according to his understanding of the project purpose. In references (Tenzer, 2001) and (Bannikov, 2010) the authors result the models of the structurally-independent database distinct from resulted above, but based on specified purpose.

Example 2. *The project purpose is the creating of a universal relational structure for storing XML-data in terms of object-oriented technology*. According to the presented approach the given structure is produced the following way:

1. Singling out a metadata set: objects and attributes class names; enabled links between the classes, links between objects.
2. The metadata table structure is to be formed according to the logical metadata attachments:
   - The object class reference table – the Classes table;
   - The reference table of enabled links between classes – the link_types table;
   - The reference table of links between objects – the obj_link table.
3. All the data can be stored as XML-objects. To do this developer may pick the BLOB or XML-data if the DBMS used allows it;
4. The data subschema is implemented in the form of a single table – objects – with the following set fields: a unique id, an object id, a value;
5. The connections between the metadata and data subschema tables are arranged according to the unique id key values.

Thus, on the basis of the described approach we receive a database structure (Fig. 5) implementing the project purpose and meeting the formulated demands.

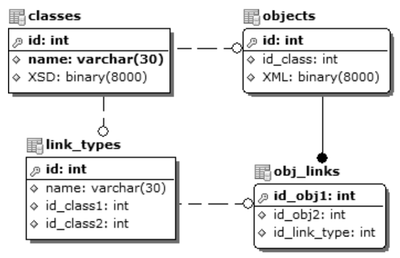

Figure 5. Model of database structure for storing XML-objects.

The model of database structure termed as a data-storage model using the XML-schema is described in the paper (Polishchuk and Thcernykh, 2009).

4 SIDB

Within the framework of presented research *project purposes are considered as a central component for storage of relational data structures with a high degree of changeability in databases with a static physical structure*. In this chapter we view the suggested approach applied to construction of the model for structure-independent databases (SIDB).
1. The metadata necessary for SIDB model include: entities and attributes names, entity-attribute connections, entity-entity connections, unique synonyms, attributes and entities brief description, logical deletion feature, order applied by the users' sorting, entities and attributes structure hierarchy.
2. Metadata can be subdivided into the following groups with the corresponding tables:

- The hierarchical reference table including all metadata except for those determining the “entity-attributes” and “entity-entity” connections. The metadata can be effectively applied both to entities and attributes, so they can be represented with one table – tSprDicrionary;
- The entities’ attributes reference table which records will refer to the hierarchical reference table connectible elements – table tEntityAttr;
- The entity instance reference table referring to the hierarchical reference table elements – tEntity table.

The speciality of the metadata describing the entity-entity connection lies in their appearance when the direct connection between entities’ instances appears. It is advisable to enter the appropriate data type - a reference to entity and to keep this kind of metadata in the data subschema.
3. A set of data types dealt with by SIDB, includes: string, number, date, BLOB and the «Entity» data type which is an entity reference link used as metadata.
4. To store data it is appropriate to use “entity-attribute-instance” triple sets (in papers Paley, 2002, Tenzer, 2001, Bannikov, 2010) the “object-attribute-value” triple sets are used). In this case entities and attributes are set implicitly as the hierarchical reference table element links. The subschema encompasses the “tEntityData*” tables;
5. The connections between the metadata subschema tables are arranged by the unique id key values.

The SIDB model obtained as a result of applying the purpose-driven approach (Fig. 6.) enables to store highly changeable data relational structures of any complexity.

