Journal of Theoretical and Applied Information Technology
30th November 2020. Vol.98. No 22

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

SEMANTIC REPRESENTATION OF MUSIC DATABASE USING NEW ONTOLOGY-BASED SYSTEM

AHMED M. DAOUD*, KHALID M. HOSNY, EHAB R. MOHAMED
Faculty of Computers and Informatics, Zagazig University, Department of Information Technology,
Zagazig 44519, Egypt
E-mail: developerdaoud@gmail.com

ABSTRACT

The Semantic Web is suffering from a lack of tools used to facilitate its users' work when extracting structured data. The development of a system that facilitates the extraction of data from databases and converts them into useful information is of great interest. In this paper, a new ontology-based method is proposed to extract the structured data from a database based on a target ontology. The main focus here is on dealing with music databases because of their popularity on the web. Since the Semantic Web built upon reusing existing ontologies, this system is going to use the existing musical ontology defined in the Music Ontology Specification. This specification provides the concepts and properties used to describe music besides other ontologies that could use in conjunction with it, such as (foaf, dc, timeline, event, etc.). The resulting music data, in RDF format, can be published and linked with existing musical data on the web. This method contributes to the music web of data and allows the Semantic Web clients to access detailed structured information about musical data easily.

Keywords: *Semantic Web (SW), Relational Database (RDBs), Resource Description Framework (RDF), Music Ontology, Music Database*

1. INTRODUCTION

The Semantic Web [1] is a relevant emerging research field where several research efforts made to convert data into a machine-readable format. This enables devices to perform manual tasks alongside humans. The World Wide Web Consortium (W3C) commit has recommended languages, such as RDF, RDFS, and OWL, to make users describe knowledge and data in a structured and semantically related way. RDF [2] is a graphical data model standardized by the W3C. It is used for describing resources in any domain to provide interoperability between web applications. It is also considered the core component constructing the Semantic Web. It can be queried by SPARQL [3], the recommended language for querying RDF data. It allows for such structured representations to be made and expresses statements about web resources as triples: subject (resource), predicate (property), and object (value). Each triple represents a piece of knowledge for describing a resource. When these representations quote other resource identifiers, they enable access to corresponding structured representations, creating a Web of structured data.

On the other hand, databases have played a dominant role in managing and storing data sources in the last decades. Backend relational databases (RDBs) store vast amounts of related data, according to He et al. [4], Almost 70% of the websites backed up using legacy relational databases. In the year 2004, the results showed that there were 450,000 Deep Web databases. It turns out that the number of pages had increased by nearly nine times after only four years. It is more effective and scalable to construct RDF data directly from the tables of these relational databases.

The W3C RDB2RDF Working Group (RDB2RDF-WG) defined two standards, direct mapping (DM) [5] and Relational Database to RDF Mapping Language (R2RML) [6] for mapping relational schema and data into OWL and RDF respectively. DM, called default mapping, directly maps the RDB schema using a collection of simple transformations. In contrast, the R2RML, called custom mapping language, is a language for manually create mappings from RDB tables to RDF output.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

In recent years, numerous RDB2RDF tools developed, some commercial, and other academics. The majority of these tools are console-based software with limited support for R2RML and RDF data processing. Each DM should create its terminology, and existing ontologies cannot use, and therefore it is not suitable for real application development. One simplified example of DM several tools is the D2RQ Platform [7]. On the other side, R2RML provides much more control for the user where a target and existing ontologies can use in the mapping process. All tools tested against the compliance test cases described in the R2RML and DM test cases [8], and the results reported in [9]. Examples of R2RML-based tools include DB2Triples(1), developed by Antidot and published under LGPL Open Source License, implements DM and R2RML, and supports MySQL, PostgreSQL, and Oracle DBs. It passes all compliance test cases for PostgresSQL but fails one test case for MySQL.

Morph-RDB(2), implemented using java and scala and published under Apache Open Source License, supports R2RML for HSQLDB, MySQL, PostgreSQL, Oracle, MonetDB, morph-GFT DBs. It fails 8 test cases for HSQLDB while the other DBMSs not yet tested. Oracle Database 19c(3), is a commercial tool called Oracle Spatial and Graph, has an integrated set of functions, procedures, data types, and data models that support graph analytics to enable data storage, access, and analysis in an Oracle database. It supports both DM and R2RML mapping by providing RDF views on relational tables, SQL views, and R2RML views result since version 12c, and it is not available for testing its compliance. RDF-RDB2RDF(4), an open-source library implemented using Perl, supports both DM and R2RML for MySQL, PostgreSQL, and SQLite databases. It failed to pass 11 test cases for PostgreSQL and 7 for SQLite. Virtuoso RDF Views(5) is an open-source and commercial web interface suit designed, by OpenLink Company, to meet data management, integration, and access needs. The open-source version only supports the Virtuoso-based RDB, whereas the commercial one helps the most well-known RDB systems. It only supports R2RML and its compliance for 29 test cases indicated with "can not Tell." Ultrawrap [10] is a commercial tool, developed by Capsenta Inc.(6), designed based on SQL views to map RDBs to RDF triples. Its primary function is that the runtime of the query is too fast compared to SQL queries that run directly against the database. It supports DM, R2RML, and D2RQ mapping for Oracle and PostgreSQL databases and passes all the test cases for them. XSPARQL(7) is a command-line tool, implemented in java and published under Apache Open Source License, designed to combine both XQuery and SPARQL for bidirectional mapping between RDF and XML. It supports R2RML for MySQL and PostgreSQL databases and passes all the test cases for them.

