Generating RDF Documents through Relational Databases

Kyawt Kyawt San
University of Computer Studies, Yangon
kyawtkyawts@gmail.com

Khin Nweni Tun
University of Computer Studies, Yangon
knntun@gmail.com

Abstract—the Semantic Web has been developed in order to publish and access a lot of information for automatically processing of data and information by machines rather than for people. The relational data model is the major sources of information for future development of Semantic Web. RDF (Resource Description Framework) data model is a proven technology for generating semantic data from existing relational data representation. The certain loss of semantics has occurred in the generation of relational data to RDF. This paper paid much attention to extracting not only explicit but implicit semantics from RDB. To accomplish this goal, the correspondences between these two relational data models are studied and an algorithm for generating RDF from relational databases with integrity constraints preservation is proposed. In order to solve an ambiguous situation of understanding the same term of different meanings, URI (Uniform Resource Identifier) for each specific instance is used. The proposed mapping solution extracts relational schema and data from an existing relational database. Then, relation URI, attribute URI and tuple URI are generated. Finally, a series of triples are generated from those URI and validated against RDF validator.

Keywords—*RDF, Semantic Web, URI*

I. INTRODUCTION

One of the principles of the Semantic Web is to extend the existing web of documents to web of data. Those data should be represented in machine-readable format. There are many formalized semantic web languages to represent web data such as RDF, DAML, and OWL. The RDF is a W3C recommendation that represents current web into machine understandable format. To contribute to creating the Semantic Web data, generating RDF data from relational databases plays a major important role.

RDF is a fundamental data model for representing any kind of information available on the Web. It is based on three elements: resources, properties and statements. The subject is a resource, the thing to describe, identified by an URI. The properties are a special kind of resource that describes relations between resources. They are also identified by URIs. A specific resource together with a named property needs an object, in order to construct a statement. The object can be either a resource or an atomic value, named literal. Being composed of three parts, RDF statements are also known as triples.

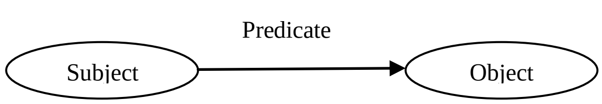

Figure1. General Form of an RDF statement

In RDF data modeling, URIs are used for describing meta data and things, also called resources, which could represent people, places, documents, images, databases, etc. In this paper, URIs is used to give RDF triples the unique and consistent identifiers. This paper proposes an approach of mapping relational database instances to RDF representation format by utilizing the advantages of URIs. There are two main approaches for solving exposing relational data in RDF representation: (1) A relational database is queried by an RDF query language. Mapping is created between relational database schema and predefined ontology and based on this mapping, relational data can be queried by semantic queries. A result of the query is a set of RDF statements. (2) Relational database content is transformed into RDF. Resulting RDF can be stored in static RDF documents or in a native RDF database.

The latter approach is adopted to produce an RDF document. This approach takes into account the relationships between tables, and the implicit semantics of relational constraints during the conversion process. A brief introduction about semantic web technologies is given in this section. The remains of this paper are organized as follows: Section II presents the existing works related to the RDB to RDF mapping approaches and Section III describes the proposed system architecture. Section IV discusses about the implementation results. Finally, Section V concludes the paper.

II. CURRENT SOLUTIONS OF MAPPING RDB TO RDF

One of the challenges of Semantic Web is to utilize relational data sources in RDF format. There has been a lot of effort on mapping relational data to relatively newer technologies, RDF. Krishna [11] proposed a methodology for representing an ER diagram in RDF, this method maps relational data to an RDF format with the extensive use of user-defined URIref vocabularies. Farouk [9] presented an approach for converting DB to RDF with additional defined rules. It focused on adding extra knowledge (user-defined rules) during mapping process.

Most of the existing methods are semi-automatic or manual and require much user effort. Single-column primary keys, foreign keys and relationships are assumed only to be 1:1. The task of extracting semantics from the data stored in the classic relational databases (RDB) is one of the major problems in Semantic Web. It can be addressed by the RDF database technology acting as a gateway to allow relational databases to expose their data semantics to the World Wide Web. In addition, we solved the limitations mentioned above with the aid of Semantic Web technology, RDF.

