

Ontology Generator from Relational Database Based on Jena

Shufeng Zhou (Corresponding author)

College of Mathematics Science, Liaocheng University
No.34 Wenhua Road, Liaocheng, Shandong 252059, China
E-mail: zhousflc@sohu.com

Haiyun Ling

College of Computer Science, Liaocheng University
No.34 Wenhua Road, Liaocheng, Shandong 252059, China

Mei Han

College of Primary Education, Jining University
No.1 Xingtian Road, Xinqu, Qufu, Shandong 273155, China

Huaiwei Zhang

College of Computer Science, Liaocheng University
No.34 Wenhua Road, Liaocheng, Shandong 252059, China

The research is financed by Natural Science Foundation of Liaocheng University (X051034)

Abstract

There are a lot of difficulties in the ontology generation from relational database such as unclear generation approaches, un-unified ontology languages and so on. So in order to provide unified ontology and improve the quality of ontology generation, approach proposed in this paper firstly extracts database metadata information from relational database using reverse engineering technique, and then analyzes the correspondent relationship between relational database and OWL ontology, and presents an ontology generation from relational database. Finally, a prototype tool of the generator, implemented based on Jena in Java development platform, and case study demonstrates the feasibility and effectiveness of the approach.

Keywords: Semantic Web, Ontology, Relational Database, OWL, Jena

1. Introduction

The Semantic Web, unlike the current web, will represent web content that is also machine-readable in order to provide better machine assistance for human users (Note 1). Ontology, an explicit specification of a shared conceptualization, play an important role in creating such machine-readable content by defining shared and common domain concepts (Note 2). And increasingly with the development of Semantic Web research, people have realized that the success of Semantic Web, in most extent, depends on the proliferation of ontology. Particularly, ontologies could resolve semantic heterogeneity by providing a shared comprehension of a given domain of interest. And at the same time people pay more and more attention to the construction of ontologies. However, there is a large quantity of content on web pages still generated from relational database. In particular, as reported in (Chang, He et al. 2004), about 77.3 percent data on the current web are stored in relational database. Therefore, it is important to generate ontology from existing rich legacy data source.

Now, there are a lot of ontology constructions tool can be used, but there some difficulties for domain expert to understand the meaning between these approaches such as unclear generation approach, un-unified ontology language and so on. So, in order to provide unified ontology and improve the quality of ontology, this paper analyze the correspondent relationship between relational database and OWL ontology, presents an ontology generation from relational database(RDB2On) based on Jena. In fact, relational database can be formalized by First Order Logic (Smullyan 1995); while the ontology uses Description Logic (Baader, Calvanese et al. 2007),

which is a subset of First Order Logic. Therefore, it is feasible to generate ontology from relational database.

In this paper, we will present a generator tool which is implemented based on Jena in Java platform. We will describe the design and implementation of the generator in detail. The remainder of this paper is organized as follows. First, the section 2 describes the formal definition of schema. Section 3 enumerates a set of significant rules of transformation from relational database schema to ontology. Second, section 4 and 5 demonstrates the design and implementation of generator and describes the related work respectively. Final, gives the summary of the paper with future work in section 6.

2. Formal Definition

Where, we give simplified definitions of concepts in order to describe the problem conveniently.

2.1 Relational Schema

Definition 1. A relational schema is a tuple $R(U, D, dom, I)$ (Note 3), where

- R Represents the name of a relation. A relation is defined as a set of tuples that have the same attributes, and a tuple usually represents an object and information about that object. At the same time a relation is usually described as a table, which is organized into rows and columns. All the data referenced by an attribute are in the same domain and conform to the same constraints.
- U Represents a finite set of attributes in a relation, such attributes usually are used to describe a entity, such as $\{A_1, A_2, \dots, A_n\}$, i.e., the relational table has n attributes. Where, every A_i is disjointed with A_j when $i \neq j$.
- D Represents a finite set of domain of the attributes. A domain describes the set of possible values for a given attribute. Because a domain constrains the attribute's values and name, it can be considered constraints. Mathematically, attaching a domain to an attribute means that all values for this attribute must be an element of the specified set. Such as $\{D_1, D_2, \dots, D_n\}$, where every D_i denotes a domain and has a set of associated values.
- dom Represents a function that maps the U to the D , i.e., every attribute A_i in U must have a valid value in D_i .
- I Represents a finite set of integrity constraint, which restricts the data instances that can be stored in the database. Constraints allow you to further restrict the domain of an attribute, and provide one method of implementing business rules in the database. Constraint also can apply to single attribute, to a tuple or to an entire relational table. There are two principal constraints named entity integrity and referential integrity.

