R2RML-based access and querying to relational clinical data with morph-RDB

Freddy Priyatna[1], Raul Alonso Calvo[2], Sergio Paraiso-Medina[2], Gueton Padron-Sanchez[2], and Oscar Corcho[1]

[1] Ontology Engineering Group, Universidad Politécnica de Madrid, Spain
{fpriyatna,ocorcho}@fi.upm.es
[2] Biomedical Informatics Group, Universidad Politécnica de Madrid, Spain
{ralonso,sparaiso,gpadron}@infomed.dia.fi.upm.es

Abstract. Semantic interoperability is essential when carrying out post-genomic clinical trials where several institutions collaborate, since researchers and developers need to have an integrated view and access to heterogeneous data sources. In this paper we present a solution that uses an ontology based on the HL7 v3 Reference Information Model and a set of R2RML mappings that relate this ontology to an underlying relational database implementation, and where morph-RDB is used to expose a virtual SPARQL endpoint over the data. In previous efforts with other existing RDB2RDF systems we had not been able to work with live databases. Now we can issue SPARQL queries to the underlying relational data with acceptable performance, in general.

1 Introduction

In the last years, clinical trials have started introducing genomic variables [10], what requires performing patient stratification when selecting the patient population to apply the clinical trials to. This involves the use of biomarkers to create subsets within a patient population that provide more detailed information about how the patient will respond to a given drug. Several datasets, commonly produced by different institutions and hence rather heterogeneous in general, need to be used for patient stratification [8]. Interoperability among those datasets is made easier by the usage of biomedical standards and vocabularies [9] . However, achieving such interoperability poses relevant technological challenges. This is the basis of the work presented in this paper, which aims at being applied in several healthcare institutions, such as the Institut Jules Bordet[3], the MAASTRO Clinic[4], and the German Breast Group[5].

In previous works [12] we have already presented a relational database implementation that is based on the HL7 version 3 Reference Information Model (RIM) [1]. This database aims to facilitate the interconnection with other data

[3] http://www.bordet.be/
[4] http://www.maastro.nl/
[5] http://www.germanbreastgroup.de/

sources where medical ontologies are also being used, and has already been used for providing some form of interoperability among real data sources [12] from the aforementioned institutions. Currently we are working in providing ontology-based support to data access, so as to facilitate such integration and allow incorporating other datasets more easily. This is the reason why we are looking into using a Relational Database to RDF (RDB2RDF) solution. We also provide a SPARQL endpoint so that users are relieved from knowing the underlying schema of the implemented database.

RDB2RDF mappings are used to expose data from relational databases as RDF datasets. Two major types of data access mechanisms are normally provided by RDB2RDF tools: i) data translation (a specific case of ETL - Extract, Transform, Load -), where data are materialized into RDF datasets and stored in a triple store (e.g., Virtuoso), which provides a SPARQL endpoint; and ii) query translation, where SPARQL queries are directly translated into SQL according to the specified RDB2RDF mappings, and evaluated against the relational database, and where results are translated back using the mappings to conform with the SPARQL query. In our case, we are interested in using RDB2RDF mappings to make the data stored in our SQL implementation available according to an ontology that reflects the HL7 version 3 RIM. Furthermore, we have a strong requirement to use a query translation approach, given the importance of having fresh results, what cannot be always ensured in the data translation approach.

Our first attempt [11] at applying RDB2RDF-based query translation was with D2R server and mappings [2]. This approach was not applicable since the evaluation of the SQL queries resulting from query translation was not efficient enough. Moreover, in some cases, queries could not be executed by the database management system (e.g., their length was excessive). This has been already hinted in [7] that describes the experience of using RDB2RDF tools in the domain of astronomy. The conclusion there was that RDB2RDF tools were not feasible to be used in such a context, and this conclusion was consistent with our first attempt.

Later, we started using morph-RDB [13] with R2RML mappings [5] for this purpose. We have obtained better results that make this approach applicable in our context. In this paper we describe our experience, which shows that it is possible to use efficient RDB2RDF tools in the medical domain.

