Mapping from Relational Database to Ontology Based on Initial Model

Vladislav V. Moiseev[a], Ivan A. Zagaichuk[a]

[a]*Ulyanovsk State Technical University, Ulyanovsk, Russia*

Abstract

The article describes the ontology compilation from the relational database. The process is based on the model called initial. The model is applicable for the design and redesign of existing software products and tuning of the final ontology, including its size reduction. The described model allows obtaining data from a relational schema. The data can be subsequently used for other more flexible formalizations. Moreover, the model can be extended by new parameters which cannot be derived from the schema but be quite useful for design tasks. The paper brings in a short overview of different approaches to ontology compilation based on relational data sources. Additionally, two extra strategies are presented. Finally, the article contains an example of these strategies implementation which is measured by performance tests on two different data sources.

Keywords

relational data schema, data model, ontology, OWL, OWL API

1. Introduction

The new redesign methods are highly appreciated due to the increasing complexity of software systems. Moreover, new methods are expected to be more performed and optimal.

Initially, mathematical logic and computer science formed an idea of a formal subject definition [1]. Thus, the theory of ontologies started. The ontology is a specific formalization that can define relationships and contradictions between the concepts and other elements of subject domain. The ontologies are general-purpose structures that can be a base for other types of domain models. For instance, it can be an object-oriented UML schema, relational model, or another kind of formalization that is necessary for information systems. Additionally, ontologies help to build a solid integration data system. Nowadays, there are many text data exchange standards that exist, for example, the SOAP protocol. The SOAP's schema is usually described by XSD-documents [2].

Axiomatics makes ontologies useful for the design and redesign of software products.

Nevertheless, there are many software products that have no subject domain ontologies. That can be explained as a lot of products are not matured and their domains seem to be internally consistent. As a result, the need for ontology might be unnecessary action. However,

Russian Advances in Artificial Intelligence: selected contributions to the Russian Conference on Artificial intelligence (RCAI 2020), October 10-16, 2020, Moscow, Russia

v.v.moiseev@ulstu.ru (V.V. Moiseev); zagaichuk.ivan@gmail.com (I.A. Zagaichuk)

0000-0003-2671-4653 (V.V. Moiseev); 0000-0002-4721-5960 (I.A. Zagaichuk)

the need appears and the initial data required for the compilation of ontology lies in the relational data storage. Thus, relational storage can be converted to ontological resource due to the need of resolving contradictions within the redesign process. However, the conversion might be complicated due to the variability between the initial data source and the result [3]. For example, the process of relational database conversion into ontology (using approached [4, 5]) can be compromised by several reasons:

- It seems not possible to extract all data about the subject domain from the relational model;
- Some elements of the relational model are redundant;
- Each relational schema has features of model representation.

The data extraction of a subject domain from a relational database is high-loaded for large information systems and represents an additional difficulty.

Therefore, to simplify the redesign process by ontological analysis, it is recomended to define the model that provides several abilities of:

- The model transformation into an ontological resource;
- The generation of other formalizations besides ontology;
- The independence of the implementations of relational databases;
- The exclusion of some entities and other elements of a relational data schema without losing model consistency;
- Supplementation of the relational model with additional properties that may be useful for the compilation of an ontological resource.

This model is called an initial because it is a base for further ontology compilation of subject domain.

2. The overview of ontology compilation strategies based on relational databases

There are many strategies for transforming relational schemas into an ontology (e.g. the concept of direct mapping [8, 9]). It describes the matching of relation schema structures to metadata description technology RDF (Resource Description Framework) developed by W3C. The formal mapping description is also given. The fine-tuning of the initial and result data set cannot be provided by the model. That fully describes direct mapping.

W3C develops another technology named as R2RML [10] for relational schema transformation into metadata. The schema helps provide the fine-tuning in a comparison with direct mapping. The W3C recommendation does not contain the formal language description of R2RML. However, it has various examples of data conversion.

There are also works describing the transformation of relational data schemas in ontology, and not just in RDF metadata. For example, [11] gives an overview of current developments in such a transformation. This article also undercovers the main types of entities and the basic rules of transformation and its own data transformation strategy.

The following entity types can be distinguished according to [11]:

- The strong entities without foreign keys (references);
- The strong entities with foreign keys;
- The weak entities:
  - Foreign and primary keys are the same;
  - The composite primary key that contains various foreign keys for all fields;
  - The composite primary key is and the attributes of the initial entity are not duplicated in referenced ones;
  - The composite primary key is and some simple attributes can be duplicated in referenced entities.

The categories can be used for the correct definition of transformation rules. For instance, the entities that have not foreign keys can be converted into ordinary ontology classes.

