RIGA TECHNICAL UNIVERSITY

Information Technology and Management Science

ISSN 2255-9094 (online)
ISSN 2255-9086 (print)
December 2018, vol. 21, pp. 81–85
doi: 10.7250/itms-2018-0013
https://itms-journals.rtu.lv

SQL Query Construction from Ontology Concept Descriptions

Henrihs Gorskis
Riga Technical University, Riga, Latvia

***Abstract* – Based on the usage of previously proposed database concepts as mapping point to a database in a domain ontology, the present paper describes the process of constructing SQL queries from them. The proposed database concepts allow for the mapping of domain concept to the source of data from a database. The paper describes the process of traversing the class hierarchy in an ontology for gathering these database concepts and constructing the SQL query. The purpose of the constructed SQL query is to obtain data from a database to populate the ontology with instances related to a selected ontology concept. The described process begins with the selection of one ontology concept, obtaining all directly related concepts, filtering and collecting database concepts, and finally constructing the SQL query.**

***Keywords* – Data access, database mapping, ontology.**

I. Introduction

The present paper is a continuation of the research on database related concepts in domain ontologies. In the previous research, these concepts have been proposed as mapping points for data access and retrieval. The previous paper described the nature of the proposed database concepts [1], the philosophy behind them and how to identify and differentiate between domain concepts and database concepts within an ontology. The mapping of said concepts and their use for the generation of SQL queries, described in the present paper, are required for the development of an ontology-based data retrieval system. This is done in the context of medical data to provide data access to medical personal [2]. The purpose of the system is to allow medical staff to use medical concepts as descriptions for required data to obtain said data from a connected database. The system would find all relations to database concepts from these medical domain concepts within a common ontology and generate SQL queries in place of the medical personal. This way, a system employing the techniques described in these papers would position itself as a semantic layer between the user and a database. Since ontologies are not meant to be used as databases themselves [3] and databases have limited semantic capabilities, a system, which generates queries instead of the user allows for data storage and data retrieval to be performed by separate technological solutions. The present paper describes the process of using these database concepts to generate valid SQL queries. Database concepts are used to provide ontologically sound information about data sources. Using ontology reasoning these sources can be derived and used to populate the ontology with data from the sources. This approach of using concepts as mapping points and ontological reasoning means that the description of how to obtain data for the ontology is part of the ontology itself and only uses functionality provided by the ontology. This approach allows for a more transparent and intuitive description of ontology-database mapping, where domain concepts are defined as extensions of basic database concepts within the same ontology description. The previous paper [1] concentrated on the database concepts themselves, whereas this paper provides a more specific description of the process of gathering the database concepts and generating a valid SQL query from them.

The described approach provides a description of information retrieval using ontologies [4] that is different from all existing approaches by using database concepts, which are part of the same ontology as other domain concepts, generates SQL queries from traversing the ontology and retrieves data from a connected database. This is done to provide a semantic layer while allowing more direct access to the current data in a database.

II. Database Concepts

Database concepts are named concepts, which are identifiable as describing a database object. They are basic concepts providing only a unique internationalized identifier IRI. The IRI of these concepts allows for the identification of a database object. All database objects use a specific prefix, which allows them to be identified as a database object. The name of the concepts must be equivalent to the name of the database object they are mapped to. These concepts are used to simplify the problem of modelling complex domains [5] by adding additional information to ontology concepts and use this information to extend what an ontology can be used for. The added information is intended to be extracted at a later stage.

There are three types of the proposed database concepts used as mapping point within a domain describing ontology: class concepts, object property concepts, and data property concepts [1]. Class type database concepts are used to point to tables and views. Database object property concepts are used to point to relations between different tables. Data property concepts point to table attributes (table columns). Using these three types of database concepts, enough mapping information is supplied to the ontology to make SQL query construction possible. This approach also provides a means of creating a meta-model, which reuses existing domain concepts [6]. For database concepts to be effective and to fulfil their role, they must be added to the domain ontology by an expert who knows the structure of the dataset. Database concepts that are not connected to domain concepts will not be able to provide data for them. Since the purpose of a domain ontology is to provide complex descriptions of these concepts, important functionality

may be skipped. Only if new data from an external source are added to more complex domain concepts, the full potential of the ontology will be utilized.