These limitations above motivate the authors to propose a new ontology-based graphical mapping system, based on R2RML specification, to support all the functionalities required when dealing with the Semantic Web. In this paper, a standalone tool with a graphical user interface (GUI) that facilitates the mapping process from RDBs to RDF and supports a diversity of other features. This tool will help not only experts but also semi-experts who intend to publish their data in RDF format. This tool is applied to the music databases and targeting the music ontology as an example of the mapping process.

The remaining of this paper is organized into five sections as follows. Section 2 presents a summary of the previous work relating to the proposed system. Section 3 explains the proposed method in detail. Section 4 discusses the experiments of the proposed method. Section 5 summarizes the results and discussion. Finally,

(1) *DB2Triples*. Available: https://github.com/antidot/db2triples/ (accessed 08 July 2020).
(2) *Morph-RDB*. Available: https://github.com/oeg-upm/morph-rdb/ (accessed 08 July 2020).
(3) *Oracle Spatial and Graph*. Available: https://www.oracle.com/database/technologies/spatialandgraph/rdf-graph-features.html (accessed 08 July 2020).
(4) *RDF-RDB2RDF*. Available: https://metacpan.org/release/RDF-RDB2RDF/ (accessed 08 July 2020).
(5) *VirtuosoUniversalServer*. Available: https://www.w3.org/wiki/VirtuosoUniversalServer (accessed 08 July 2020).
(6) *Ultrawrap*. Available: http://capsenta.com/ultrawrap (accessed 08 July 2020).
(7) *XSPARQL*. Available: https://sourceforge.net/projects/xsparql/files/xsparql/ (accessed 08 July 2020).

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Section 6 provides a conclusion of the paper with future work.

2. RELATED WORK

The mapping from a relational database to RDF is the subject of numerous research work in a range of fields. Also, there is considerable interest in dealing with multimedia, especially music-related data. Raimond et al. [11] proposed a Semantic Web framework for providing a shared knowledge management environment for integrating multimedia processing algorithms in a management information system. The primary purpose of this framework is to create a scalable management system that deals with the instances of knowledge machines. This framework developed using three stages—first, the RDF Semantic Web technologies for describing entities or resources. Second, a set of domain ontologies such as OWL for expressing knowledge about a particular domain. Third, SPARQL for querying the Semantic Web knowledge machines using SPARQL endpoints. It used for the music information management using the event concepts and the timeline concepts representing the ontology used. The main benefit of using Semantic Web technologies is that it applied to real-world applications. The resulting RDF documents are published and linked to each other on the web [12] to create an interlinked web of data known as Linked Open Data (LOD) project. The application's RDF content can be interlinked using RDF links with different data sources constructing machine-understandable data that can be harvested by machines.

Raimond et al. [13] defined an ontology providing a set of web identifiers and corresponding structured representations for an ontology of the music domain. This Music Ontology used for describing music-related data such as artists, albums, tracks, performances, etc. The Music Ontology makes use of terms used in the existing ontologies such as FOAF [14], Timeline [15], Event [16], etc. to connect it with other available ontologies on the Semantic Web. The music data expressed in a structured way interlinked with other datasets published on the web by using the proposed ontology with existing ontologies. The data expression' process results in a vast music knowledge environment. This process creates a machine-understandable web of data that interlinks the data to its meaning. It divided into three levels, the editorial information, including (tracks, people, labels, etc.), the production workflow information, including (compositions, arrangements, performances, etc.) and the event decomposition information including (event, time, etc.). Passant and Raimond [17] showed three various methods for suggesting musical recommendations. These methods include using social-networking and music, tag-based, and LOD cloud recommendations. The social-networking and music recommendations method depend on representing the listening habits in the social networks using the Music Ontology [13] and the other related ontologies like Events and FOAF. The Tag-based recommendations method allows people to tag contents using URIs instead of simple keywords and then use the relationships between these URIs to suggest the related data. The LOD cloud recommendations method finds relevant musical content by finding the relevant paths between concepts using the publicly available distributed data sources of music-related information from the LOD cloud for making useful musical recommendations. The extracted RDF from these methods, combined, is used for answering complicated SPARQL queries and for reasoning and recommendations.

Raimond et al. [18] demonstrated the benefits of using the Semantic Web technologies to create the web of data and develop an On-demand interpreter called Henry, with a Henry instance service, for dynamically creating and publishing linked data. It used for combining the music analysis results and contextual data. It has a SPARQL endpoint interface interpreting signal processing workflows and providing on-demand content-based data for running SPARQL queries. Thus, a vast number of various data relating to music open data sources have been published and interlinked to the LOD project. Fazekas et al. [19] overviewed the Semantic Web technologies and numerous applications developed in the OMRAS2 project. Then show how these applications, together with the music ontologies, can be used in musicological work to publish and interlink several music-related datasets, such as (Magnatune[8], Jamendo[9], Peel[10], etc.) in the DBTune server[11],

[8] *Magnatune*. Available: http://dbtune.org/magnatune/ (accessed 08 July 2020).
[9] *Jamendo*. Available: http://dbtune.org/jamendo/ (accessed 08 July 2020).
[10] *Peel*. Available: http://dbtune.org/bbc/peel/ (accessed 08 July 2020).

