SQuaRE: A Visual Support for OBDA Approach

Michał Blinkiewicz and Jarosław Bąk

Institute of Control and Information Engineering,
Poznan University of Technology,
Piotrowo 3a, 60-965 Poznan, Poland
firstname.lastname@put.poznan.pl

Abstract. We present SQuaRE, the SPARQL Queries and R2RML mappings Environment which provides a visual approach for creating R2RML mappings which can be immediately tested by executing SPARQL queries. SQuaRE is a web-based tool with easy to use visual interface that can be applied in the ontology-based data access applications. We describe SQuaRE's main features, its architecture as well as implementation details. We present an example use case to indicate how SQuaRE can be useful in an OBDA-based scenario.

1 Introduction

Ontologies, as a way of expressing knowledge, are becoming more and more popular in various research and practical fields. They allow definition of a knowledge base using abstract concepts, properties and relations between them. A properly created ontology can be then automatically processed to obtain new inferences and, as a result, a newer version of the knowledge base. Ontologies can be expressed in the Web Ontology Language 2 (OWL 2) [11]. This is a well-known format of ontologies and widely used. Ontologies require data to be in a format of RDF[1] triples. Then, using an appropriate reasoner we can obtain new data in the same format. Moreover, we can query such RDF data using SPARQL[2] queries. In such a way we are adding semantics to data and as a consequence, may obtain semantic results by executing SPARQL queries.

Nevertheless, ontologies and data need to follow the RDF-based representation. Since most of data are stored in different formats, any application of an OWL/OWL2 ontology rises the integration problem between an ontology and stored data. In this case we can transfer our current data format into RDF-based representation and change our software and architecture environment or we can create mappings between the ontology and our data and use an appropriate tool that handles such a solution. The first option is very cost-expensive and needs a lot of changes in the current software architecture. The second approach is easier and cheaper since minor changes in the software architecture are required. We need to create mappings and then query non-RDF data with SPARQL using

[1] https://www.w3.org/RDF/
[2] https://www.w3.org/TR/sparql11-overview/

ontology, mappings and a tool that enables on-the-fly transfer from non-RDF into RDF data. In this method the most important part is to create appropriate mappings. Currently, a very popular standard for expressing mappings from relational databases to RDF data is W3C's R2RML[3]. The standard allows to use existing relational data in the RDF data model, and then use SPARQL to query such data.

In this paper we provide a detailed description of SQuaRE, the SPARQL Queries and R2RML mappings Environment, which provides a visual editor for creating and managing R2RML mappings as well as for creating and executing SPARQL queries. SQuaRE is a web-based application that simplifies the creation of mappings between a relational database and an ontology. It also enables to test created mappings by defining and executing SPARQL queries in a textual manner.

The remainder of this paper is organized as follows. Firstly, we provide preliminary information, then we describe main features of SQuaRE. Next, we present its architecture as well as implementation details. Then, we provide a simple instruction on how to use SQuaRE by presenting an example use case. Finally, we provide conclusions along with future development plans.

2 Preliminaries

SQuaRE is an OBDA-oriented tool which helps an inexperienced user to create mappings between a relational database and an ontology, and then to test those mappings by creating SPARQL queries. Moreover, the tool can be used to write and execute SPARQL queries in a text-based form whereas results are presented in a graphical way. In this section we present the main overview of OBDA and R2RML.

Ontology-based Data Access (OBDA) is an approach [12] to separate a user from data sources by means of an ontology which can be perceived as a conceptual view of data. Moreover, by using concepts and relations from the ontology one can define a query in a convenient way. In this case the user operates on a different abstract level than data source. As a result the user defines queries using concepts and relations from the domain of interest and creates complex semantic conditions instead of expressing queries in terms of a relational data model. Nevertheless, in order to use OBDA approach with relational data one needs to develop mappings between a relational database and an ontology.

The R2RML recommendation provides a language for expressing mappings from a relational database to RDF datasets. Those mappings allow to view the relational database as a virtual RDF graph. Then, the relational database can be queried using the SPARQL language. Each R2RML mapping is a triples map (an RDF graph) that contains: a logic table (which can be a base table, a view or a valid SQL query); a subject map which defines the subject of all RDF triples that will be generated for a particular logical table row; and a set

[3] `https://www.w3.org/TR/r2rml/`

of predicate-object maps that define the predicates and objects of the generated RDF triples. In order to create R2RML mappings manually, one needs to know about ontologies (OWL/OWL2), RDF, R2RML and SQL at the same time.