III. Generation of RDF Data with Relational Constraints Preservation

We can classify the mapping approaches used to map relational data to RDF as either **direct conversion** or mapping from domain ontology back to the underlying relational database. A mapping is the specification of a mechanism for transforming the elements of a model conforming to a particular meta model into elements of another model that conforms to another (possibly the same) meta model.

The proposed RDB to RDF mapping algorithm is shown in Figure 3. The proposed algorithm extracts implicit semantics from relational databases in order to produce an RDF document that reflects the same structure as that of the database. To maintain data integrity between relational data and generated RDF triples is the main goal of the proposed system.

Figure 2 shows the proposed system architecture for converting data from RDB to RDF with extracted semantics. The proposed system architecture can be broadly classified as follows: The first task is extracting relational schema, its integrity constraints and relational data. The second task is to generate URI for relational data and then concatenation of the generated URI to form RDF triples. During the URI generation process, the proposed system encodes the relational constraints including composite keys, avoiding blank nodes, handling null fields etc. Hence, a method which allows automatic extraction of data from RDB and their restructuring in the form of RDF graphs to make them available for the Semantic Web is proposed. In order to accomplish the conversion process, the correspondence between relational data model and RDF data model is taken below:

Relation Name ⟷ Subject

Field Name ⟷ Predicate

Value ⟷ Object

The last task is to generate RDF document generated by the proposed mapping algorithm. And then validate and transform to any RDF format. The proposed approach generates mapping relational data to RDF and generates RDF document. This approach aims to develop an easy and efficient generation of RDF document. In order to accomplish this goal, we studied the correspondences between the relational data model and RDF data model.

Definition 1: RDF is a

- Triple data model: **<subject, predicate, object>**
- **Subject**: is anything that can have a URI, such as "http://www.w3schools.com/RDF"
- **Predicate**: is a Resource that has a name, such as "author" or "homepage"
- **Object**: value (of the predicate), literal or blank node

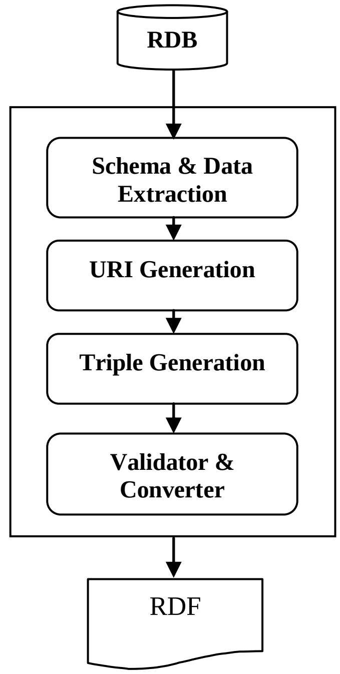

Figure2. The Proposed System Architecture for Generating RDF Document

Definition 2: A relational model consists of:

- Rel(r): r is a relation name
- Attr(a, r): a is an attribute of relation r
- PK_n (a_1,. . . , a_n, r): (a_1, . . . , a_n) ($n \geq 1$) is a primary key in r
- FK_n (a1, . . . , a_n, r , b_1, . . . , b_n, s): (a_1, . . . , a_n) ($n \geq 1$) is a foreign key in relation r that references to (b_1, . . . , b_n) in relation s
- Value (v, a, t, r): **v** is the value of attribute **a** in a tuple with identifier **t** in relation **r.**
- a function $Concat_n$(s_1, . . . ,s_n, s) holds if s is the concatenation of the strings s_1, . . ., s_n.

Each table is characterized by its name, the list of its attributes and a list of its records. Several restrictions have been incorporated over the years in the initial relational model. The most prominent ones are the primary and foreign key constraints. The primary key constraint indicates that each table has a column that uniquely identifies every row of the table. This column is the primary key of the table and contains distinct values. The foreign key constraint states that a column of a table is restricted to only values from the primary key of another table.