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

with the LOD project. These applications categorized into two types. The first is on-demand applications that contribute to the Semantic Web by aggregating data resulted from running SPARQL queries against SPARQL endpoints such as Henry[12]. The second is end-user applications that use Semantic Web technologies to enhance user experiences such as Sonic Visualizer[13] and Sonic Annotator [14]. Moreover, Kolozali et al. [20] investigated the musical instruments' knowledge representation issues on the Semantic Web and found that many classification schemas represent ill-defined semantics.

Raimond and Sandler [21] proposed a framework for evaluating and comparing the expressiveness of the Music Ontology. It based on the data-driven and task-based ontology evaluation methodologies, with other music-related frameworks. The comparison results show that approximately 70% of the information in a dataset, results from queries done by the casual user, is expressible in the Music Ontology Framework. They found that any system that utilizes the Music Ontology framework could answer the user queries appropriately. They also found that the Music Ontology framework outperforms other music representation frameworks in terms of lexical comparison.

Many platforms developed to deal with Semantic Web linked data to facilitate the query process for non-expert users. They can use the SINA keyword search framework [22] to answer questions about interlinked data. Besides, the music artists' data published can be linked and discovered using the MusicWeb platform [23]. Its API integrates various LOD resources with Semantic Web ontologies, based on semantic metadata, to process and link artists' information.

Recently, some Semantic Web tools developed. Rodriguez-Muro and Rezk [24] apply an efficient approach for SPARQL to SQL queries conversion. Kyzirakos et al. [25] proposed an Open Source tool, GeoTriples, that generates and processes the extended R2RML and RML mappings that transform the geospatial data in both relational databases and raw files into RDF graphs. Musto et al. [26] Proposed a framework used the descriptive properties publicly available in the Linked Open Data (LOD) cloud for generating natural language explanations for the suggestions generated by recommendation algorithms. These properties used to build a graph that connects the recommendations the user received to the previously liked items. Andjelkovic et al. [27] introduced, Moodplay, a music-artists interactive discovery and recommendation system. It is a mood-based system that enables the exploration and discovery of new artists through an interactive mood space. It has a visualization interface in which the information linked to the artist's data in the last.fm.

3. PROPOSED SYSTEM

First, each table in a database is represented by a mapping that consists of several triples maps where each one corresponds to a logical table and acts as a general rule to transform each row of the logical table into a collection of RDF Triples. Next, a triples map consists of a logical table, a subject map, and multiple predicate-object maps. The logical table could be either a base table or a SQL view or an R2RML view. The R2RML view is a valid SQL query that does not modify the database with an optional schema and/or owner and/or SQL version that indicates the specific version of SQL that used. A triples map must contain exactly a single logical table. The subject map defines a rule used for generating subjects of RDF triples that will be created by the triples map, and it will use for creating several triples. A triples map must contain exactly a single subject map that determines how each row in the logical table generated. A triples map may comprise a set of predicate and object map pairs. Each predicate-object map pair defines the rules for creating both predicate and object maps of RDF triples. Finally, for each row in the logical table, these predicate and object map pairs together with the subject map will be used for generating one or more RDF triples. The proposed system software architecture diagram is shown in Figure 1, which shows a two-phase ontology-based system.

1. **Customize and generate R2RML mapping:** is the first phase that produces a custom

(11) *DBTune server*. Available: http://dbtune.org/ (accessed 08 July 2020).
(12) *Henry*. Availablehttp://code.google.com/p/km-rdf/wiki/Henry/ (accessed 08 July 2020).
(13) *Sonic Visualizer*. Available: http://sonicvisualiser.org/ (accessed 08 July 2020).
(14) *Sonic Annotator*. Available: http://omras2.org/SonicAnnotator (accessed 08 July 2020).

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

R2RML mapping document based on the targeted user-selected ontology that later may provide as an input for a dumping engine or any other R2RML processor to create the RDF triples. After delivering the database connection, the base URI and other namespaces URIs, firstly, he loads the database tables and can optionally explore columns with their data types, then passes recursively through a series of screens or wizards for adding triple maps. A triples map consists of a logical table, subject map, and multiple predicate-object maps where each one is specified using a separate wizard. The logical table wizard is used to create a new triples map, edit or delete an existing one that is named by a table or a view name and an ascending number for ease of identification. The subject map wizard used for adding the subject map that could be an IRI or a blank node, often generated using the primary fundamental values, to the selected triple. The predicate-object map wizard used for adding, editing, or deleting multiple predicate-object maps to the selected triples map. The constructed document displayed after each step showing how correctly the user is performing. The resulting final output is an RDF of the R2RML mapping called R2RML mapping graph written in the Turtle RDF recommended syntax (.ttl) and is used to describe the relationship between the relational database and ontology.

2. **Dumping Engine:** is the second phase that accepts the resulting R2RML mapping document generated from the first phase in Turtle RDF syntax (.ttl) [28]. It also agrees with the database connection configuration and the desired RDF syntax as inputs. It-dumps the RDF datasets in nearly all formats like Turtle, N Triples, Notation 3, NQuads, TriG, etc. this phase produces or materializes the RDF triples graph based on the specified properties.