It is also necessary to take into account the constraints of entities. For example, the NOT NULL constraint can be specified by MinCardinality in OWL notation, the UNIQUE constraint can be assigned by InverseFunctionalProperty [11].

Some transformation algorithms from relational DBMS to ontology (e.g. [5]) propose a phased transition: classification of tables, their mapping, mapping columns to data properties, mapping relationships and mapping constraints.

3. The description of the initial data model for formation of ontologies

The initial data model for ontology compilation is denoted as M. Thus, the model can be expressed as:

$$M =< E, R > \tag{1}$$

where:

$E = \{E_1, E_2, \dots, E_n\}$ - entities of relational schema
$R = \{R_1, R_2, \dots, R_m\}$ - the relations between entities
n - the number of entities
m- the number of relations

The entity can be described as:

$$E_i = (Name, A, C) \tag{2}$$

where:

$A = \{A_1, A_2, \dots, A_n\}$ - many attributes of entity i
$C = \{C_1, C_2, \dots, C_n\}$ - many constraints of entity i
n - the number of attributes of entity i

The entity attribute can be presented as:

$$A_i = (Num, Name, Type, P) \tag{3}$$

where:

Num is an ordered number of attribute i within the cortege of entity attributes
$Name$ - the name of attribute which must be unique within the entity
$Type = [Int, Float, Double, Char, Decimal, Bit]$ - data type
$P = \{P_1, P_2, \dots, P_n\}$ - many additional attribute properties

Additional attribute properties may be as follows:

- maximum length (for string fields);
- maximum number of digits (for integer and decimal numbers);
- number of decimal places (for decimal numbers);
- text description.

The constraint of entity P can be written as:

$$P_i = F(C) \tag{4}$$

where:

C - many attributes that constraint was applied to
F - the function that defines the type of constraint

The constraints can be described as follows:

- F_PK is a primary key constraint;
- F_NN is a NOT NULL constraint;
- F_UQ is a unique constraint;

- F_FK is a foreign key constraint.

The entity relations can be described as:

$$R = D(F_{FK,i}, F_{PK,j}, RP) \tag{5}$$

where:

D is a function that describes the type of relation;
$F_{FK,i}$ is a foreign key constraint of entity i;
$F_{PK,j}$ is a primary key constraint of entity j;
$RP = \{RP_1, RP_2, \dots, RP_n\}$ - many of auxiliary parameters of relation.

The conclusion that is based on (2), (3) and (5) states the definition of the R relation can be simplified as:

$$R = E_i \circ_R E_j \ : \ \{E_i, E_j\} \subseteq E \tag{6}$$

where:

$\circ_R$ is the operation of binding entities E_i and E_j.

It is proposed to record the resulting model using the XML markup language using the following rules:

- $Schema$ is a key element of the model;
- $Entity$ is the descriptive element;
- $Column$ is an attribute for description;
- $Schema$ and Name mean the names of entities, attributes, and constraints;
- $IsPrimaryKey$ is an attribute of a primary key constraint;
- $IsNullable$ is NOT NULL constraint;
- $ForeignKey$ is a constraint for the foreign keys.

4. Ontology compilation strategies

There are two proposed options for the ontology compilation:

1. Filling the basic ontology of a relational data warehouse by individuals.
2. Ontology creation with entity classes and subsequent filling by individual records.

The first approach is the simplest but least flexible. In this case, there is a basic ontology with the classes “entity”, “relationship”, “entity field”, “procedure”, “procedure argument”, “presentation”, etc. Object relations are set for the classes. For example, the basic transitive object property “is part” is defined, which is used for relationships between entities and their fields, as well as procedures with their own arguments.

The particular elements of the relational schema are filled into the ontology as individuals (individuals) with the axioms of class membership (Class Assertion).

This approach has a significant ability to represent any relational schema while expanding the initial ontology. This strategy allows for making similar ontologies from software products of the same subject domains. Thus, they will differ only by sets of individuals and related class axioms. Finally, that helps to simplify the further analysis for integration systems design.

The main drawback of this approach appears while expanding the basic ontology. In other words, a software update is required to provide new individuals in accordance with the initial schema. Moreover, this approach does not support all the opportunities provided by ontologies. For example, it is difficult to search for contradictions because the relationships between entities are represented as individuals, and not as object properties with their attributes.

The second proposed strategy differs from the first that the initial data model is used as a base for a separate ontology resource that is not associated with any basic one. In this case, the ontology classes are entities, procedures, and other elements of the initial relational schema. The entity relations convert into object properties, the fields of entities, and procedure arguments are into the data properties. There is also possible to use relational data records to fill the ontology that are used as individuals for conducting an ontological resource solvability.