The process of translating some of the objects found in the database can also be used to create a basic set of fundamental concepts for an ontology. This basic set of concepts can form the beginning of a new ontology. The creation of a new ontology can be necessary when no existing ontology is satisfying the needs of the current data access task. Thereby, the proposed method does not only provide mapping to a database but also falls in the category of ontology learning [7]. However, it is not the main purpose of this approach to create a new ontology. Domain concepts must be presented to allow for semantic data access. Without domain concepts known to the user, only database objects described as domain concepts are available, which may serve only as a weak semantic layer. The database concepts extend definitions of domain concepts within the ontology [2]. An existing domain ontology can be extended with database concepts by adding the database concepts to the descriptions of already used domain classes, object properties or data properties. Such a domain ontology extended with a database concept contains enough mapping information for the process of generating SQL queries and may provide enough semantics for the user.

III. Constructing SQL

The process of constructing the SQL query for the retrieval of data begins with the selection of a concept from the ontology. The purpose of ontology-based data access is to allow the user to work with familiar concepts and still achieve the results of a data query [4], [8]. The concepts with which the user works can be of any type – domain or database concept. By selecting a concept from the ontology, the user expects to obtain information that is related to and encompassed by the selected concept. This is achieved by using the mapping information combined with ontology reasoning [9] to obtain the required queries for data extraction. The selection of concepts replaces the process of writing data selecting queries by the user. Since all concepts should be in some way connected to database concepts, which allow for the extraction of data, and domain concept should be more familiar to the user compared to database concept, usually a domain concept should be selected at this stage. By selecting familiar domain concepts and trusting in the definitions provided in the ontology as well as the database mapping, data extraction from the database should become simplified.

The database mapping information that is provided in the ontology allows for the construction of SQL queries as well as provides data to the user. After the user selects a concept for which he desires to obtain data from the database, this selected concept becomes the base for all further SQL generation steps. The class hierarchy of the ontology is traversed to find all classes that are related to the selected concept. Related concepts have some subclass-superclass relationship. This relationship may be direct, or indirect with some other classes in-between. The related classes that are identified to be database concepts are gathered and retained. Depending on the type of class, these concepts provide different information for SQL construction. The found definitions provide information for template SQL queries. Some classes will point to tables in the database by their name only. Other, more complex, classes may use database fields and values to indicate how records related to the desired class can be obtained.

A. Base Concept Selection

The selection of the concept for which data are to be gathered and extracted from the database is the first step in the creation of the SQL query. In case of a named concept, the IRI of the concept can be provided. Alternatively, a new concept may be created and added to the ontology. This would be a query concept, which would entail definitions and restrictions not existing in the ontology beforehand. A query concept might not be necessary for the description of the domain and it might be desirable to remove such a concept from the ontology once the required data are retrieved. A means of creating and defining a new concept to serve as a query concept are outside of the scope of this paper. This query concept is in no way different from any other concept within the ontology, except for its purpose. It is meant to describe a new set of properties, which have not yet been defined and are related to a set of data, which can be retrieved from the connected database.

Once this concept is selected or defined, it can be reasoned about in the ontology to gain addition information about it and to generate the SQL query needed to obtain fitting entries from the database.

B. Ontology Traversal

Every class concept in the ontology is part of the hierarchy of concept in the ontology, defined by the “is-a” relation. Every class concept may be a sub- or super-class of any other concept [10]. Concepts might also not have a clearly defined relation to one another. However, every concept is both a sub- and super-concept to itself, a sub-concept of the top concept “Thing” and a super-concept of the bottom concept “Nothing”.

Before the class hierarchy of the ontology can be traversed, an ontology-reasoner has to establish any implicit relations between the classes from the explicit definitions given by the ontology. Based on definition end properties of classes, some sub-class relationships can be concluded by the reasoner and any such relations should be made known.