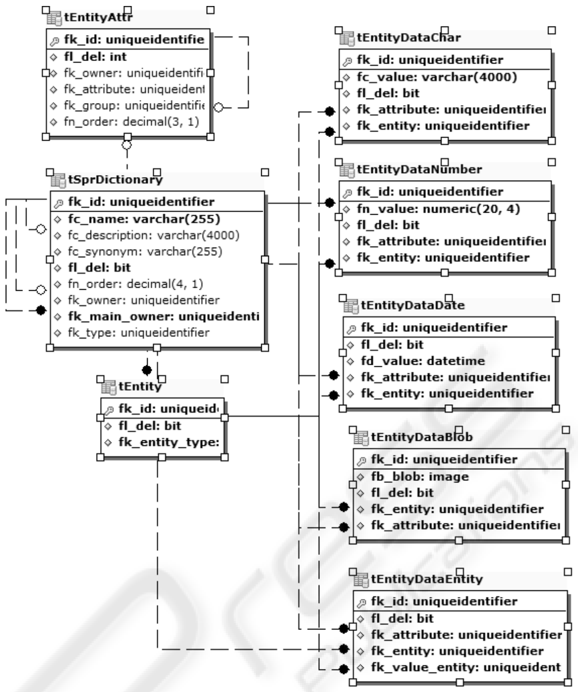

Figure 6. Model of SIDB database structure.

5 CONCLUSIONS

The approach described in this paper enables to construct SIDBs for storage data with changeable structures applying relational technology. There are considerable advantages of worked out SIDBs:

- Limiting the structural complexity of database queries by means of fixing the amount of relational tables at the physical level;
- Enhancing the speed of data operations by using relational technology;
- The lack of the database interdependence at physical and logical level provides the significant increasing of its flexibility;
- Low maintenance cost without involving any professional database specialists.

Applying the approach presented in the paper a number of various SIDBs can be developed via relational technology. The implementation of each stage depends on a project purpose and its understanding by developer. It is not confined to the examples treated in the paper and can be applied to models implementation analysis outgoing from the independence between the physical database structure and the stored data logic structure. To make the particularities obvious, developer has to consider every model implementation through the three-dimensional matrix images in the basis “entity – attribute – instance”.

REFERENCES

Bourret, R., 2010. *XML and Databases*. // http://rpbourret.com, Last call 2010

Agrawal, R., Gehani, N., Srinivasan J., 1995. OdeView: The Y. Papakonstantinou, H. Garcia-Molina, and J. Widom. *Object Exchange Across Heterogeneous Information Sources*. Proceedings of the Eleventh International Conference on Data Engineering, pp. 251-260, Taipei, Taiwan

Paley, D., 2002. *Quasistructure Data Modeling* // Open Systems. № 09, pp. 57-64.

Tenzer, A., 2001. *A Database as an Object Storage* // Computer-Press. №8, pp. 144-145.

Bannikov, N., 2010. *The “Universal Database” Project* // www.stikriz.narod.ru, Last call 2010

Polishchuk, Yo.V. , Thcernykh, T.A. 2009. *The Modeling of Information storage subsystems for Storing Quasistructured objects* // Modern Technology, Information Technology. №1, pp. 66-71

Date, C.J., 2008. *An Introduction to Database Systems*, Eighth Edition. ISBN 0321197844

Codd, E.F., 1970. *A Relational Model of Data for Large Shared Data Banks*// Communications of the ACM, Volume 13, Number 6

Stead, W.W., Hammond, W.E., Straube, M.J., 1982. *A Chartless Record—Is It Adequate?* // Proceedings of the Annual Symposium on Computer Application in Medical Care, pp. 89–94.

Nadkarni, P., 2010. *An Introduction to Entity-Attribute-Value Design for Generic Clinical Study Data Management Systems* // Center for Medical Informatics, Yale University Medical School. http://med.yale.edu/., Last call 2010

Dinu,V., Nadkarni, P., 2007. *Guidelines for the effective use of entity–attribute–value modeling for biomedical databases*. Elsevier, Ireland. International journal of medical informatics № 76. pp. 769–779.

Brandt, C., Morse, R., Matthews, K., Sun, K., Deshpande, A., Gadagkar, R., Cohen, D., Miller, P., Nadkarni P., 2002. *Metadata-driven creation of data marts from an EAV-modeled clinical research database*. Elsevier, Ireland. International journal of medical informatics № 65. pp. 225–241.