SQuaRE overcomes the aforementioned issues. The main goal of the tool is to support creation of R2RML mappings and SPARQL queries in a graphical manner. However, at the current state of development SQuaRE supports a graphical editor for R2RML mappings and a text-based interface for creating and executing SPARQL queries. Nevertheless, results returned by a SPARQL query may be presented as a graph to a user.

3 SQuaRE, the SPARQL Queries and R2RML Mappings Environment

3.1 Features

The SQuaRE environment is aimed at providing easy-to-use functions that will support creation and execution of SPARQL queries as well as creation of R2RML mappings. Moreover, SQuaRE allows for management of queries and mappings. A user can save both: mappings and queries for future reference, execution and management. Currently, the tool supports a graphical interface for the creation of mappings and the text-based creation of SPARQL queries. Moreover, results returned by a SPARQL query can be presented in a graphical way (according to the query type). Currently, SPARQL results of queries DESCRIBE and CONSTRUCT are presented on a graph, ASK queries return 'yes' or 'no' while results of SELECT queries are presented in a table.

Nevertheless, SQuaRE provides the following useful features (appropriate figures are shown in Section 4):

1. Browsing a relational database – a user can choose a data source and browse its schema. In this view the user sees table names, column names as well as data types stored in each column. An example view of a relational database is shown in Figure 2. The figure contains two tables named `emp` and `dept` with four and three columns, respectively.
2. Browsing an OWL ontology – a user sees hierarchies of classes, object properties and datatype properties. The user can browse an ontology (Figure 3) and search for its elements.
3. Browsing mappings – a list of all created mappings is shown to a user. The user can choose a mapping and then edit it in the mapping creation view (shown in Figure 5).
4. Graphical creation of R2RML mappings – in a mapping creation view a user can create R2RML mappings. The user needs to choose tables that are going to be mapped. Then, she/he needs to search for an appropriate classes and properties to create mappings using a graphical interface. An example mapping is shown in Figure 5. Classes are represented with an orange background, datatype properties with a green background and object properties with a blue background.

5. Management of R2RML mappings – each created mapping can be saved for future reference. A user can delete mappings, correct them or generate an R2RML file that contains all or selected mappings.
6. Textual creation of SPARQL queries – current version of SQuaRE provides an option to create SPARQL queries using a text-based interface. A user can write and execute a query. The view of a user interface for creating queries is shown in Figures 6, 7, 8, 9 and 10. Graphical editor for creating queries is one of our future development plans.
7. Management of SPARQL queries – each constructed SPARQL query can be saved and used in future. A user can execute, delete or export a SPARQL query. Moreover, the user can select few queries (or all of them) and generate a separate .txt file that contains their definitions.
8. Execution of SPARQL queries – created SPARQL queries can be executed and results are shown in a form of a table, an RDF graph or answer 'yes' or 'no'.

The aforementioned main list of features provides an intuitive ontology-based access to relational data. Moreover, by exporting functionality (importing features are still in development) a user can use SQuaRE to create mappings and test them by creating SPARQL queries, and then save everything into external files. This allows to import queries and mappings into another tool that supports both SPARQL and R2RML.

3.2 Architecture and Applied Tools

SQuaRE is developed in Java as a web application. The architecture of SQuaRE is presented in Figure 1. The tool consists of modules that provide different functionality.

The SQuaRE architecture is divided into two separate sides – client and server. The client side consists of modules working on a client's machine in a user's web browser and provides a presentation layer.

The main module, from the user's point of view, is Visual R2RML Mapper which provides tools for visual (graph based) mappings of relational database metadata, such as table columns, and user provided ontology entities.

Moreover, there are modules responsible for data source configuration and ontology management. The former allows the user to configure a data source by providing DBMS, host location and port, username and password. The latter provides an interface to import an ontology and browse hierarchies of classes and properties.

The server-side modules consist of Data Source and DBMS Manager which manages the user defined data sources and provides JDBC-based access, Ontology Handler for OWL ontology processing, and SPARQL Query Executor which utilizes already defined mappings and allows to execute SPARQL queries in the context of a relational database.