A. *Proposed Algorithm for Generating RDF*

```
Algorithm Map2RDF
Input: Relational Schema and Database Instances
Output: An RDF Document
Begin
for each relation T_i of DB loop
{
    base← <http://www.ucsy.edu.mm/db>

    RelURI(T_i,Y)←Rel(T_i),
                  Concat(base, Rel(T_i),Y);
    If (T_i has primary Key)
    {  TupleID(A,T_i,Z)←Rel(T_i),PK(A_1,..., A_n,T_i),
                         Value (V_1, A_1, I, T_i),….,
                         Value(V_n, A_n, I, T_i )
                        Concat(base, Rel(T_i ),
                        "#", A_1, "=", V_1,
                          A_2, "=", V_2,….,
                          A_n, "=",V_n, Z)          }
   Else
   {
     TupleID(I,T_i,Z)← Rel(T_i),
                       Concat ( _:, Rel(T_i),- ,I,Z);

     TupleID(I,T_i,Z)← Rel(T_i),
                      Concat(base,Rel(T_i),- I,Z);
    }
  For each Tuple in T_i loop

 Triple(S, rdf:type,O) ←Rel(T_i),Value(V, A, I, T_i),
                        TupleID(I,T_i,S),
                        RelURI(T_i,O);

   for each attribute A_j in T_i loop
    {
     AttrURI(A_j,T_i,Z) ← Rel(T_i ),Attr(A_j, T_i ),
                        Concat(baseURI,Rel(T_i ),
                         "#", A_j, Z);

  If (A_i is Foreign Key) then
   {
      Triple(S,P,O)←FK_n(A_i,T_i, Bi,Q_i),
                     Value(V_i, A_i, I, T_i ),…,
                     Value (V_i, B_i, J, T_i),…,
                     TupleID(T_i,A, S),
                     AttrURI(A_i,T_i, P),
                      TupleID(J,Q_i,O);
     }
        Else
     {
      Triple(S, P,O) ←Rel(T_i),Value(O, A, I ,T_i),
                      TupleID(I,T_i, S),
                     AttrURI(A, T_i, P)
      }
    }
}

End.
```

Figure3. The Proposed Algorithm for Generating RDF Document

B. *Illustration*

In this section, we demonstrate generating URIs for database instances. To illustrate our proposed algorithm, sample database "***db***" is used as a running example. Consider for example the relational schema given below.

Professor(PID,PName,DOB,Address,DID, CID)
Student (SID,SName,DOB,Address)
Course (CourseID, CName,C_Code)
Department (ID, DName)
teaches (ProfDID, CID)

TABLE I. (A) PROFESSOR

PID	PName	DOB	Address	DID	CID
001	Davis	1977	Yangon	01	111
002	Sue	1976	Mandalay	02	114

TABLE II. DEPARTMENT

ID	DName
01	Information Science
02	Software

TABLE III. TEACHES

ProfID	CID
001	112
002	111

C. *Generating URIs*

The Web provides a general form of identifier, called the Uniform Resource Identifier (URI), for identifying (naming) resources on the Web. RDF takes this advantage and URI becomes an essential component of RDF graphs.

It is necessary to resolve any duplication, either by producing URIs based on fully qualified names of schema elements, or by producing them randomly. Every resource is identified by a Uniform Resource Identifier (URI). In the case of a Web page, the URI can be the Unified Resource Locator (URL) of the page. The URI does not necessarily enable the access via the Web to the resource; it simply has to unambiguously identify the resource. The use of Uniform Resource Identifiers (URI) for entities along with the ability to link them together using predicates enables RDF to effectively integrate data from multiple sources.

In the translation of relational data to RDF approaches, it should be generated for relations, attributes and tuples. The choice which entities to use for identifiers (e.g. convert primary keys to URIs) is not always obvious. Therefore, generating suitable URIs for the RDF "resources" becomes one of the critical issues. All URIs are generated by appending to a base URI. According to our proposed algorithm, relational URI, attribute URI and tuple URI are generated.

The name of the database is denoted as "*db*", "*base*" is the base URI, "*rel*" is the relation name, "*attr*" refers the attribute name, "***pk***" is primary key of a relation and "***pkval***" is the primary key value.

Relation URI is an URI formed from the concatenation of the base URI and table name.