The process of traversing the ontology begins at the selected base concept. The process examines the class hierarchy in both directions. Super-classes of the base concept provide known facts (necessary features) about the base concepts. If there are database concepts in the set of super-classes of the base class, information from them can be applied to the query directly. Any restrictions in super-classes, which use database concepts and values, can be used as strict restrictions on the data to be obtained from the database. Since these are necessary properties of the desired concept, any database records that do not comply with these definitions may be ignored. However, concept definitions, fields and values found in sub-classes indicate sufficient but not necessary properties. This means that they only offer weak restrictions. They can be added to the query but should not restrict the amount of data retrieved from the

database. While super-class restrictions may be used to lower the amount of data retrieved from the database, sub-classes should not. Should data be found that have these sufficient properties, they are known to belong to the result set and belong to the query concept.

C. Gathering Database Concepts

Every time a class is encountered in the set of super-classes, it is checked to be a database concept, if it is it is remembered. In the case of a named class, the table or view is noted. In the case of a complex class, which describes a property or feature of the class and contains a database concept (object property or data property), the property concepts are noted. Any values that are indicated directly in these definitions are also noted and used to create restrictions in the query.

During the process of traversing the database, any encountered concepts are checked. In the case of a named concept, the IRI of this named concept provides a clue towards the type of the concept. The prefix of the IRI indicates whether it is a database or a domain concept. In the case of a complex concept, the property and object must also be checked to see if they are database concepts or values.

Once all related database concepts are found and retained, they may be summarized, and the SQL query can be built. Each of the concepts is used to provide parts of the final query.

D. Building SQL Queries

Based on the used mapping approach, all SQL queries, which will be built, follow a standard structure and will not require any exotic features of the SQL language. Since mapping is realized using specific pointers, the generated queries will simply extract the maximum necessary data to determine all individuals belonging to be selected query concept. At the same time, any restrictions found in the concept definitions will be used to restrict a lower number of database records. All queries are of the following type:

```
SELECT <neccessary fields> FROM <found tables> WHERE <found restrictions>
```

This template is used to select the necessary data. The fields, tables and restrictions are found by analyzing the class hierarchy of the selected query concept. Every part of this template is expanded and filled out by the SQL query generation process. The generator may be simpler and use only a simple template. More advanced queries may also be generated, combining multiple tables in different ways, based on the positions of concepts within the class hierarchy. In the case of tables found in the set of super-classes, they may be added in the form of an SQL inner join. Tables found in the set of sub-classes may be added in the form of an SQL left join. The same is applicable to tables and database features found in properties related to the query concept.

Simpler queries will return more data, since they must obtain multiple simple sets from all possible related tables. This means that the ontology reasoner will have to do more work, while more complex queries can return fewer database entries, thereby simplifying the ontology reasoning process.

E. Query Execution and Data Management

Once the query building process is finished, it is executed. The result is a dataset, which fits the restrictions found in the class hierarchy. Therefore, this dataset will contain all individuals (records), which are possible candidates for the selected query concept. Based on the quality of the description of the concept, this result may not be specific enough [10], [11]. Some discrepancies are possible at this stage. The quality of the extracted data is reliant on the descriptions of the domain and database concepts. A weakly restricted ontology may result in too many records being returned as fitting the given restrictions. At some point, it may be necessary to update the ontology to reflect changes in a way information is stored or described in the domain. Some defaming approaches may be used for this task [12]. Alternatively, the domain expert and database administrator must update the descriptions provided in the ontology manually.

All returned database records are converted into individuals and added to the ontology. Once they are added, the ontology reasoner is employed one more time. At this stage, the ontology reasoner inspects all added individuals and tests whether they are fit to be classified as belonging to the query concept. This is necessary since not all aspects of ontology reasoning are convertible to the SQL query. All individuals who are classified as belonging to the selected query concept are returned as a result of the data retrieval process.

IV. Query Example

Let us assume that a medical staff member requires a list of all current positive blood samples registered in the database. This user does not possess the necessary knowledge about SQL and the structure of the database to write a query themselves. However, an ontology exists describing relevant domain concepts and has database concepts as mapping points. The relevant part of the ontology containing concepts related to blood tests is shown in Fig. 1. The user selects the concept named “med:Positive_blood_sample”. For the purpose of this example, shortened names are used instead of full IRI. The user selects the concept using an interface, which replaces concept IRI with more elegant labels. The process of the user selecting the concepts is not shown in this example, since the interface of the system is outside of the scope of this paper.