SQuaRE applies well-known tools to handle OWL ontologies, relational data and SPARQL queries. The main tools that SQuaRE uses are the following:

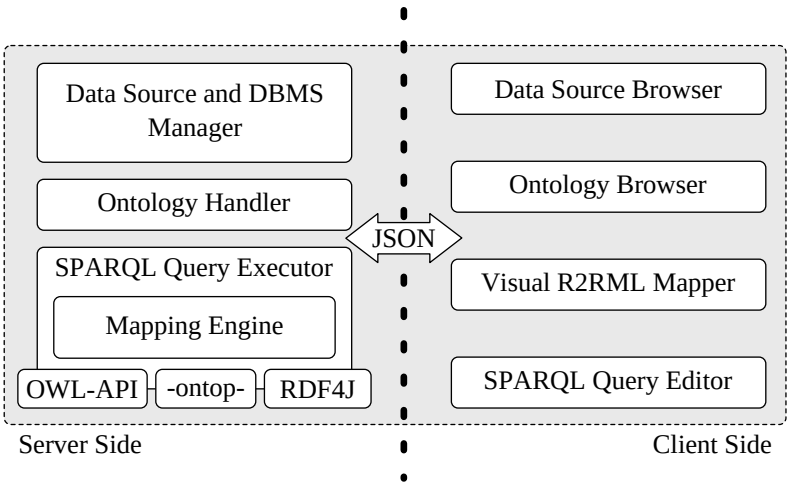

Fig. 1: The architecture of SQuaRE.

– OWL-API [7] – this is a very often used Java library to handle OWL/OWL2 ontologies. It contains a lot of features that are useful when manipulating ontology elements, using reasoner or serialising ontologies. The tool is open source[4].
– The Spring Framework[5] – it is an application framework for the Java platform. Among others, it allows for easy creation of RESTful web services and building backend API. The above mentioned features are heavily used by SQuaRE server side modules interface. In order to simplify application deployment and development Spring Boot module is used. It provides convention-over-configuration solution for stand-alone, production-ready applications. Moreover, Spring Boot embeds Tomcat or Jetty web application server which enables creation of self-contained application.
– -ontop-[6] – it is a platform to query relational databases as virtual RDF graphs using SPARQL. The tool accepts mappings in R2RML and its own OBDA mapping language. SQuaRE uses -ontop- to query relational database using mappings and SPARQL.
– RDF4J[7] – it is a framework for processing RDF data. It enables to parse, store, infer and query of/over such data. The tool supports SPARQL in version 1.1 and is used in many third party storage applications. SQuaRE uses it to save all data connected with created mappings and queries.

[4] `http://owlapi.sourceforge.net/`
[5] `http://projects.spring.io/spring-framework/`
[6] `http://ontop.inf.unibz.it/`
[7] `http://rdf4j.org/`

– Javascript libraries – we use a set of popular Javascript tools such as: AngularJS[8], jQuery[9], Cytoscape.js[10] with CoSE Bilkent layout [6], jsPlumb[11] and jsTree[12] in order to provide visual and interactive functions.

4 Example Use Case

In order to present the usability of SQuaRE we provide an example use case and a simple instruction how to use the tool. We used the W3C's R2RML [4] examples which are easy to understand and show fundamental capabilities of the presented tool.

After creating a new project (here named "W3C (1) Project") user needs to configure a data source providing DBMS, a database location and name, username and password. When all settings are correct the data source view may look like in Figure 2. It consists of two tables named `emp` and `dept` with

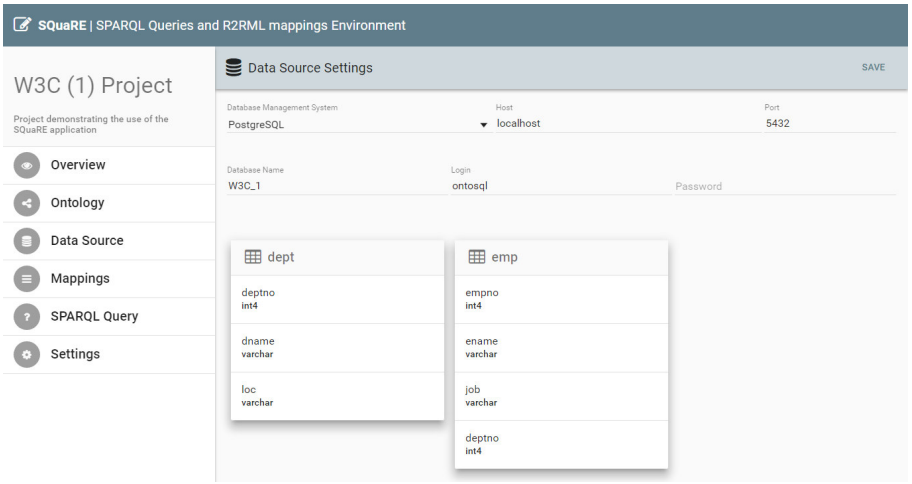

Fig. 2: Data source view of the W3C (1) Project.

information about employees and departments. Each table contains one row which content may be seen in Section 2.1 of [4].