This paper is organized as follows. Section 2 discusses the necessary background such as our current model for storing medical data, the HL7 RIM ontology, the R2RML mapping language, and our query translation engine morph-RDB. We discuss some optimization techniques in Section 3. In Section 4 we discuss the set of queries that we use for the evaluation. Finally in Section 5 we provide some conclusions and describe some of our future work in this area, including our deployment plans in aforementioned healthcare institutions.

2 Background: HL7, R2RML, and morph-RDB

In this section we will review the main foundations of the work that we present in the paper, namely HL7 and the HL7 RIM, the R2RML language, and morph-RDB.

2.1 HL7 RIM

Recent years have witnessed a huge increase of biomedical databases [6]. This larger availability opens up new opportunities, while setting some new important challenges, especially in what respects to their integration, which is crucial to obtain a proportional increment of knowledge on the biomedical area.

Among the many Detailed Clinical Models that have been reviewed for the integration of biomedical datasets [4], the HL7 v3 is one of the most relevant. The HL7 v3 standard defines the RIM at its core, this definition consists of a UML class diagram (it does not define a data structure or a database model). Besides, issues such as the management of data types are not trivially translatable into a database model. As a consequence, we have defined in the past a relational model for it, as described in [12].

The HL7 RIM backbone contains three main classes (`Act`, `Role` and `Entity`), which are linked together using three association classes (`Act-Relationship`, `Participation` and `RoleLink`). The core of the HL7 RIM is the `Act` class. An `Act` is defined as "a record of an event that has happened or may happen". Any healthcare situation and all information concerning it should be describable using the RIM by including the type of act (what happens), the actor who performs the deed and the objects or subjects `Entity` that the act affects to `Role`. Some additional information may be provided to indicate location (where), time (when), manner (how), together with reasons (why) or motives (what for). `Act` and `Entity` classes have some specializations that add some attributes, such as `Observation` (a subclass of `Act`), or `Person` (a subclass of `Entity`).

This standard is able to represent all kinds of healthcare situations and any kind of information associated with it. Based on this idea, we have defined a subset of the HL7 RIM schema, where we implement the classes and attributes that are necessary to represent the scenario for sharing clinical data of breast cancer clinical trials:

- `Act`, with the subclasses `Observation`, `Procedure`, `SubstanceAdministration`, and `Exposure`.
- `Role`.
- `Entity`, with the sub-classes `LivingSubject`, `Person`, and `Device`.
- The classes; i)`ActProcedureApproachSiteCode`, ii) `ActMethodCode`, iii) `ActTargetSiteCode`, iv) `ActObservationInterpretationCode`, and v) `ActObservationValues` related to `Act`.

Besides, attribute data types are rather complex on the RIM, so they are changed according to the mentioned scenario, following HL7 recommendations.

Therefore some attributes were simplified in the relational model compared to those defined by HL7 v3 standard. To improve a better performance and knowledge on the HL7 RIM schema, it is defined a set of views. These views cover the access retrieval requirements for the clinical scenario. Therefore it is implemented a different view for obtaining patient data due to its provenance (`Observation`, `Procedure`, `SubstanceAdministration`, and `Exposure`).

Therefore, the defined HL7 RIM based CDM above fulfills the requirements needed by breast cancer in clinical trials scenario. Furthermore, we have created an ontology that reflects the HL7RIM model[6], available for others to reuse.

2.2 R2RML

R2RML [5] is a W3C recommendation for the definition of a mapping language from relational databases to RDF. An R2RML mapping document consists of a set of Triples Maps `rr:TriplesMap`, used to specify the rules to generate RDF triples from database rows/values. A TriplesMap consists of:

- A logical table `rr:LogicalTable` that is either a base table or SQL view, used to provide the rows to be mapped as RDF triples.
- A subject map `rr:SubjectMap` that is used to specify the rules to generate the subject component of RDF triples.
- A set of predicate object maps `rr:PredicateObjectMap` that is composed by a set of predicate maps `rr:PredicateMap` and object maps `rr:ObjectMap` (to generate the predicate and object components of RDF triples, respectively). If a join with another triples map is needed, a reference object map `rr:RefObjectMap` can be used. The other triples map to be joined is specified in `rr:parentTriplesMap` and the join condition is specified via `rr:Join`

Subject maps, predicate maps, and object maps are term maps, which are used to specify rules to generate the corresponding RDF triples element, and those rules can be specified as a constant `rr:constant`, a database column `rr:column`, or a template `rr:template`.