2.2 OWL Ontology

Definition 2. An ontology is a tuple $O(C, P^C, R, P^R, H)$ (Note 4), where

- O Represents the name of ontology. Ontology represents domain knowledge. An ontology is defined as an explicit specification of a conceptualization. In Semantic Web domain, an ontology is defined as a set of knowledge terms, including the vocabulary, the semantic interconnections, and some simple rules of inference and logic for some particular topic. (Hendler 2001)
- C Represents a finite set of concepts in ontology. Where, a concept also be called a class which defines a group of individuals that belong together because they share some properties. Classes can be organized in a specialization hierarchy. For example, Tom and John are both members of the class Person.
- P^C Represents a finite set of properties of concepts. Properties can be used to state relationships between individuals or from individuals to data values. For example, if $c_i \in C$, then the properties of c_i can be denoted by $P^C(c_i)$. The concepts are different as results of the concepts always have different properties.
- R Represents relationship between concepts in ontology. Where, the concepts always are classes. A relationship usually consists of two parts, one is the domain, and the other is range. And the domain always is a concept, while the range always is either a concept or interval. We can also say that property is a special relationship when the range is interval.
- P^R Represents a finite set of properties of relationship, which restrict the relationship in some extent. For example, we have a relationship named hasAge, and its range is integer domain. Further, we could restrict that its valid values are from one to 100.
- H Represents hierarchy in concepts, properties and classes. Class hierarchies may be created by making one or more statements that a class is a sub-class of another class. For example, the class Person could be

stated to be a subclass of the class Mammal. From this a reasoner can deduce that if an individual is a Person, then it is also a Mammal. Likewise, Property hierarchies may be created by making one or more statements that a property is a sub-property of one or more other properties. For example, C_i is sub-class of C_j , if and only if every instance in C_i is instance in C_j .

3. Rules for Ontology Generation

The generation process is done progressively as following rules. Each relational table is transformed into a class and each column is transformed into a property respectively. In addition, if the relational database table has foreign key references to other tables, these can be transformed to object property. Likewise, if the column value is not nullable, the cardinality of property corresponding is set to one, etc. However, in order to save space we only describe the main rules as follows, and at the same time give the formal representation.

Rule 1. Each table in relational database should be transformed into a class with same name corresponding to the table, and comment of table be transformed into comment of class, respectively.

$$\forall T \in RDB \rightarrow Class(ID(T))$$

Where, RDB represents relational database, ID represents the unique identifier function of table. $Class$ represents a function to create class according to table.

Rule 2. For table in relational database, each column, which is neither primary key nor foreign key, could be transformed into a data-type property with same name corresponding to the column. And the domain is the class created by this table, range the data-type of the column in the database.

$$\forall T \in RDB \wedge F \in Attr(T) \wedge \neg isFKey(F, T) \wedge \neg isPKey(F, T) \rightarrow \\ DatatypeProperty(ID(F), domain(ID(T)), range(datatype(F)))$$

Where, $Attr$ get all columns from table. $isFKey$ and $isPKey$ determine whether a column is a foreign key and primary key respectively. $DatatypeProperty$ is also a function to create datatype property in OWL language.

Rule 3. Each column, which belongs to primary key in table, should be transformed into functional property in OWL.

$$\forall F \in Attr(T) \wedge isPKey(F, T) \rightarrow FunctionProperty(ID(F), domain(ID(T)), range(datatype(F)))$$

Where, $FunctionProperty$ is a function to create object property.