After connecting to a database, the second task is to import OWL domain ontology which then is processed and presented to the user in a form of hierarchies of classes, object type properties and datatype properties (Figure 3).

[8] `https://angularjs.org/`
[9] `https://jquery.com/`
[10] `http://js.cytoscape.org/`
[11] `https://jsplumbtoolkit.com/`
[12] `https://www.jstree.com/`

When data source and ontology are selected the user is able to start creating new mappings. The mapping process is preceded by choosing tables (Figure 4) which will be used during the creation of a particular mapping.

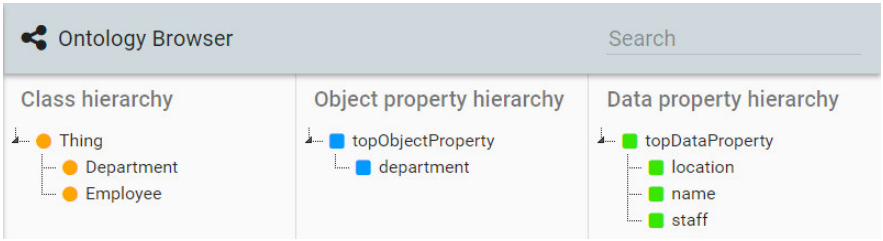

Fig. 3: Ontology view of the W3C (1) Project.

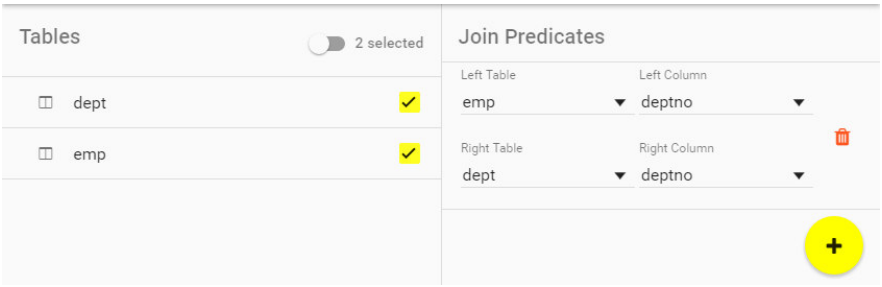

Fig. 4: Tables selection and specification of join predicates.

If the user selects more than one table she/he should define join predicates (Figure 4) which will be used for proper joining records from the selected tables.

The main mapping part consists of selecting classes, object type properties and/or datatype properties from the right-hand side ontology view shown in Figure 5). The user may drag and drop them into the center located canvas. Then she/he may link selected ontology entities with each other as well as with the left-hand side located columns descriptions of particular data source tables.

A complete mapping of W3C's R2RML example (sections 2.1 to 2.5 inclusive of [4]) is presented in Figure 5. Then the last step before the execution of SPARQL queries is to save the mapping.

Based on the aforementioned data source, ontology and mapping the user may begin executing SPARQL queries. SQuaRE is capable of executing all four SPARQL query forms. The `SELECT` form returns variables and their bindings which are presented in a form of a table (shown in Figure 6).

The SPARQL `DESCRIBE` and `CONSTRUCT` forms return RDF graphs which are presented in a form of graphs where orange color indicates objects, grey