The vantage of this strategy is the flexibility and application of the entire ontological tools. Although the relational schema does not support all required data for the ontology compilation, this can be achieved by initial schema extension. For example, it is possible to define the attributes (e.g. transitivity, symmetry, etc.) of object properties for the initial schema. The resulting ontology will most fully describe the domain already at the initial stage of compilation.

The disadvantage of this strategy is the need to supplement the initial schema after it is obtained from the relational database or to refine the resulting ontology after it is get.

5. The implementation of the proposed strategies

Since the chosen strategies include two stages (the formation of the initial model and the compilation of the ontology), the implementation was also divided into two stages.

Firstly, a generator and designer of the initial data model have been developed with Visual C# and Windows Forms technology that allows rapid UI development. Microsoft SQL Server 2012 was chosen as a data management tool based on the relational data paradigm. The DBMS allows easy access to the description of the data schema. However, the choice of a particular DBMS was up to the experience of the authors, rather than the significant advantages of the current DBMS system over analogues. The serialization of the initial model is performed using the XML language with an attributive approach to the description of the properties of elements. That also helped to simplify the processes of work with the data model and reduced size of serialized data.

Java and OWL API libraries were used for the generation of ontologies. The choice is justified by the prevalence, simplicity, and open-source type. The software helps to deserialize the initial models and generates, based on settings, the ontology in OWL XML. The format was chosen due to its readability, prevalence, and convenient debugging and overview tools (protégé and others).

The basic ontology must be opened for the implementation of the first strategy. It is done using the following code:

```
OWLOntologyManager manager =
  OWLManager.createOWLOntologyManager();
OWLOntology ontology = manager.loadOntologyFromOntologyDocument(
  new File("sql-base-ontology-1.0.0.owl"));
OWLDataFactory df = manager.getOWLDataFactory();
```

The following are the main classes of the basic ontology (parts of duplicated code omitted):

```
final String ontologyIRI = "urn:vladdy-moses.sql-base-ontology";
OWLClass entityTableType = df.getOWLClass(ontologyIRI + "#Table");
// ...
OWLDataProperty schemaDataProp = df.getOWLDataProperty(
  ontologyIRI + "#schema_name");
// ...
```

A named individual is created for each entity of relational storage and the following set of axioms is applied to it:

- *DataPropertyAssertionAxiom* is used to determine the data schema and entity name;
- *ClassAssertionAxiom* is used to assign an entity to an ontology class (for example, an entity can be a table, view, or procedure).

A named individual is created for each entity field and the following set of axioms is determined on it:

- *DataPropertyAssertionAxiom* is used to determine the field name;
- *ClassAssertionAxiom* is used to assign an individual to a class of entity fields;
- *ObjectPropertyAssertionAxiom* is used to associate a field with an entity.

Each relationship between entities (defined by a foreign key in the traditional schema) is also represented by a named individual with the following set of axioms:

- *DataPropertyAssertionAxiom* is used to determine the data schema and name of the relationship;
- *ClassAssertionAxiom* is used to assign the relationship to an ontology class;

- *ObjectPropertyAssertionAxiom* is used to describe whether a relation belongs to an entity field on the one hand and to the primary key of an external entity on the other.

The new resource will be created in a similar way as in the case with the secondly proposed strategy.

The class with axiom DeclarationAxiom is instantiated for each entity.

Each entity's property is converted to the data property and supports the next axioms:

- *DeclarationAxiom*;
- *DataPropertyDomainAxiom* is used to determine the domain of the data property.

Each relationship of the relational model is converted into an object property with the following axioms:

- *DeclarationAxiom*;
- *ObjectPropertyDomainAxiom* is used to determine the domain of an object property.

This strategy allows generating individuals for each database record based on this relational schema. In this case, each record has an individual with the following axioms:

- *ClassAssertionAxiom* is used to associate a record with a specific class that describes the entity;
- *ObjectPropertyAssertionAxiom* is used for foreign keys;
- *DataPropertyAssertionAxiom* is used for the calculation of each non-null field.

The early presented strategies saving procedures coincide with the saving of the ontology and occurs as follows:

```
OWLOntologyManager manager
  = OWLManager.createOWLOntologyManager();
OWLDocumentFormat documentFormat = new OWLXMLDocumentFormat();
manager.saveOntology(ontology, documentFormat,
  IRI.create(new File("result.owl")));
```

6. Generation result

It is necessary to take several databases from various subject areas to test the operability of the proposed ontology strategies compilation from a relational data schema.