The system also supports the following additional features:

1. Generate the DM document that could be customized to achieve the desired mapping.
2. Perform automatic mapping by generating all the triples from the selected database.
3. Provide a SQL Editor for running SQL queries against the selected database using a fast, full-featured, and colored query editor.
4. Provide a SPARQL Editor for importing datasets either from URI or an RDF document, writing, loading or saving SPARQL (.rq) queries using a query editor and running them against the imported dataset.
5. Convert, validate, and export various RDF formats. First, provide an RDF document and choose whether to convert, validate, or export it into another one of the RDF formats. Those formats include (Turtle – NTriples – NQuads – GZipped NQuads – Notation 3 – TriG – GZipped TriG– Trix – Gzipped Trix – CSV – TSV).

For some features, the user has to provide the base URI and the database connection configuration, which provided through a wizard displaying the data source, server name, authentication type with optional username, and password, including testing the connection and selecting the desired database name. The main GUI of the system is shown in Figure 2

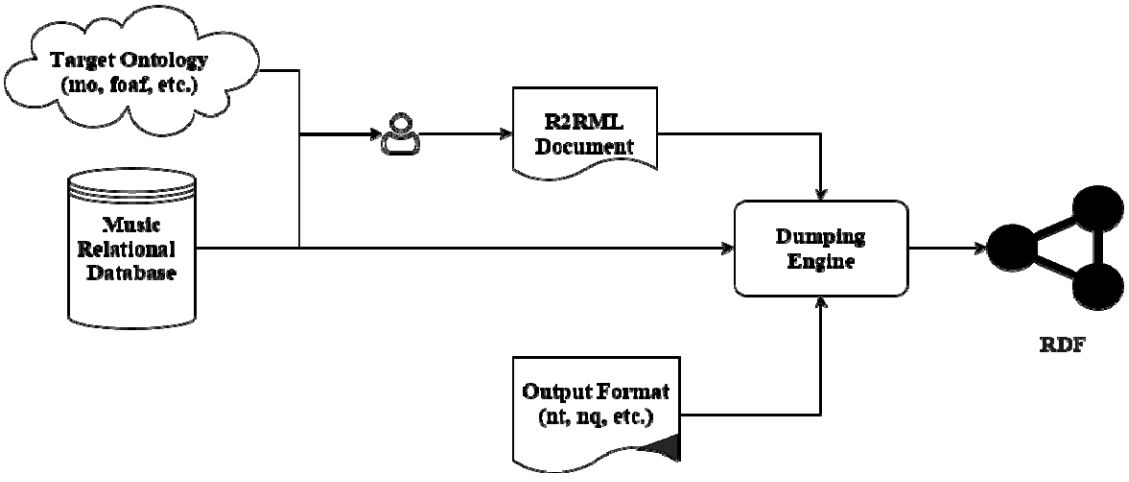

Figure 1. The Ontology-Based System Software Architecture Diagram

Journal of Theoretical and Applied Information Technology
30th November 2020. Vol.98. No 22