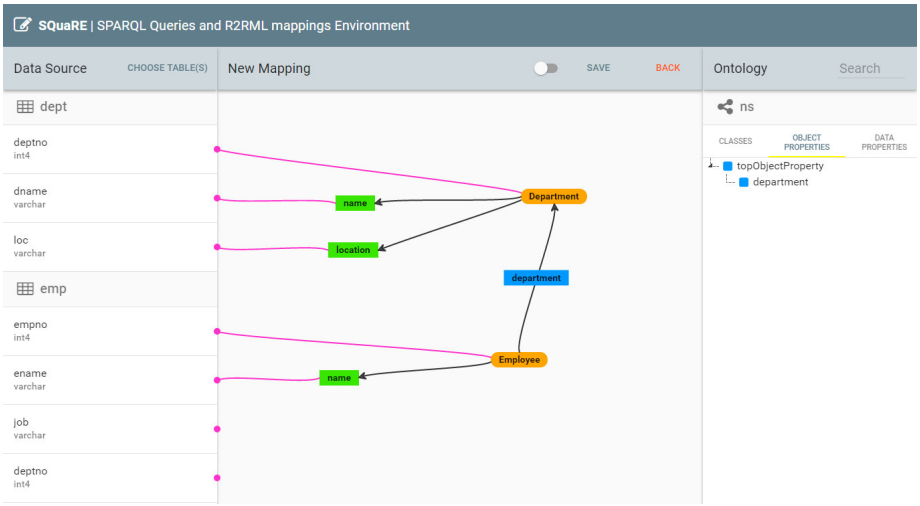

Fig. 5: Mapping creation view of W3C (1) Project.

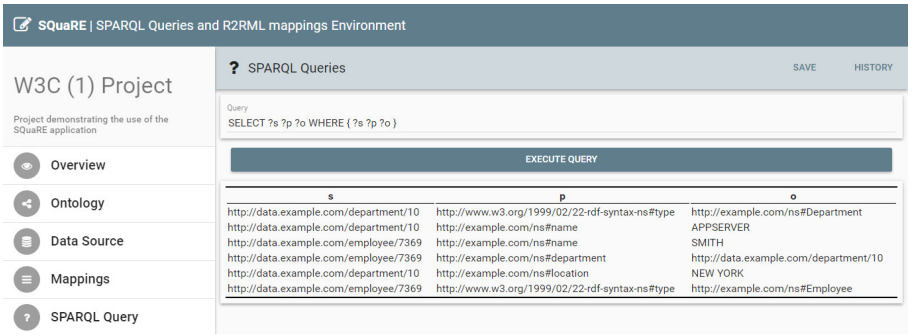

Fig. 6: The SPARQL SELECT query result.

color indicates literals, blue and green colors indicates object type properties and datatype properties, respectively (shown in Figures 7 and 8).

The SPARQL `ASK` query form returns boolean answer indicating whether a query pattern matches or not. Boolean query results are presented in Figure 9 (positive) and in Figure 10 (negative).

The W3C R2RML Recommendation's section 2.6 example [4] shows mapping of many-to-many relationship. The description includes a sample tables with exemplary contents. It also includes R2RML mappings and expected output triples. The above-mentioned mappings mapped with the use of SQuaRE application are shown in Figure 11. The example SPARQL `CONSTRUCT` query result is shown in Figure 12.

As a result we obtained an environment that is ready to be used by an inexperienced user.

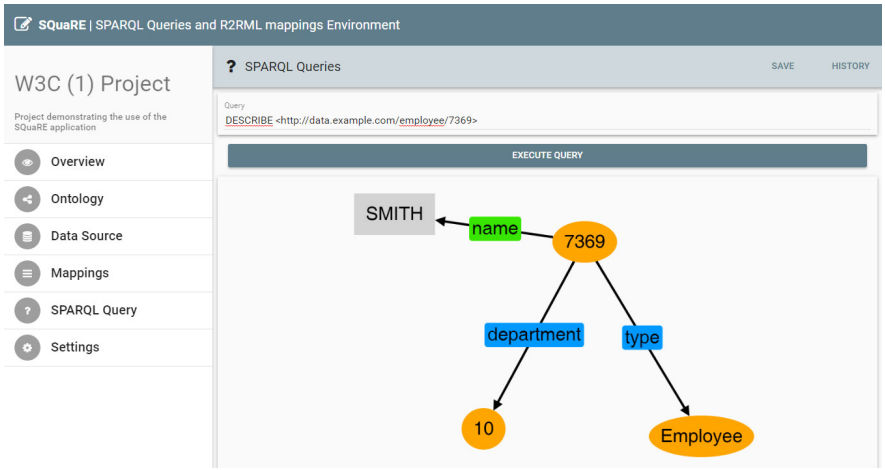

Fig. 7: The SPARQL DESCRIBE query result.