Rule 4. The foreign key in T_1 , which references to primary key of T_2 in database, are transformed into two inverse object-properties. The domain of one object-property is class corresponding to T_1 and the range is class corresponding to T_2 respectively. Those of the other are reverse.

$$\forall T_1, T_2 \in RDB \wedge \forall F \in Attr(T_1) \wedge isFKey(F, T_1) \wedge Referce(F, T_1, T_2) \rightarrow \\ ObjectProperty(ID(F), domain(C_1), range(C_2)), \\ ObjectProperty(invID(F), domain(C_2), range(C_1))$$

Where, $Referce$ is a function that one relational table is related with the other by foreign key.

Then, we give some simple constraints on ordinary columns and foreign keys.

Rule 5. In a relational table, if a column (except foreign key) is declared as NOT NULL, then the cardinality of the property corresponded is set to one.

Rule 6. In a relational table, if a column (except foreign key) is declared as UNIQUE, then the maximal cardinality of the property corresponded is set to one.

Rule 7. In a relational table, if a foreign key is declared as NOT NULL, then the minimal cardinality of the object property corresponded is set to one.

Rule 8. In a relational table, if a foreign key is declared as NULLABLE, then the maximal cardinality of the object property corresponded is set to one.

4. Design and Implementation of the Ontology Generator

The aim of the ontology generator is to make full use of the existing relational database to generate ontology automatically, to save the time of development and to improve the quality of the ontology generation. In fact, there are a lot of explicit and implicit conceptual model and information resource in relational database, so mining and using them as the knowledge base of conceptual design of ontology to enable the ontology generated

to have more rich semantics.

Based on the rules introduced above, an ontology generator from relational database (RDB2On) was developed. The RDB2On can directly extract the relational schema from database, and then translates them into an OWL ontology written in RDF/XML syntax (Note 5). The function modules of the generator consist of three parts.

- Database Analyze Module. In this module, we adopt reverse engineering technique (Roger, Terence et al. 1996) to acquire database metadata from database. Firstly, analyzes the database structure and extracts the metadata about all tables from the relational database. And then acquires the metadata information about all columns from every table. Finally, collects all the information together as the description of entire relational database. Saving space, we show a case in figure 1 below. Using MySQL, we created the database product and described as an example.
- Schema Transform Module. This module transforms the relational database description into OWL ontology according to the rules introduced above. Firstly, according to the metadata of relational database, we transform all tables into classes, and columns into data type property or object property. Secondly, using integrity constraints relationship, we created some cardinality constraints on the columns.
- OWL Ontology Module. This module mainly deals with the final ontology document, including writing the ontology into a document and reading the ontology document to the screen for end user. For convenience, we write the document into RDF/XML syntax (see figure 2). At the same, user can also change to other style according to their demand.

The generator is implemented based on Jena2.6.0 (Note 6) in Java1.6 (Note 7) development platform. Jena is open source and grown out of work with the HP Labs Semantic Web Programme (Note 8). At the same time Jena is a Java framework for building Semantic Web applications. It provides a programmatic environment for OWL. In the prototype tool, we use DatabaseAnalyser Java class to extract the database metadata about all tables in database and use them as input information to our translation algorithm, and then an standard OWL ontology document is automatically generated from the original database. We use MySQL5.0.28 (Note 9) databases as the original database, because they provide specific views about the database metadata.

5. Related Work

In the literature there are several approaches for addressing ontology construction from relational database. Volz et al. in (Stojanovic, Stojanovic et al. 2002) propose an approach based on semi-automatic generation of a F-logic ontology from a relational database model. The ontology generation process takes in account different types of relationship between database tables and maps them to suitable relations in the ontology. The process is not completely automatic and a user intervention is needed when several rules could be applied to choose the most suitable. Relational.OWL (de Laborda and Conrad 2005) is an OWL ontology representing abstract schema components of relational databases. Based on this ontology, the schema of any relational database can be described and in turn be used to represent the data stored in that specific database. However, other approaches, such as D2R (Bizer 2003), use a declarative, XML-based language to describe mappings between relational database models and ontologies implemented in RDF Schema. In D2R, basic concept mappings are defined using class maps that assign ontology concepts to database sets. The class map is also the container of a set of attribute and relation mapping elements called bridges.