In this example, the “med” part in the shortened names points to the prefix of the IRI. There are two prefixes in this example: “med” and “db”. Concepts having the “med” prefix are concepts from the medical domain. Concepts with the “db” prefix are database concepts. Any nameless complex concepts containing a reference to a database concept are also handled as database concepts.

Figure 1 shows all relations the class concepts have. Arrows are “is-a” type relations between concepts. Black arrows are defined relations. Green arrows are inferred relations. By traversing the class hierarchy, all related database concepts are found. In this case, the concept “med:Positive_blood_sample”

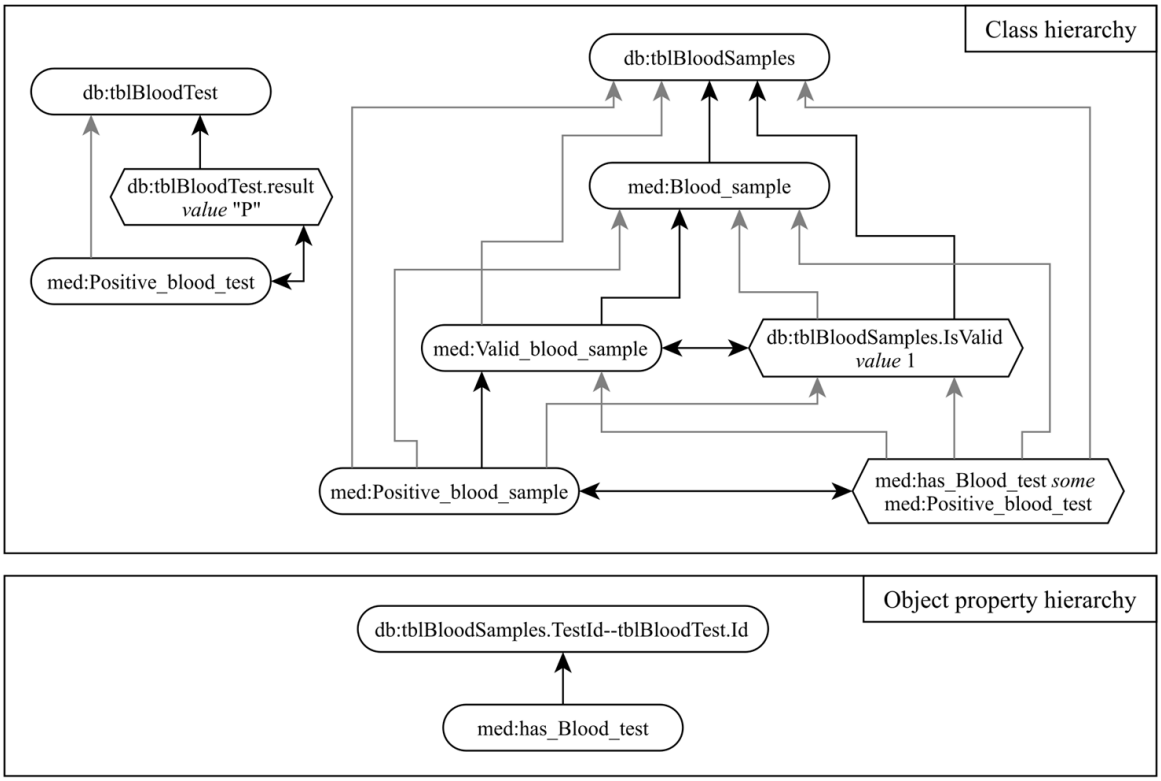

Fig. 1. Class hierarchy and object property hierarchy excerpts from the example ontology.

is directly related to the database concepts "db:tblBloodSamples" and "db:tblBloodSamples.IsValid value 1". However, the concept "med:has_Blood_test some med:Positive_blood_test" references an object property, which in turn is a sub-object property of a database relation. Therefore, it must be further reviewed. This concept connects positive blood samples with blood tests using a concept describing a database relation. This means that a positive blood test must also be obtained from the database. This further reveals the database concept "db:tblBloodTest.result value "P"". All found database concept are as follows:

- "db:tblBloodSamples";
- "db:tblBloodSamples.IsValid value 1";
- "db:tblBloodSamples.TestId--tblBloodTest.Id";
- "db:tblBloodTest.result value "P"";
- "db:tblBloodTest".