2.3 morph-RDB

morph-RDB, which belongs to the morph suite[7], i.e., receives as an input the connection details to a relational database, an R2RML mapping document and a SPARQL query and translates the query into SQL according to the R2RML mapping, evaluates it into the underlying relational database and translates back those results into the format required as a result of the SPARQL query evaluation. The query translator component in morph-RDB implements the algorithm described in [13], which extends previous work in [3] that defines a set of mappings and functions in order to translate SPARQL queries posed into RDB-backed triples stores into SQL queries, and prove that the correctness of

[6] `http://www.gib.fi.upm.es/hl7rim-common-data-model/`
[7] `http://www.oeg-upm.net/index.php/en/technologies/334-morph`

the query translation using the notion semantic-preserving, in other words, the results of the SPARQL and SQL return the same answers. We extend their work by relating those mappings and functions with the R2RML mapping elements.

Example 1. Consider the following table `v_person(patientId, patientName, gender, actId)` that stores the information about patients. This table is mapped to class `Patient` with the attribute `patientId` as the identifier (together with base URI for class `Patient`) of the instances. Attributes `patientId` and `patientName` are mapped to ontology properties `hasID` and `hasName`, respectively. Now let's add another table `v_observation(actId, title, code)` that describes observations. This table is mapped to class `Observation` with `actId` as the identifier of the instances, and the attribute `title` is mapped to property `hasTitle`. `patientId` and `actId` are primary keys of the tables `v_person` and `v_observation`, respectively. Furthermore, the `actId` of table `v_person` is a foreign key that refers to the column `actId` of table `v_observation`, and this relation is mapped to property `hasObservation`.

The instances of the tables can be seen in Figure 1.

v_person

patientId	patientName	gender	actId
1	Alice	F	33
2	Mary	F	23
3	Kate	F	12
4	Andrea	F	45

v_observation

actId	title	code
12	pN3 category (finding)	49182004
15	N0 category (finding)	62455006
23	T2 category (finding)	67673008
29	Neutropenic disorder (disorder)	303011007
33	Body mass index (observable entity)	60621009
36	Leukopenia (disorder)	84828003
41	Anemia (disorder)	271737000
45	G2 grade (finding)	1663004

Fig. 1. Tables Person and Observation

Now consider the following graph pattern.

```
{ ?p :hasID ?pID ; :hasName ?pName . }
```

The SQL query translated by morph-RDB will be:

```
SELECT T1.patientId, T2.patientName
FROM T1 INNER JOIN T2 ON T1.patientId = T2.patientId;
```

where `T1 = (SELECT patientId FROM v_person WHERE patientId IS NOT NULL)` and `T2 = (SELECT patientId, patientName FROM v_person WHERE patientId IS NOT NULL AND patientName IS NOT NULL)` are the results of translating the first and second triple patterns, respectively.

3 Query Optimizations.

The query translation technique presented above do not necessarily generate optimal SQL queries. Based on the set of SPARQL queries we have evaluated, we have observed that several patterns occur frequently. Hence we describe optimization techniques that can be applied in order to generate more efficient queries.

– **Self-join elimination**. A set of triple patterns connected by the `AND` operator and sharing the same subject occur frequently. We call this pattern *Subject Triple Group* (STG). Using a naïve query translation, each triple pattern corresponds to an INNER join in the generated SQL query. We have extended our query translation technique so that in addition to handling triple patterns, each of the mappings/functions is also able to handle STG.

  Example 2. Consider again the SPARQL query in Example 1. With self-join elimination, now the result of translating that query is:

```
SELECT T1.patientId, T1.patientName FROM v_person
WHERE patientId IS NOT NULL AND patientName IS NOT NULL;
```

– **Left-outer join to inner join**. Another pattern is a *Subject Triple Group with Optional* (OSTG), that is an `OPTIONAL` pattern that consists only one triple pattern, preceded by an STG pattern or a triple pattern. Because the `OPTIONAL` keyword corresponds to a left outer join, naively translating this pattern produces one left-outer join for each `OPTIONAL` pattern. We extend our query translation technique, so that the optimized query translation generates an inner join, cheaper to evaluate than left-outer join, by removing the conditional expression `IS NOT NULL` corresponding to the function *genCondSQL* of the triple pattern in the `OPTIONAL` pattern.