Our approach generates ontology from relational database directly and automatically. At the same time, we consider that user intervention may be needed later to refine the generated ontology with the help of domain experts. Subsequently, we could get high quality ontology to provide better semantics for local database source in special domain. However, the ontology, generated from relational database using our generator, could be viewed as the local data source ontology without any instances, and several local ontologies could be integrated into a global ontology in the future for the entire system. For global ontology, the advantage of wrapping each information source to a local ontology is to allow the development of source ontology independently of other sources or ontologies. So we believe that this approach is more effective than a massive dump.

6. Summary and Future Work

In this paper, we present an ontology generator tool implemented based on Jena2.6.0 in Java1.6 platform. We use MySQL as the original database, because they provide specific views about the database metadata. The main contributions of this paper are listed as follows. First, we describe a new approach which can provide unified ontology and improve the quality of ontology. Second, this approach can mine the implicit conceptual relationship in relational database, i.e., hierarchy relationship, concept constraints, property constraints and so on. Final, the ontology generated using our prototype tool could be integrated into a global ontology in the future. In

addition, case study demonstrates that this approach is feasible and effective.

References

- Baader, F., D. Calvanese, et al. (2007). The description logic handbook, Cambridge University Press New York, NY, USA.
- Bizer, C. (2003). D2R MAP-A database to RDF mapping language. The 12th International World Wide Web Conference(WWW2003).
- Chang, K. C. C., B. He, et al. (2004). Structured databases on the web: Observations and implications, ACM New York, NY, USA. 33: 61-70.
- de Laborda, C. P. and S. Conrad. (2005). Relational. OWL: a data and schema representation format based on OWL, Australian Computer Society, Inc. Darlinghurst, Australia, Australia: 89-96.
- Hendler, J. (2001). Agents and the semantic web. IEEE INTELLIGENT SYSTEMS AND THEIR APPLICATIONS, IEEE COMPUTER SOCIETY. 16: 30-37.
- Roger, H. L. C., M. B. Terence, et al. (1996). A Framework for the Design and Evaluation of Reverse Engineering Methods for Relational Databases, Elsevier Science Publishers B. V. 21: 57-77.
- Smullyan, R. M. (1995). First-order logic, Dover Publications.
- Stojanovic, L., N. Stojanovic, et al. (2002). Migrating data-intensive web sites into the semantic web. New York, NY, USA, ACM Symposium on Applied Computing (SAC2002). Madrid, Spain: 1100-1107.

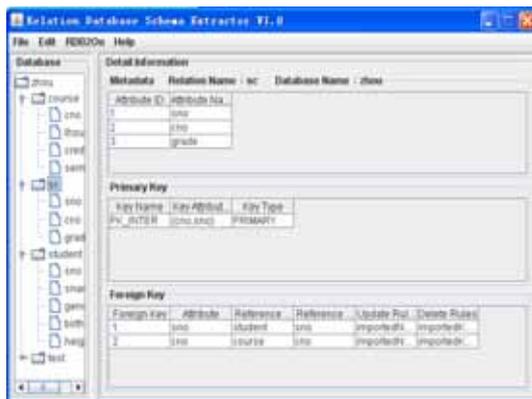


Figure 1. Relational Database Schema



Figure 2. Ontology from Relational Database

Notes

- Note 1: <http://www.w3.org/2001/sw/>
- Note 2: <http://www.w3.org/TR/owl-features/>
- Note 3: <http://en.wikipedia.org/wiki/>
- Note 4: <http://www.w3.org/TR/owl-features/>
- Note 5: <http://www.w3.org/TR/owl-xmlsyntax/>
- Note 6: <http://jena.sourceforge.net/>
- Note 7: <http://java.sun.com/>
- Note 8: <http://www.hpl.hp.com/semweb/>
- Note 9: <http://www.mysql.com/>