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

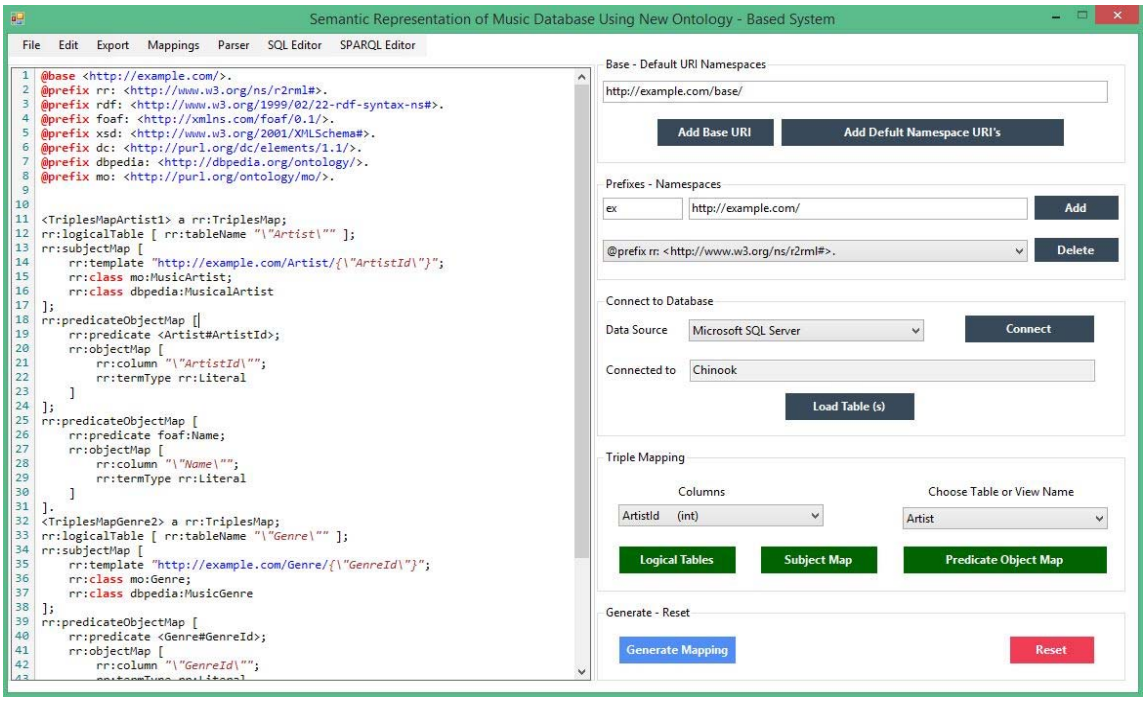

Figure 2. Main GUI of The Ontology-Based System

Figure 2 shows the main GUI of the proposed system. It is used to customize the mapping of the chinook database attached to the SQL server. The main wizard primarily consists of two panels. The first panel is a colored rich text-editor, while the second one is for setting the input parameters for the mapping. This panel includes various sections for the base URI, the prefixes and namespaces, the database connection configurations with the loading of the existing tables and views, the triple mapping, and the last to generate mapping and reset all the inputs.

1. Add the base URI and optionally add some default namespace URI's.
2. Add or delete other prefixes and their relative namespaces.
3. Choose the data source which initiates a wizard for providing privileges to connect to a specific database, test connection, and choose one of the listed databases.
4. Load the tables and views and optionally see their columns.
5. Iteratively choose the table or the view and start specifying the logical table. Using the logical table wizard, one can either create a new triples map or select, modify or delete an existing one created for a table or SQL view or R2RML view. Using the subject map wizard, one can select the desired map and add multiple classes, constants, columns, templates, etc. to it as specified by the R2RML standard. Using the predicate object map wizard, one can select the desired map and add, modify, or delete multiple predicate and object maps to it. After each wizard in the iterative step, a preview of the triples map is displayed.

4. EXPERIMENTS

The proposed system implemented using C# under the. NET Framework 4.6 and runs on a personal laptop with the following specifications: Windows 8.1 Pro 64-bit OS, Intel (R) Core (TM) i7-4500U CPU @ 1.80GHZ 2.40GHZ, 8 GB RAM and one terabyte disk space. The proposed system tested against all DM and R2RML compliance test cases [8], and it passes all of them (24 test cases for DM and 62 test cases for R2RML). The Chinook database is a model representing a digital media store that includes tables for Artists, Albums, etc.

The public database, Sakila, is an open-source musical database used in representing a DVD rental store in a normalized schema and contains tables for Films and Actors. This additional database utilized in calculating the running time of dumping data. Both databases, Chinook and Sakila, are used to test the performance of the proposed system in SQL Server, My SQL, and SQLite DBMS with the scheme in Table 1.

Journal of Theoretical and Applied Information Technology
30th November 2020. Vol.98. No 22

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table 1. The schema for two accessible relational databases

Database	Views	Tables	Columns	Rows
Chinook	0	11	64	15607
Sakila	7	16	131	46273

Not all database tables' data published on the web. The only data that published are those that can be made public. The chinook music database contains some tables representing music data such as Artist, Album, Track, Genre, and MediaType tables. While other tables are representing personal related data such as Customer, Employee, Invoice, Invoiceline, Playlist, PlaylistTrack, the music-related tables' data can be publicly published and linked with other music data on the web. The direct mapping results in converting all data in all tables into RDF.

One can generate the Direct Mapping document as shown in Figure 3. Then, the dumping engine is used to generate the RDF triples as shown in Figure 4. Similarly, one can generate the custom mapping document as shown in Figure 5 and use it to generate the RDF triples as shown in Figure 6.

```
@base <http://example.com/>.
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix xsd: <http://www.w3.org/2001/XMLSchema#>.
@prefix rr: <http://www.w3.org/ns/r2rml#>.
@prefix foaf: <http://xmlns.com/foaf/0.1/>.

<http://example.com/ArtistTriplesMap> a rr:TriplesMap;
                                        rr:logicalTable [rr:tableName "Artist"];
                                        rr:predicateObjectMap [rr:predicateMap [rr:constant <http://example.com/Artist#ArtistId>] ;
                                                               rr:objectMap [rr:column "ArtistId" ;
                                                                             rr:datatype xsd:integer]],
                                                              [rr:predicateMap [rr:constant <http://example.com/Artist#Name>] ;
                                                               rr:objectMap [rr:column "Name"]];
                                        rr:subjectMap [rr:class <http://example.com/Artist> ;
                                                       rr:template """http://example.com/Artist/ArtistId={\"ArtistId\"}"""].
<http://example.com/GenreTriplesMap> a rr:TriplesMap;
                                      rr:logicalTable [rr:tableName "Genre"];
                                      rr:predicateObjectMap [rr:predicateMap [rr:constant <http://example.com/Genre#GenreId>] ;
                                                             rr:objectMap [rr:column "GenreId" ;
                                                                           rr:datatype xsd:integer]],
                                                            [rr:predicateMap [rr:constant <http://example.com/Genre#Name>] ;
                                                             rr:objectMap [rr:column "Name"]];
                                      rr:subjectMap [rr:class <http://example.com/Genre> ;
                                                     rr:template """http://example.com/Genre/GenreId={\"GenreId\"}"""].
```