  Example 3. Consider the following pattern:

```
{ ?p :hasID :pid . OPTIONAL { ?p :hasName ?pname . } }
```

  Without left-outer join elimination, the result of translating this query is:

```
SELECT T1.patientId, T2.patientName FROM
(SELECT patientId FROM v_person
WHERE patientId IS NOT NULL) T1 LEFT OUTER JOIN
(SELECT patientId, patientName FROM v_person
WHERE patientId IS NOT NULL AND patientName IS NOT NULL) T2
ON T1.patientId = T2.patientId;
```

  By changing the type of join from left-outer to inner, and removing the conditional expression `name IS NOT NULL`, and applying the self-join elimination (O1), the optimized query generated becomes:

```
SELECT T2.patientId, T2.patientName FROM v_person T2;
```

– **Phantom triple pattern introduction**.

  Example 4. Consider the following pattern, which is neither an STG pattern nor an OSTG, thus, none of the aforementioned optimizations can be applied.

```
{ ?p :hasObservation ?o .OPTIONAL { ?s :hasTitle ?t . } }
```

  In order to exploit those optimisations we have presented so far, this query has to be transformed into another query whose resulting query translation can be optimised. To do that, we use the fact that for every IRI x, the fact (x *a rdf:Resource*) holds, so that we can safely add this triple pattern to the query without changing its semantics. We call such triple pattern a *phantom triple pattern*. The result of adding the phantom triple pattern is:

```
{ ?p :hasObservation ?s . ?s rdf:type rdf:Resource .
OPTIONAL { ?s :hasTitle ?t . } }
```

Now with the new pattern that emerged, the optimization for OSTG pattern can be applied.

4 Evaluation

We have collected a total of 45 SPARQL queries that are used in the patient recruitment and cohort selection scenario for breast cancer clinical trials. The complete list of queries and their natural language descriptions are available at `http://bit.ly/1LecA7L`. From this query list, we asked our domain experts to group the queries into a set of five groups that are representative of the whole set, and they are shown in Table 1.

Representative query	Similar queries	TP	subjects	OPT	FILTER
Q01	1, 2, 5, 15, 19, 37, 41, 42	4	2	1	0
Q10	3, 10, 11, 40	25	9	21	4 (IN)
Q14	4, 6, 9, 14, 16, 17, 18, 19, 21, 35, 38, 39, 43	15	6	5	2 (IN, arithmetic)
Q34	22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 44	6	3	1	1 (IN)
Q45	7, 8, 12, 13, 20, 36, 45	14	5	4	2 (IN)

Table 1. Grouping of all queries according to the use case queries

The query characteristics of each the representative query are as follows:

- **Demographics query (Q01)**. This query retrieves the information of all patients. It contains 4 triple patterns, 2 unique subjects, 1 triple pattern that is inside an OPTIONAL block, and 1 FILTER pattern.
- **Substance administration query (Q10)**. This query retrieves the information of patients who were administered diphosphonate, including the information associated with the target site, the method used, and the approach site, if it exists. It consists of 35 triples patterns with 9 unique subjects. Most of the triple patterns are inside nested OPTIONAL blocks. There are 21 OPTIONAL blocks in this query, some of which are nested under another OPTIONAL block. A FILTER pattern is used to filter results based on a certain condition.
- **Laboratory results query (Q14)**. This query retrieves the information of patients who suffer anemia and whose body mass index is less than or equal to 30. It contains 15 triple patterns, with six unique subjects and 5 OPTIONAL patterns. Furthermore, some of the OPTIONAL blocks are nested inside a parent OPTIONAL block. This query also contains two FILTER patterns to filter results for particular code values and to perform some arithmetic calculations.
- **Procedure query (Q34)**. This query retrieves the information of all patients who were administered chemotherapy. It consists of 6 triple patterns with one of them located inside an OPTIONAL pattern, and one FILTER pattern. There are 3 unique subjects in this query.

– **Observation query (Q45)**. This query retrieves the information of patients who have been detected a category T2 breast tumor. It consists of 14 triple patterns and 5 unique subjects. There are 4 OPTIONAL patterns, one of them nested, and 2 FILTER patterns.