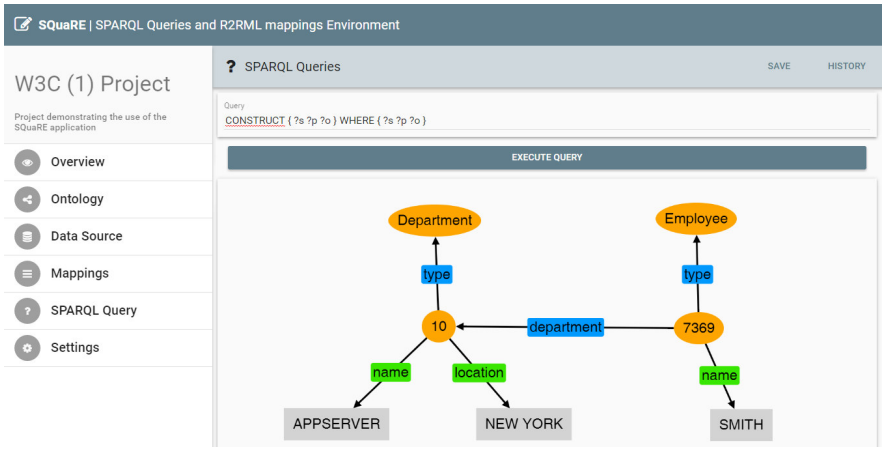

Fig. 8: The SPARQL CONSTRUCT query result.

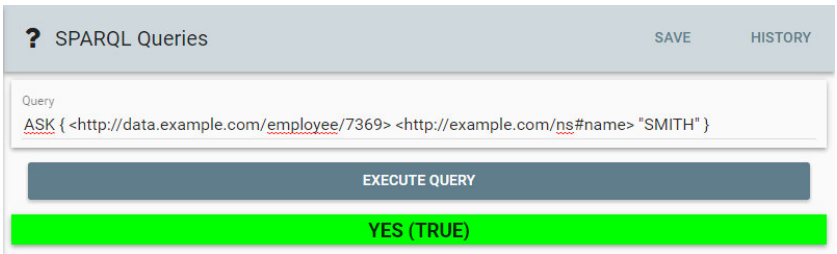

Fig. 9: The SPARQL ASK query result with positive answer.

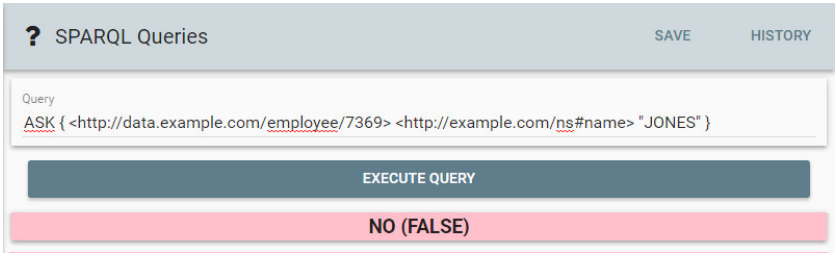

Fig. 10: The SPARQL ASK query result with negative answer.

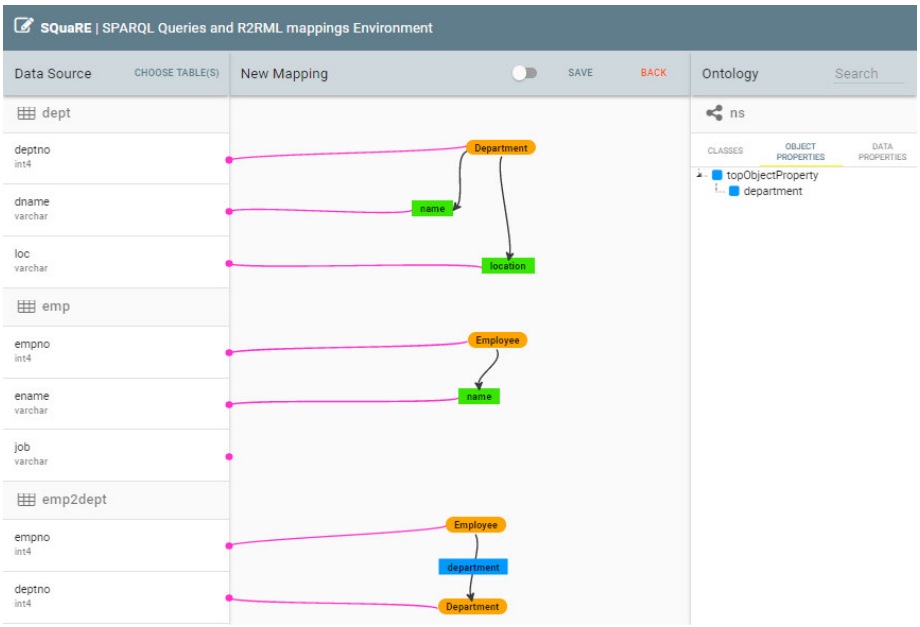

Fig. 11: The mappings of many-to-many relationship.