Figure 3. Sample of the Direct Mapping Document

```
<http://example.com/Artist/ArtistId=3> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://example.com/Artist> .
<http://example.com/Artist/ArtistId=3> <http://example.com/Artist#ArtistId> "3"^^<http://www.w3.org/2001/XMLSchema#integer> .
<http://example.com/Artist/ArtistId=3> <http://example.com/Artist#Name> "Aerosmith" .
<http://example.com/Genre/GenreId=1> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://example.com/Genre> .
<http://example.com/Genre/GenreId=1> <http://example.com/Genre#GenreId> "1"^^<http://www.w3.org/2001/XMLSchema#integer> .
<http://example.com/Genre/GenreId=1> <http://example.com/Genre#Name> "Rock" .
```

Figure 4. Sample of RDF Triples Resulting From the Direct Mapping

Figure 3 shows a sample of the Direct Mapping document generated for the Chinook database. It is clearly shown that Direct Mapping only defines simple transformations and provides the basis for the definition and comparing more complex transformations. It follows some simple rules that map table to class, column to property, row to resource, cell to a literal value, besides, cell to URI if there is a foreign key constraint. The Base URI for the whole graph/dataset defined in the first line of the document.

Figure 4 shows a sample of the result of generating the RDF triples graph using the Direct Mapping document. This represents the conversion of the chinook relational database into RDF, by making explicit the semantics encoded in the relational schema. This results in a set of triples that have a common subject formed by concatenating the base IRI, table name, primary key column name, and primary key value. The predicate for each column is represented by an IRI formed by concatenating the base IRI, table name, and column name. The values are literals formed from the column value. In the case of a foreign key, it produces a triple with a predicate formed by concatenating the foreign key column names, the referenced table, and the referenced column names. References to rows in tables without a primary key are expressed by triples with blank nodes for objects, where that blank node is the same node used for the subject in the referenced row. The materialized RDF graph can be queried by SPARQL or traversed by an RDF graph API.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

```
@base <http://example.com/>.
@prefix rr: <http://www.w3.org/ns/r2rml#>.
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix foaf: <http://xmlns.com/foaf/0.1/>.
@prefix xsd: <http://www.w3.org/2001/XMLSchema#>.
@prefix dc: <http://purl.org/dc/elements/1.1/>.
@prefix dbpedia: <http://dbpedia.org/ontology/>.
@prefix mo: <http://purl.org/ontology/mo/>.

<TriplesMapArtist1> a rr:TriplesMap;
rr:logicalTable [ rr:tableName "\"Artist\"" ];
rr:subjectMap [
    rr:template "http://example.com/Artist/{\"ArtistId\"}";
    rr:class mo:MusicArtist;
    rr:class dbpedia:MusicalArtist
];
rr:predicateObjectMap [
    rr:predicate <Artist#ArtistId>;
    rr:objectMap [
        rr:column "\"ArtistId\"";
        rr:termType rr:Literal
    ]
];
rr:predicateObjectMap [
    rr:predicate foaf:Name;
    rr:objectMap [
        rr:column "\"Name\"";
        rr:termType rr:Literal
    ]
].
<TriplesMapGenre2> a rr:TriplesMap;
rr:logicalTable [ rr:tableName "\"Genre\"" ];
rr:subjectMap [
    rr:template "http://example.com/Genre/{\"GenreId\"}";
    rr:class mo:Genre;
    rr:class dbpedia:MusicGenre
```

Figure 5. Sample of the Custom Mapping Document

```
<http://example.com/Artist/3> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://purl.org/ontology/mo/MusicArtist> .
<http://example.com/Artist/3> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://dbpedia.org/ontology/MusicalArtist> .
<http://example.com/Artist/3> <http://example.com/Artist#ArtistId> "3"^^<http://www.w3.org/2001/XMLSchema#integer> .
<http://example.com/Artist/3> <http://xmlns.com/foaf/0.1/Name> "Aerosmith" .
<http://example.com/Genre/1> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://purl.org/ontology/mo/Genre> .
<http://example.com/Genre/1> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://dbpedia.org/ontology/MusicGenre> .
<http://example.com/Genre/1> <http://example.com/Genre#GenreId> "1"^^<http://www.w3.org/2001/XMLSchema#integer> .
<http://example.com/Genre/1> <http://purl.org/dc/elements/1.1/title> "Rock" .
```

Figure 6. Sample of RDF Triples Resulting From Custom Mapping

Figure 5 shows a sample of the custom mapping document generated for the Chinook database. Each logical table - database table, database view, or an SQL query - is mapped to a set of triples by a rule called triples map. There are two parts for every triple map: a subject map and several predicate-object maps (combining predicate and object maps). It shows the custom patterns for the subjects URI's, adds existing additional ontology to the subjects, uses predefined predicates, etc.