The machine used in our evaluation has the following specifications: CPU Intel(R) Xeon(R) CPU E5-2650 0 @ 2.00GHz, 8 GB of RAM, 750 GB HDD with Ubuntu Server 12.04 and MySQL Server 5.5. The dataset contains information of 3 months of historical clinical data, with 4674 patients and 65056 acts, among many other tables. The total size of the database is 105 MB. This database will be growing in the future, as more data is added as a result of the data integration processes carried out in the context of the projects where the database is being generated.

We were interested in comparing morph-RDB with another well-established RDB2RDF engine, such as D2R, considering the total time required for the execution of the SPARQL queries. That is, we include in our calculation the time required to initialize the engine, the time needed for SPARQL-to-SQL query translation, the time needed to evaluate the SQL queries, and the time needed to translate back the result from the database using the mappings into the result expected by the SPARQL queries. Figure 2 provides details for the five selected queries, which are also similar to the results obtained for the other queries in our query set. We can easily see that in most cases our total execution time is much lower than the one required for D2R Server. In some cases (queries Q14 and Q45) D2R Server was not able to produce results in less than five minutes. The results for the rest of the queries are available at `http://bit.ly/1LecA7L`.

We were also interested in how the SQL queries that result from the query rewriting approach perform in comparison to the SQL queries that would have been natively created by a SQL expert. For this reason, we asked a domain expert with good knowledge of the HL7 RIM relational database to construct SQL queries that were semantically equivalent to the corresponding SPARQL queries. In other words, without taking into account the mapping elements, such as template or URI generation, the SPARQL and SQL queries should return the same answer.

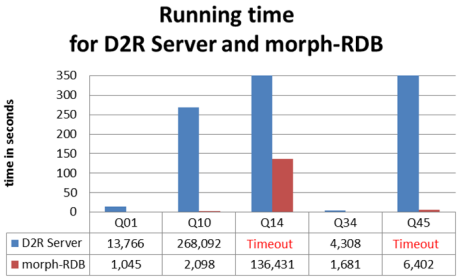

Fig. 2. Running time for our selected queries on morph-RDB and D2R (in seconds)

We evaluated each query 5 times in cold and warm modes. In the cold mode, we restart the server and empty the cache before we evaluate the next query. In

the warm mode, we skip these steps and execute the queries directly one after the another. We measure the averages of query execution time and normalize the query evaluation time to the native query evaluation time. As an additional note, we can only do this type of evaluation using morph-RDB and native queries, as D2R Server produces multiple SQL queries in many cases and performs joins in memory, what makes it not comparable with the native or morph-RDB queries.

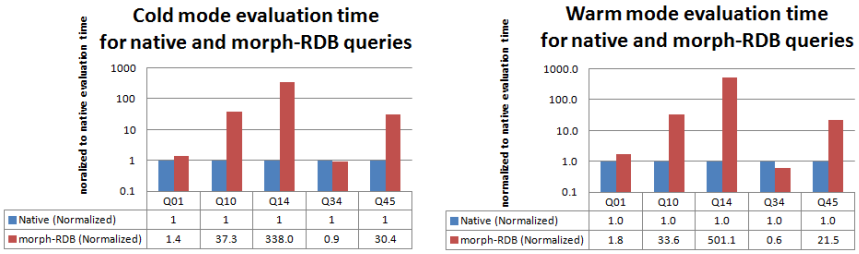

Fig. 3. Query evaluation time in warm and cold modes

The results from both evaluation modes show a similar trend. Furthermore, we observed that in the warm mode, the database server doesn't lose its capability of reusing previous results of the query cache. This is reflected by the fact that only the first run of the query takes more time to complete, while subsequent queries can be evaluated with only a fraction of that time. Some of those queries produced by morph-RDB can be evaluated in a reasonable time. For example, the resulting query translation of query Q01 can be evaluated in a similar time as the native query Q01. Furthermore, the resulting query translation Q34 can be evaluated in less time than its corresponding native queries, which can be an indicator that there might be still room for improving the corresponding native query. Some other queries, such as Q10 and Q45, need more time to be evaluated, being in the range of 20-35x slower than the corresponding native queries, in which we still consider them as acceptable.The query Q14, however, needs more investigation, as it takes a lot of time to be evaluated, 380-500x slower in terms of normalized time to native queries. We suspect this is caused by the arithmetic operation that is performed over the resulting translation queries.