TABLE IV. URI GENERATION SCHEME

DB Element	URI Template
Database	{base}/{db}/
Relation	{base}/{db}/{rel}/
Attribute	{base}/{db}/{rel}/#{attr}
Tuple	{base}/{db}/{rel}/{pk}.{pkval}

Assume given a base URI "***base***" for the relational database to be translated (e.g., "http://www.ucsy.edu.mm/db/"), then the proposed algorithm produces relation URI according to running example:

<http://www.ucsy.edu.mm/db/Professor>,
<http://www.ucsy.edu.mm/db/Student>, etc.

Then, **tuple URI** is generated for each relation having a primary key. Thus, given that the facts PK_1 *(PID, Professor)* and *VALUE("001", "PID", "id1", "Professor")* hold in our example, the URI *http://example.edu/db/Professor#PID=001* is the identifier for the tuple in table Professor with value *001* in the primary key. Then, according to the algorithm, table triple is produced:
("http://www.ucsy.edu.mm/db/Professor#PID=001", "rdf:type", "http:// www.ucsy.edu.mm /db/ Professor ").

URI for each attribute is generated before generating Reference triples. It is formed from the concatenation of the base URI, table name and the column name.
http://www.ucsy.edu.mm/db/Professor#PID, PName, DOB, Address, DID,CID, etc., is generated for each attribute in relation Professor.

For each attribute, the following literal triples are generated from relation Professor:

<http://www.ucsy.edu.mm/db/professor/PID=002>
<http://www.ucsy.edu.mm/db/professor#PName> "Sue".
<http://www.ucsy.edu.mm/db/professor/PID=002>
<http://www.ucsy.edu.mm/db/professor#DOB> "1976" . etc,.

Each row is turned into a series of triples with a common subject. We get the final RDF triples by one processing step. Once information is in RDF form, it becomes easy to process it.

D. Encoding Constraints in RDF

In this section, we show some of the encoded relational constraints in RDF.

(1)MultiColumns Primary Keys:Primary keys may also be composite. If, in the above example, the primary key for teaches were (ProfID, CourseID) instead of ID, *the identifier for the only row in this table would be the following triples:*

<http://www.ucsy.edu.mm/db/teaches/ProfID=001;CourseID =112> rdf:type <http://www.ucsy.edu.mm/db/teaches>.etc.,

(2)Foreign Keys: For those triples that store the references generated by foreign keys, Reference triples are generated as follows:

http://www.ucsy.edu.mm/db/Professor#PID=001,
http:// www.ucsy.edu.mm/db/Professor #DID,
http://www.ucsy.edu.mm/db/Department#ID=01

(3)N-ary Relationships: usually in cases where the primary key of a relation is composed of foreign keys to more than two other relations. In the example database, " ***teaches"*** *relation takes n-ary relation.*

(4)Handling of Null Fields: In general, these can be entirely exercised from the RDF graph, but schema knowledge is required, as it is possible in RDBs to assign meaning to the absence of a value. Relational databases typically operate with the Closed World Assumption (what is not stated in true is false), whereas in the Semantic Web World the open world assumption pertains (what is not stated is unknown). In this paper, for every allowable null value, the empty string is used.

(5) Avoiding BNodes: *Blank nodes are generated for the tables not having a primary key. But in our approach, blank nodes are avoided by concatenating the base URI, relation name and tuple identifier.*

IV. DISCUSSIONS AND RESULTS

Some advantages of the proposed system are an easy and automatic production of RDF document once the classic relational database is loaded into the system. Some tests have been done on an experimental set of standard relational databases. A prototype has been developed to show the proposed transformation system of RDF triples and the results shows the promising results. The mapping process is implemented using java and mysql. In order to complete the process, we first export relational data content. Then, assigning URI to relational data content is done by concatenating the relative URI to the base URI in accordance with the proposed algorithm. Each tuple becomes a series of triples with a common subject.

In our approach, all types of relationships between tables are considered. Blank nodes problems are also solved by assigning URI references to blank nodes. In addition, we overcome the converting problems of composite primary keys and foreign keys. Resulting RDF can be stored in static RDF documents or in a native RDF database.

During the mapping process, a corresponding RDF instances are automatically generated. Generated RDF documents are validated against RDF Validator and Converter and obtain the RDF triples. The validated RDF code is written in NTriples format. RDF code for entire example database is too long and we show some of the triples only.