The data of addresses and houses from the Russian Federal Information Address System, Ulyanovsk Oblast (denoted as *DB*1) and data of overhaul in Ulyanovsk region (denoted as *DB*2) will be used. Both relational storages correspond to 3rd normal form. Therefore, there would not be any difficulties with the compilation of ontologies.

Table 1
The characteristics of relational storages.

Parameter	DB1	DB2
The number of tables	2	15
The number of fields	67	338
The number of foreign keys	5	19
The number of records	651,883	2,126,054
Data DBMS size	159,936 Kb	355,176 Kb

Table 2
The result of ontology generation.

Parameter	DB1-ST1	DB1-ST2	DB2-ST1	DB2-ST2
Ontology compilation time, s	2	36	3	93
Ontology saving time, s	< 1	21	< 1	48
The size of Ontology (OWL XML format), Kb	114	212,687	204	569,827
Number of classes	13	2	13	15
The number of individuals	2	651,883	2	2,126,054
Number of data properties	180	2,065,082	784	5,645,010
Number of object properties	146	2,045,211	544	1,785,652

The characteristics of relational storages are presented in table 1.

After the initial data received, the schemas $IS1$ and $IS2$ are formed and based on $DB1$ and $DB2$ respectively. Their serialized sizes are 35 Kb and 86 Kb respectively.

The $ST1$ and $ST2$ strategies were used to generate ontologies and the results are in table 2.

As it is seen, the size of the compiled ontology many times more than data in DBMS format (varying from 1.33 to 1.6). Firstly, it depends on the OWL XML format that keeps each of the axioms as an element of XML language and stores parameters as nested elements. However, other formats were not capable to minimize the size of the final ontology.

Obviously, the $ST2$ strategy (assumes generation of entity classes and individual records) increases the ontology compilation time more than the $ST1$. It is due to the fact that data extraction from the relational database requires additional time for setting up a connection and execution of SQL requests. The saving time stays linear to the size of ontology.

7. Conclusion

The proposed data model based on ontologies is useful for software products redesign. The model does not rely on the physical implementation of data storage. Moreover, it owns such properties as unity and flexibility which help to build ontologies on its base.

The described strategies that are used for ontology compilation from a relational database helps to set up this process for systems based on relational data storage. The strategies are both applicable for the implementation of systems' interfaces and for ontology compilation. The ontology can help to find contradictions within schema and data of the database.

References

[1] Guarino N. Formal ontology in information systems: Proceedings of the first international conference (FOIS'98), June 6-8, Trento, Italy. (1998). pp. 3-15.

[2] Simple Object Access Protocol (SOAP) 1.1, W3C Note 08 May 2000, Avaliable at: https://www.w3.org/TR/2000/NOTE-SOAP-20000508/.

[3] Yarushkina N., Moiseev V. Analytical Review of Data Transformation for the Task of Integrating Various Representations on the Example of Ontologies and Relational Databases. CEUR Workshop Proceedings, Vol. 2413 (2019). pp. 191-197.

[4] Bumans G. Mapping between Relational Databases and OWL Ontologies: an Example, Avaliable at: https://www.lu.lv/materiali/apgads/raksti/756_pp_99-117.pdf.

[5] Astrova A., Kalja A. Mapping of SQL Relational Schemata to OWL Ontologies, Avaliable at: http://wseas.us/e-library/conferences/2006elounda1/papers/537-193.pdf.

[6] Freitas, Ricardo André Pereira. "Relational databases digital preservation." (2013), Avaliable at: https://repositorium.sdum.uminho.pt/bitstream/1822/25655/1/Ricardo%20Andr%C3%A9%20Pereira%20Freitas.pdf

[7] .edmx File Overview (Entity Framework), Avaliable at: https://docs.microsoft.com/en-US/previous-versions/dotnet/netframework-4.0/cc982042(v=vs.100), last accessed 2019/04/05.

[8] A Direct Mapping of Relational Data to RDF, http://www.w3.org/TR/rdb-direct-mapping/.

[9] Sequeda1 J., Arenas M., Miranker D.: A Completely Automatic Direct Mapping of Relational Databases to RDF and OWL, https://www.researchgate.net/publication/267232319_A_Completely_Automatic_Direct_Mapping_of_Relational_Databases_to_RDF_and_OWL.

[10] R2RML: RDB to RDF Mapping Language, https://www.w3.org/TR/r2rml/#dfn-r2rml-mapping-document.

[11] Louhdi M., Behja H., Alaoui S.: Transformation Rules For Building OWL Ontologies from Relational Databases, https://airccj.org/CSCP/vol3/csit3822.pdf.