The found concepts can now be applied within a simple query template. The first concept points to the database table "tblBloodSamples", to which the query concept is directly related. This establishes that all fields from this table will be necessary as a result of the query. This also adds the table name to the FROM part of the query.

The second database concept restricts the value of a table field. This restriction will be added to the WHERE part of the query template. Since this concept references a table, which is already part of the query, this table will not be added for a second time. The third database concept establishes a required relation to another table record. It references another table with the name "tblBloodTest". It also provides the necessary restriction for the WHERE part of the query to select only connected records. The fourth concept provides a restriction for the second table. The final concept is not necessary since this table has already been added to the query.

Combining all these parts results in the query:

```
SELECT tblBloodSamples.* FROM
tblBloodSamples, tblBloodTest WHERE
tblBloodSamples.IsValid = 1 AND
tblBloodSamples.TestId = tblBloodTest.Id
AND tblBloodTest.result = 'P'
```

The resulting SQL query is a valid query and will return all database records required for further examination using ontology reasoning. Since ontology reasoning may be complex and difficult to translate directly to a SQL query, it is necessary to insert all obtained database records into the ontology as individuals for further classification. The process of classification in the ontology may find all classes which the obtained individuals will be examples of.

V. Ontology Population

Once the SQL query has been generated, it is possible to execute it and obtain data. All found records are added to the ontology. For each record a new individual is created. The new individuals are provided with a generated name. Individuals are concepted using the object properties found in the query concept description. For all values returned by the database,

applicable data properties are used to concept individuals to the data values. In the case of database table columns, which are not defined in the ontology, new data properties are created using a generated name consisting of the table and column name provided by the database.

Data are added to the ontology to perform additional reasoning and conclude all possible classes for these new individuals. This is necessary for cases when ontology reasoning cannot be directly stated in a query restriction. Additionally, adding data to the ontology can provide additional information to the user in form of relevant classification of data and individuals.

VI. Conclusion

The present paper has described the process of using the previously proposed database concepts for generating database queries using SQL. As shown, it is possible to generate valid SQL queries using these database concepts and obtain data relevant to a selected query concept. This is done by defining a new or selecting an existing concept as a query concept. By traversing the class hierarchy from this query concept, all necessary information for SQL query generation can be obtained. By using this method, it is possible to provide ontological reasoning to the process of data extraction from a database. A simple selection of concepts allows for the dynamic creation of queries within an ontology-based data access system, simplifying the process and making data retrieval from databases accessible to non-experts. A non-expert only needs to select a set of known domain concepts to create a new definition, which can be mapped to data from the database using its relations to database concepts in the class hierarchy of the ontology.

The use of database concepts provides a simple, non-intrusive and transparent way of creating a database to ontology mapping using tools and functions provided by the ontology itself. No additional mapping tools or documents are needed to describe the mapping to database tables, columns and relations. Mapping is defined within the class hierarchy itself. Even though an external tool is required for the selection of database concepts and the parsing of the concepts contents, the ontology remains a valid domain description with some additions.

The process of creating SQL queries is reliant only on the ontology itself and reasoning provided by ontology reasoners. By analyzing all related classes and parsing the names of database concepts, an SQL query template is filled out and executed.

The proposed approach has some limitations. It is only possible to obtain data from a database that follows basic patterns. No complex queries are possible with the proposed mapping solution. The proposed method of generating queries is reliant on ontology reasoning. By shifting to ontology reasoning instead of SQL queries, some new problems may arise for the data user. In cases of some complex domain concept, the result of the query may be different from the expected result. This can happen when the data user is no familiar with ontological reasoning.

Another shortcoming of this approach is the loss of query like functionality. The purpose of this ontology-based approach is to hide database structures and values behind common and understandable concepts. By replacing queries with concept definitions, the selection of fields, aggregate functions and other functionality, which is available in queries, has not yet been considered. This is a direction for further investigation, since such functionality is often needed.