Figure 6 shows a sample of the result of generating the RDF triples graph using the custom mapping document. The triples are formed using subject maps, predicate maps, and object maps. By default, all RDF triples are placed in the default dataset graph. A triples map can place some or all of the triples into named graphs instead by adding graph maps in the custom document. The proposed system also contains a processor for which, given the mapping and an input database, provides access to the output dataset.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

5. RESULTS AND DISCUSSION

The output results in Table 2 represent the average of running the proposed system ten times for each database on each DBMS. It shows the average time for generating the musical database mapping document directly. The average time for creating the triples using the resulting direct mapping document. The average time for automating the entire process for both of them with a single click. The size of the resulting Turtle (.ttl) document generated by Direct Mapping. The count of the triples in that document. The size of the resulting RDF document in (.nq) format. Finally, the count of the RDF triples in that document. The average automatic triples generation time displayed in Figure 7, which shows that the time is slightly different. The query execution time is somewhat different from one DBMS to the other. For Direct Mapping, the number of triples generated from any table calculated using equation (1) and the total number of triples in the database calculated using equation (2). For custom mapping, the number of generated triples depends on the same equations in addition to the number of rr:class statements in the subject map, the number of predicate maps in the predicate-object maps, the number of graph maps in both subject and predicate-object maps, etc.

Table 2. Performance Of The Proposed System Against Two Accessible Databases

Database	Driver	Chinook	Sakila
Average Direct Mapping Time (ms)	SQL Server	99.8000	3144.90
	My SQL	2471.80	3498.90
	SQLite	6.10000	11.1000
	Average	859.200	2218.30
Average Triples Generation Time (ms)	SQL Server	16847.4	71979.6
	My SQL	16996.8	72930.1
	SQLite	16232.9	74134.5
	Average	16692.4	73014.7
Average Automatic Mapping Time (ms)	SQL Server	16947.8	75124.9
	My SQL	19469.0	76429.4
	SQLite	16239.6	74146.3
	Average	17552.1	75233.5
Direct Mapping Document Size[.ttl] (KB)		26.8	41.6
Direct Mapping RDF Triples Count		471	713
Automatic Mapping Document Size[.nq] (MB)		16.2	65.2
Automatic Mapping RDF Triples Count		113951	455444

$$T = (C + Ref + 1) * R - (N + NRef) \quad (1)$$

Where T: number of triples in a table, C number of columns, Ref: number of references, R: number of rows, N: number of null values, NRef: number of null references in a table

$$D = \sum_{k=1}^{n} (C_k + Ref_k + 1) * R_k - (N_k + NRef_k) \quad (2)$$

Where D: number of triples in a database, n is the number of tables, Ck: number of columns, Ref k: number of references, Rk: number of rows, Nk: number of null values, NRefk: number of null references in table k. The + 1 indicating the additional triple generated for each row showing the type of table. The proposed system offers a simple GUI for users and supports a variety of database drivers. This enriching the semantic web with a new tool that facilitates the extraction process of data from databases

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

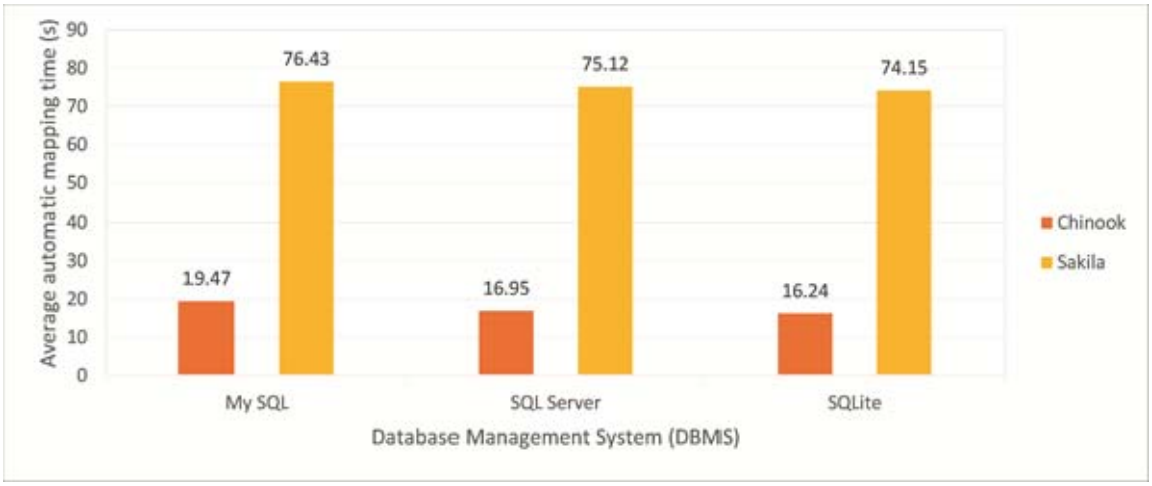

Figure 7. The Average Automatic Mapping Time In Seconds For Chinook And Sakila Databases

6. CONCLUSION AND FUTURE WORK

In this paper, a new ontology-based software devised and used for converting the music relational databases into RDF triples dataset using a set of steps for both experts and semi-experts. This system facilitates the conversion process using a set of GUI wizards through which the user can select a specific database table and define its mapping. The conversion process could be directly executed or customized according to the desired targeted ontology. After that, the dumping process used to generate RDF. The resulting RDF could be utilized by any SPARQL endpoint to run queries and extract meaningful information. This method is useful for the Semantic Web contributors as they could map and integrate their relational data, chosen to be public, with other linked data on the web. The system is applied to a specific musical database and could apply to any other database. Also, the system proved to be well-performing against multiple RDBMS, such as SQLite, MySQL, and SQL Server. They are automating the process of both generating, publishing, and interlinking RDF target as a future work to be accomplished as a scheduled or manual task.