5 Related Tools

Several tools have been implemented to support a user in defining mappings between data sources and ontologies. We provide the list of the most similar tools to SQuaRE:

– OntopPro [3] – it provides a mapping editor inside Protégé. It allows to create mappings and to execute SPARQL queries. We can also generate RDF data according to an ontology, mappings and a data source. Users need to fill special templates in order to create mappings. In this case when creating mappings the user needs to understand how they work and how to define them. There is no graphical layout of mappings.

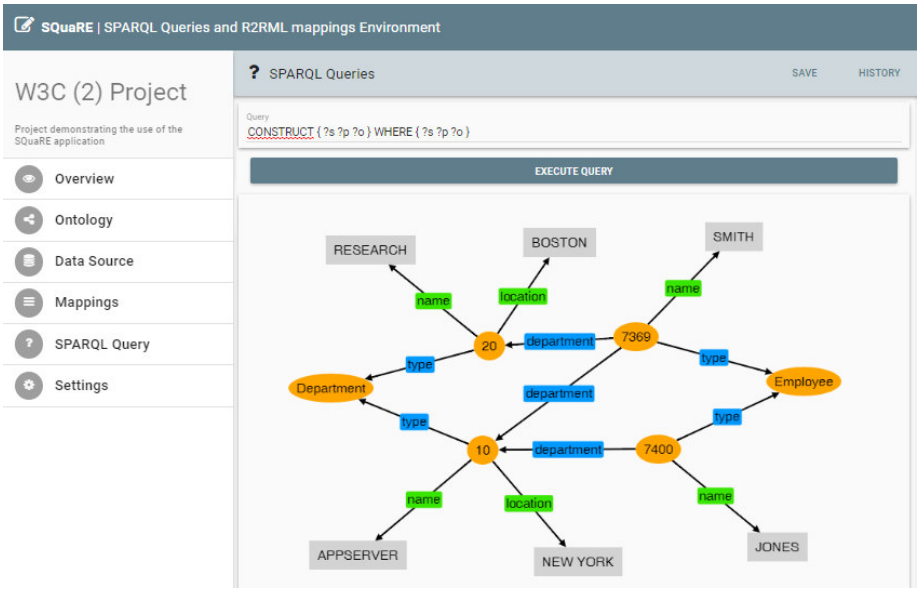

Fig. 12: The mappings of many-to-many relationship.

- Map-On [14] – it provides a graph layout for creating mappings as well as viewing ontologies and databases. This kind of representation is very convenient but when manipulating a number of mappings it is quite easy to be lost which element comes from an ontology and which one from a database. Nevertheless, the tool is very handy and easy to use.
- ODEMapster [13] – it provides a method of creating mappings in a graphical way. Mappings are expressed in the R2O language[1]. It supports OWL and RDF(S) ontologies (not OWL2) and MySQL/Oracle databases. The tool supports a tree graphical layout for database schema and ontology.
- Karma [9] – it is a web application that provides a graphical interface for creating and managing mappings between ontology and different data sources (relational databases, JSON, CSV etc.). It supports R2RML recommendation and tree layout of an ontology. However, data sources are represented with a table-like layout. Mappings are represented as a graph.
- RBA (R2RML By Assertion) [10] – it supports a tree layout for displaying databases and ontologies. R2RML mappings can be created by defining SQL queries (aka views) and then generating appropriate mappings. However, we are not able to see a graphical form of mappings.

The aforementioned tools provide different features that overlap in some cases. However, none of them provide the comprehensive functionality for OBDA-based scenario. SQuaRE provides features for creating and managing of both: R2RML mappings and SPARQL queries. Moreover, it supports users in the execution of queries and presents results in a graphical way. Moreover, we are going

to implement support for a graphical creation of SWRL rules [8], which will be another difference to the aforementioned tools.

SQuaRE is aimed at providing a simple user interface and easy to use methodology. Nevertheless, it should be perceived as a tool that tries to acquire the best features of other applications and provide them in a graphical way with an easy-to-use interface. The most similar tool at this stage of development is Karma, but without handling SPARQL queries and results in a graphical manner (but Karma provides more mapping methods than SQuaRE and more features regarding data integration). It is worth to notice that SQuaRE is still at the early stage of development whereas most of the tools from the list are being developed in the last few years. Some of them are even discontinued, like ODEMapster or RBA.