The present paper shows that it is possible to generate valid SQL from information placed in an ontology and that the information placed in the ontology, using standard ontology elements, is sufficient to do so. Such an ontology can be used to hide query specific information from a user who is unable to write such query himself but requires current information from the database, which is unavailable in a standardized report.

References

[1] H. Gorskis, L. Aleksejeva, and I. Poļaka, "Database Concepts in a Domain Ontology," *Information Technology and Management Science*, vol. 20, no. 1, 2017, pp. 69–73. https://doi.org/10.1515/itms-2017-0012

[2] H. Gorskis, L. Aleksejeva, and I. Poļaka, "Ontology-Based System Development for Medical Database Access." *Environment. Technology. Resources: Proceedings of the 11th International Scientific and Practical Conference*. vol. 2, 2017, pp. 24–29. https://doi.org/10.17770/etr2017vol2.2572

[3] M. Sir, Z. Bradac, P. Fiedler, "Ontology versus Database," *IFAC-PapersOnLine*, vol. 48, no. 4, 2015, pp. 220–225. https://doi.org/10.1016/j.ifacol.2015.07.036

[4] K. Munir, M. S. Anjum, "The use of ontologies for effective knowledge modelling and information retrieval," *Applied Computing and Informatics*, vol. 14, no. 2, 2018, pp. 116–126. https://doi.org/10.1016/j.aci.2017.07.003

[5] A. T. Elve, H. A Preisig, "From Ontology to Executable Program Code," *Computers & Chemical Engineering*, 2018, accepted manuscript. https://doi.org/10.1016/j.compchemeng.2018.09.004

[6] Y. Biletskiy, J. A. Brown, G. R. Ranganathan, E. Bagheri, I. Akbari, "Building a business domain meta-ontology for information pre-processing," *Information Processing Letters*, vol. 138, 2018, pp. 81–88. https://doi.org/10.1016/j.ipl.2018.06.009

[7] A. Konys, "Knowledge systematization for ontology learning methods," *Procedia Computer Science*, vol. 126, 2018, pp. 2194–2207. https://doi.org/10.1016/j.procs.2018.07.229

[8] R. Kontchakov, M. Rodríguez-Muro, M. Zakharyaschev, "Ontology-Based Data Access with Databases: A Short Course." In Reasoning Web. Semantic Technologies for Intelligent Data Access. Reasoning Web 2013. Lecture Notes in Computer Science, vol. 8067. https://doi.org/10.1007/978-3-642-39784-4_5

[9] G. Santipantakis, K. Kotis, G. A. Vouros, "OBDAIR: Ontology-Based Distributed framework for Accessing, Integrating and Reasoning with data in disparate data sources," *Expert Systems with Applications*, vol. 90, 2017, pp. 464–483. https://doi.org/10.1016/j.eswa.2017.08.031

[10] M. Benedikt, B. Cuenca Grau, E. V. Kostylev, "Logical foundations of information disclosure in ontology-based data integration," *Artificial Intelligence*, vol. 262, 2018, pp. 52–95. https://doi.org/10.1016/j.artint.2018.06.002

[11] D. Lembo, M. Lenzerini, R. Rosati, M. Ruzzi, D. F. Savo, "Inconsistency-tolerant query answering in ontology-based data access," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 33, 2015, pp. 3–29. https://doi.org/10.1016/j.websem.2015.04.002

[12] S. D. Cardoso, C. Pruski, M. Da Silveira, "Supporting biomedical ontology evolution by identifying outdated concepts and the required type of change," *Journal of Biomedical Informatics*, vol. 87, 2018, pp. 1–11. https://doi.org/10.1016/j.jbi.2018.08.013

Henrihs Gorskis is a Researcher in the field of Information Technology at Riga Technical University (RTU). He defended his *Dr. sc. ing.* degree in 2018. His research interests include data mining, ontology engineering, ontology-based database access and evolutionary computing and programming. He is especially fond of the Java programming language and uses it for both work and personal application development.
E-mail: henrihs.gorskis@rtu.lv
ORCID iD: http://orcid.org/0000-0002-7297-8054