SQL query construction from database concepts

Henrihs Gorskis
Department of Modelling and Simulation
Riga Technical University
Riga, Latvia
henrihs.gorskis@rtu.lv

***Abstract*—The paper extends on previously proposed database concepts and their use as mapping point to a database in a domain ontology. It describes the process of constructing SQL queries from them. The proposed database concepts allow for the mapping of domain concept to the sources of data from a database. The paper describes the process of traversing the class hierarchy in an ontology. This is done to gather database concepts and to construct a SQL query. The purpose of the constructed SQL query is to obtain data from a database and to populate the ontology. It is populated with instances related to the selected ontology concept. The described process begins with the selection of one ontology concept, continues with obtaining all directly related concepts from the ontology, filtering and collecting database concepts, and y finishes with constructing the SQL query.**

Keywords—Ontology, database, data access

I. INTRODUCTION

The paper is a continuation of the work on database related concepts in domain ontologies and their use as mapping points for data access. 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. This is done in the context of medical data to provide data access to medical personal [2]. The previous paper together with this paper are parts of a larger project aimed at implementing an ontology-based database access tools, which serves as a semantic layer and is usable for medical personal. The 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. The approach allows for a more transparent and intuitive description of ontology-database mapping. 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.

II. DATABASE CONCEPTS

Database concepts are named concepts, which are identifiable as describing a database object. They are basic concepts only providing names as unique internationalized identifiers (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 database object. The name of the concepts must be equivalent to the name of the database object they are mapped to.

There are three types of 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. For database concepts to be effective and fulfil their role, they must be added to the domain ontology by an expert who knows the structure of the dataset. Database concepts which 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 the more complex domain concepts, will the full potential of the ontology be utilized.

The database concepts are used to 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 database concept contains sufficient mapping information for the process of generating SQL queries.

III. CONSTRUCTING SQL

The process of constructing the SQL query for the extraction 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 [3]. 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 [4] 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 the extraction of data, and domain concept should be more familiar to the user compared to database concept, a domain concept should be usually selected at his stage. Selecting familiar domain concepts and trusting in the definitions provided in the ontology as well as the database mapping, should simplify data extraction from the database.

The database mapping information, which is provided in the ontology allows the constructions of SQL queries and

978-1-7281-0098-2/18/$31.00 ©2018 IEEE

provides data to the user. After the user selected 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, which are related to the selected concept. The related classes 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.

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 a 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, so it might be desirable to remove such a concept from the ontology once the required data are retrieved.

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 superclass of any other concept [5]. 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. Based on definitions end properties of the 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.

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 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.

During the process of traversing the database, any encountered concepts are checked. In 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 domain concept. In case of a complex concept the property and object must also be checked to see if they are database concepts.

Once all related database concepts are found and retained, they may be summarized and the SQL query may