5 Conclusion

In this paper we have shown that SPARQL queries can be used as a means to query relational clinical data that is integrated into an HL7 version 3 RIM database implementation. We collected a set of 45 real SPARQL queries required by our application domain and that will be deployed in a set of medical institutes, chose five of them as the most representatives ones, and evaluated those queries using D2R Server and morph-RDB as our RDB2RDF tools. We have shown that, in general, we got a better result with morph-RDB than D2R Server, what allows now using this approach for accessing relational data using SPARQL.

However, there are still some important remaining challenges to be considered. We still have queries that require too much time to be evaluated, (e.g. query Q14), because of the arithmetic operation that is included in the SPARQL query

and in its resulting translation into SQL. Investigating and designing optimizations for dealing with this type of query will be part of our future work.

Acknowledgements

This work has been funded by Ministerio de Economía y Competitividad (Spain) under the project "4V: Volumen, Velocidad, Variedad y Validez en la Gestión Innovadora de Datos" (TIN2013-46238-C4-2-R), by the European Commission through the EURECA (FP7-ICT-2011-7-288048) project and also by the Ministry of Health of the Spanish Government under Grant PI13/02020.

References

1. G. Beeler, J. Case, J. Curry, A. Hueber, L. Mckenzie, G. Schadow, and A. Shakir. HL7 reference information model. 2003.
2. C. Bizer and R. Cyganiak. D2R server-publishing relational databases on the semantic web. In *Poster at the 5th International Semantic Web Conference*, 2006.
3. A. Chebotko, S. Lu, and F. Fotouhi. Semantics preserving SPARQL-to-SQL translation. *Data & Knowledge Engineering*, 68(10):973–1000, 2009.
4. J. F. Coyle, A. R. Mori, and S. M. Huff. Standards for detailed clinical models as the basis for medical data exchange and decision support. *International journal of medical informatics*, 69(2):157–174, 2003.
5. S. Das, S. Sundara, and R. Cyganiak. R2RML: RDB to RDF mapping language. 2012.
6. M. Y. Galperin and X. M. Fernández-Suárez. The 2012 nucleic acids research database issue and the online molecular biology database collection. *Nucleic acids research*, 40(D1):D1–D8, 2012.
7. A. Gray, N. Gray, and I. Ounis. Can RDB2RDF tools feasibily expose large science archives for data integration? In *Proceedings of the 6th European Semantic Web Conference on The Semantic Web: Research and Applications*, pages 491–505. Springer-Verlag, 2009.
8. W. R. Hersh. Adding value to the electronic health record through secondary use of data for quality assurance, research, and surveillance. *Clin Pharmacol Ther*, 81:126–128, 2007.
9. J. Kaufman. Healthcare and life sciences standards overview-technology for life: NC symposium on biotechnology and bioinformatics. In *Biotechnology and Bioinformatics, 2004. Proceedings. Technology for Life: North Carolina Symposium on*, pages 31–41. IEEE, 2004.
10. L. M. McShane, M. M. Cavenagh, T. G. Lively, D. A. Eberhard, W. L. Bigbee, P. M. Williams, J. P. Mesirov, M.-Y. C. Polley, K. Y. Kim, J. V. Tricoli, et al. Criteria for the use of omics-based predictors in clinical trials. *Nature*, 502(7471):317–320, 2013.
11. J. M. Moratilla, R. Alonso-Calvo, G. Molina-Vaquero, S. Paraiso-Medina, D. Perez-Rey, and V. Maojo. A data model based on semantically enhanced HL7 RIM for sharing patient data of breast cancer clinical trials. *Studies in health technology and informatics*, 192:971–971, 2012.
12. S. Paraiso-Medina, D. Perez-Rey, A. Bucur, B. Claerhout, and R. Alonso-Calvo. Semantic normalization and query abstraction based on SNOMED-CT and HL7: Supporting multi-centric clinical trials. 2015.
13. F. Priyatna, O. Corcho, and J. F. Sequeda. Formalisation and experiences of R2RML-based SPARQL to SQL query translation using morph. In *Proceedings of the 23rd International World Wide Web Conference*, 2014.