REFERENCES:

[1] N. Shadbolt, T. Berners-Lee, and W. Hall, "The semantic web revisited," IEEE intelligent systems, vol. 21, pp. 96-101, 2006.

[2] F. Manola, E. Miller, and B. McBride, "RDF 1.1 primer, W3C working group note," 2014.

[3] E. Prud'hommeaux and A. Seaborne, "SPARQL query language for RDF, W3C Candidate Recommendation (2006)," http://www.w3.org/TR/rdf-sparql-query/, 2007.

[4] B. He, M. Patel, Z. Zhang, and K. C.-C. Chang, "Accessing the deep web: A survey," Communications of the ACM, vol. 50, pp. 94-101, 2007.

[5] A. Marcelo, B. Alexandre, P. h. Eric, and S. Juan. (2012, 27 september 2012). A Direct mapping of Relational Data to RDF. Available: https://www.w3.org/TR/rdb-direct-mapping/

[6] D. Souripriya, S. Seema, and C. Richard (27 september 2012). R2RML: RDB to RDF Mapping Language. Available: https://www.w3.org/TR/r2rml/

[7] C. Bizer and R. Cyganiak, "The D2RQ Plattform," W3C Workshop on RDF Access to Relational Databases 25-26 October, 2007 — Boston, MA, USA D2RQ Lessons, 2009.

[8] M. H. Boris Villazón-Terrazas. R2RML and Direct Mapping Test Cases. Available: https://www.w3.org/TR/rdb2rdf-test-cases/

[9] M. H. Boris Villazón-Terrazas. (14 August 2012). RDB2RDF Implementation Report. Available: https://www.w3.org/TR/rdb2rdf-implementations/

[10] J. F. Sequeda and D. P. Miranker, "Ultrawrap: SPARQL execution on relational data," Journal of Web Semantics, vol. 22, pp. 19-39, 2013.

[11] Y. Raimond, S. A. Abdallah, M. Sandler, and M. Lalmas, "A scalable framework for multimedia knowledge management," in

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

International Conference on Semantic and Digital Media Technologies, 2006, pp. 11-25.

[12] C. Bizer, T. Heath, D. Ayers, and Y. Raimond, "Interlinking open data on the web," in Demonstrations track, 4th european semantic web conference, innsbruck, austria, 2007.

[13] Y. Raimond, S. A. Abdallah, M. B. Sandler, and F. Giasson, "The Music Ontology," in ISMIR, 2007, p. 8th.

[14] D. Brickley and L. Miller, "FOAF vocabulary specification 0.91," ed: Citeseer, 2007.

[15] Y. Raimond and S. Abdallah, "The timeline ontology," OWL-DL ontology, 2006.

[16] Y. Raimond and S. Abdallah, "The event ontology," 2007.

[17] A. Passant and Y. Raimond, "Combining Social Music and Semantic Web for Music-related Recommender Systems," 2008.

[18] Y. Raimond, M. B. Sandler, and Q. Mary, "A Web of Musical Information," in ISMIR, 2008, pp. 263-268.

[19] G. Fazekas, Y. Raimond, K. Jacobson, and M. Sandler, "An overview of semantic web activities in the OMRAS2 project," Journal of New Music Research, vol. 39, pp. 295-311, 2010.

[20] S. Kolozali, M. Barthet, G. Fazekas, and M. B. Sandler, "Knowledge Representation Issues in Musical Instrument Ontology Design," in ISMIR, 2011, pp. 465-470.

[21] Y. Raimond and M. Sandler, "Evaluation of the music ontology framework," in Extended Semantic Web Conference, 2012, pp. 255-269.

[22] S. Shekarpour, E. Marx, A.-C. N. Ngomo, and S. Auer, "Sina: Semantic interpretation of user queries for question answering on interlinked data," Journal of Web Semantics, vol. 30, pp. 39-51, 2015.

[23] A. Allik, M. Mora-Mcginity, G. Fazekas, and M. B. Sandler, "MusicWeb: Music Discovery with Open Linked Semantic Metadata," in International Semantic Web Conference (Posters & Demos), 2016.

[24] M. Rodriguez-Muro and M. Rezk, "Efficient SPARQL-to-SQL with R2RML mappings," Journal of Web Semantics, vol. 33, pp. 141-169, 2015.

[25] K. Kyzirakos, D. Savva, I. Vlachopoulos, A. Vasileiou, N. Karalis, M. Koubarakis, et al., "GeoTriples: Transforming geospatial data into RDF graphs using R2RML and RML mappings," Journal of Web Semantics, vol. 52, pp. 16-32, 2018.

[26] C. Musto, F. Narducci, P. Lops, M. de Gemmis, and G. Semeraro, "Linked open data-based explanations for transparent recommender systems," International Journal of Human-Computer Studies, vol. 121, pp. 93-107, 2019.

[27] I. Andjelkovic, D. Parra, and J. O'Donovan, "Moodplay: interactive music recommendation based on artists' mood similarity," International Journal of Human-Computer Studies, vol. 121, pp. 142-159, 2019.

[28] D. Beckett, T. Berners-Lee, E. Prud'hommeaux, and G. Carothers, "RDF 1.1 Turtle," World Wide Web Consortium, 2014.