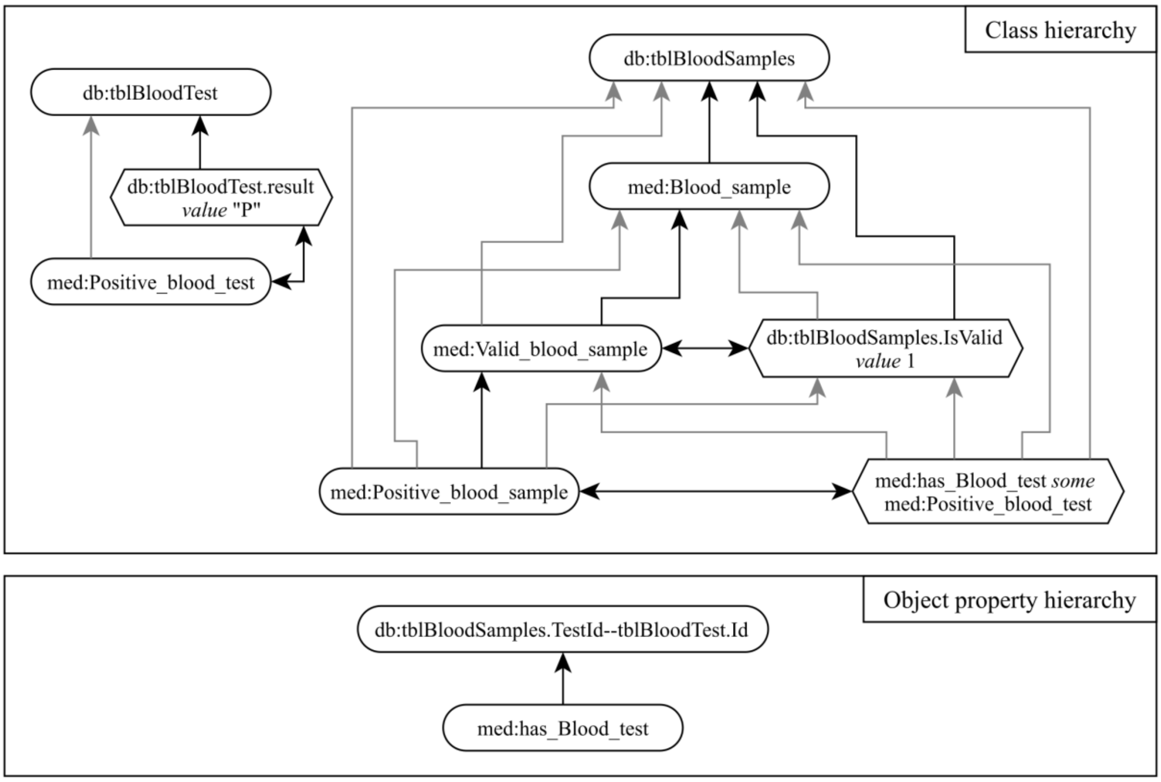

Fig. 1. Class hierarchy and object property hierarchy excerpts from the example ontology

be built. Each of the concepts is used to provide parts of the query.

D. Building SQL queries

Based on the used mapping approach, all SQL queries which will be build 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 the selected query concept. 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.

E. Query execution and data management

Once the query is built 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 [5, 6]. Some discrepancies are possible at this stage. The quality of the extracted data is reliant on the descriptions of the domain and database concepts.

IV. Query Example

Let us assume 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 him/herself. 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 figure 1. The user selects the concept named “med:Positive_blood_sample”. For the purposes of this example, shortened names are used instead of full IRI. The user selected the concept using an interface, which replaces concept IRI with more elegant labels. 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. Arrow 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” 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 its turn is a sub object property of a database relation. Therefore, it too must be further reviewed. This concept connects positive blood samples with blood tests using concept describing a database relation. This means that positive blood test must also be obtained from the database. This further reveals the database concept “db:tblBloodTest.result value "P"”. All found database concepts 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 point to the database table “tblBloodSamples” to which the query concept is directly related to. This establishes that all fields from this table will be necessary in the 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 a second time. The third database concept establishes a required relation to another table’s records. 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 as follows:

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

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 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 the cases when ontology reasoning cannot be directly stated in a queries restriction. Furthermore, adding data to the ontology can provide additional information to the user in the form of relevant classifications of data and individuals.

VI. Conclusions

The paper described the process of using previously proposed database concepts for generating database queries. As was shown, it is possible to generate valid SQL queries using these database concepts and obtain data relevant to a selected query concept. By using this method, ontological reasoning to the process of data extraction from a database can be provided. A simple selection of concepts allows for the dynamic creation of queries, simplifying the process and making data retrieval from databases accessible to non-experts.

The use of database concepts provides a simple, non-intrusive and transparent way to create a database to ontology mapping using tools and functions provided by the ontology 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, which 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 concepts, 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.

References

[1] H. Gorskis, L. Aleksejeva, and I. Poļaka, "Database Concepts in a Domain Ontology," Information Technology and Management Science, vol. 20(1), 2017, pp. 69–73.

[2] H. Gorskis, L. Aleksejeva, and I. Poļaka, "Ontology-Based System Development for Medical Database Access." In Environment. Technology. Resources : Proceedings of the 11th International Scientific and Practical Conference. Vol.2, 2017, pp.24-29.

[3] 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

[4] 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.

[5] 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.

[6] D. Lembo, M. Lenzerini, R. Rosati, M. Ruzzi, D. Fabio 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.