6 Summary and Future Work

In this paper we presented the SQuaRE tool which is a web-based environment that provides: (i) creation of R2RML mappings between relational databases and OWL ontologies, and (ii) creation and execution of SPARQL queries. The tool provides a lot of useful features that can be applied in an OBDA-based scenario.

Currently, we are developing a graph-based method for creating SPARQL queries. In this case we will fully support a graphical environment for handling R2RML and SPARQL. We also plan to include RuQAR [2] to extend reasoning capabilities and provide support for SWRL rules. Moreover, the long term plans are to support other mapping languages, like D2RQ[13] and RML [5]. As a result we will be able to map different data sources like CSV, JSON and others. We plan to make SQuaRE open source as soon as we finish the graphical method of creating SPARQL queries.

Acknowledgments. The work presented in this paper was supported by 04/45/DSMK/0158 project.

References

1. Jesús Barrasa, Óscar Corcho, and Asunción Gómez-pérez. R2o, an extensible and semantically based database-to-ontology mapping language. In *in In Proceedings of the 2nd Workshop on Semantic Web and Databases(SWDB2004*, pages 1069–1070. Springer, 2004.
2. Jarosław Bąk. RuQAR : Reasoning with OWL 2 RL using forward chaining engines. In *Informal Proceedings of the 4th International Workshop on OWL Reasoner Evaluation (ORE-2015) co-located with the 28th International Workshop on Description Logics (DL 2015), Athens, Greece, June 6, 2015.*, pages 31–37, 2015.
3. Diego Calvanese, Benjamin Cogrel, Sarah Komla-Ebri, Roman Kontchakov, Davide Lanti, Martin Rezk, Mariano Rodriguez-Muro, and Guohui Xiao. Ontop: Answering sparql queries over relational databases. *Semantic Web*, (Preprint):1–17.

[13] `http://d2rq.org/`

4. Souripriya Das, Richard Cyganiak, and Seema Sundara. R2RML: RDB to RDF mapping language. W3C recommendation, W3C, September 2012. http://www.w3.org/TR/2012/REC-r2rml-20120927/.
5. Anastasia Dimou, Miel Vander Sande, Pieter Colpaert, Ruben Verborgh, Erik Mannens, and Rik Van de Walle. RML: a generic language for integrated RDF mappings of heterogeneous data. In *Proceedings of the 7th Workshop on Linked Data on the Web*, April 2014.
6. Ugur Dogrusoz, Erhan Giral, Ahmet Cetintas, Ali Civril, and Emek Demir. A layout algorithm for undirected compound graphs. *Information Sciences*, 179(7):980 – 994, 2009.
7. Matthew Horridge and Sean Bechhofer. The OWL API: A Java API for OWL Ontologies. *Semant. web*, 2(1):11–21, January 2011.
8. Ian Horrocks, Peter F. Patel-schneider, Harold Boley, Said Tabet, Benjamin Grosof, and Mike Dean. SWRL: A semantic web rule language combining OWL and RuleML. 2004. Accessed: 04/04/2013.
9. Craig A. Knoblock, Pedro Szekely, José Luis Ambite, Aman Goel, Shubham Gupta, Kristina Lerman, Maria Muslea, Mohsen Taheriyan, and Parag Mallick. *Semi-automatically Mapping Structured Sources into the Semantic Web*, pages 375–390. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.
10. Luís Eufrasio T. Neto, Vânia Maria P. Vidal, Marco A. Casanova, and José Maria Monteiro. *R2RML by Assertion: A Semi-automatic Tool for Generating Customised R2RML Mappings*, pages 248–252. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.
11. W3C OWL Working Group. *OWL 2 Web Ontology Language: Document Overview*. W3C Recommendation, 11 December 2012. Available at `http://www.w3.org/TR/owl2-overview/`.
12. Antonella Poggi, Domenico Lembo, Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Riccardo Rosati. *Linking Data to Ontologies*, pages 133–173. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.
13. Jesús Barrasa Rodriguez and Asunción Gómez-Pérez. Upgrading relational legacy data to the semantic web. In *Proceedings of the 15th international conference on World Wide Web*, pages 1069–1070. ACM, 2006.
14. Álvaro Siciliaa, German Nemirovskib, and Andreas Nolleb. Map-on: A web-based editor for visual ontology mapping. *Semantic Web Journal*, (Preprint):1–12.