<http://www.ucsy.edu.mm/db/department/ID=01> rdf:type
<http://www.ucsy.edu.mm/db/department>

<http://www.ucsy.edu.mm/db/department/ID=01> <http://www.ucsy.edu.mm/db/department#ID> "01".

<http://www.ucsy.edu.mm/db/department/ID=01> <http://www.ucsy.edu.mm/db/department#DName> "Information Science".

<http://www.ucsy.edu.mm/db/teaches/ProfID=001; CourseID=112> rdf:type <http://www.ucsy.edu.mm/db/teaches>

<http://www.ucsy.edu.mm/db/teaches/ProfID=001; CourseID=112> <http://www.ucsy.edu.mm/db/teaches#ProfID> "001".

<http://www.ucsy.edu.mm/db/teaches/ProfID=001; CourseID=112> <http://www.ucsy.edu.mm/db/teaches#CourseID> "112".

Table 3 illustrates the comparisons between existing methods (D2RQ, Virtuoso RDF View, and Relational.OWL) and the proposed method. Compared with some other existing methods, the proposed system generates semantic meta data automatically. RDB equivalent RDF triples are obtained by utilizing the proposed algorithms.

TABLE V. COMPARISON OF EXISTING METHODS AND THE PROPOSED METHOD

	D2RQ	Virtuoso RDF View	Relational.OWL	Proposed System
Data Source	RDB	RDB	RDB	RDB
ETL or Virtual	Both	Both	ETL	ETL
Data Sync:	Bidirectional	SPARQL	Static	Static
M:N Relations	True	True	False	True
Handling Null	False	False	False	True
Mapping Automat.	Manual	SemiAuto	Auto	Auto
Constraints	True	True	True	True
Blank Nodes	True	True	True	True
Data Types	True	True	True	True
Literal to URI	True	True	False	True

V. CONCLUSON

Mapping between RDB to RDF is very important and many researches investigate this point. However, so far, there is none full-automatic mapping tool of SQL to RDF, most of which is defined manually. Further, with RDF parsers being available easily today, data expressed in RDF can be parsed and processed easily by a machine for any desired purpose. The mapping described in this paper can be enhanced further so as to represent higher feature of integrity constraints. The most significant advantage is that RDF instances are generated automatically from RDB and user does not need to have knowledge of the relational schema. The experimental result demonstrates that the proposed system performs well and processes automatic generation. It can be obviously seen that mass generation of Semantic Web meta data is needed. The proposed system can be used where data is stored in a relational database and there is a need for generating RDF automatically without domain expert.

REFERENCES

[1] Bakkas. J and Bahaj, M., *Generating of RDF graph from a relational database using Jena API*, International Journal of Engineering and Technology", ISSN : 0975-4024 Vol 5 No 2 Apr-May 2013.

[2] Bakkas. J. and Bahaj,M., *Direct Migration Method of RDB to Ontology while Keeping Semantics*, International Journal of Computer Applications (0975 – 8887) Volume 65– No.3, March 2013.

[3] Byrne, K., *Populating the Semantic Web—Combining Text and Relational Databases as RDF Graphs*. 2008, University of Edinburgh: Edinburgh.J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[4] C.Lei, Y.Nansheng, *Publishing Linked Data from Relational Databases Using Traditional Views* , IEEE, 2010.

[5] Chen,L. and Nansheng, Y. , *Publishing Linked Data from Relational Databases Using Traditional Views* , IEEE, 2010.

[6] Christian Bizer, R.C., D2R Server – *Publishing Relational Databases on the Semantic Web*. 2006, Freie Universitat Berlin: Berlin.

[7] Christian Bizer, R.C. (2007) D2RQ — Lessons Learned. W3C.

[8] Cyganiak, R., *RDB2RDF mapping with D2RQ and D2R Server*. 2009.

[9] Farouk, M. and Ishizuka, M., *Converting DB to RDF with Additional Defined Rules*, 8th International Conferenceon Web Information Systems and Technologies, 2012.

[10] Gherabi,N. and Addakiri,K., *Mapping relational database into OWL Structure with data semantic preservation* , (IJCSIS) International Journal of Computer Science and Information Security, Vol.10, No.1, 2012.

[11] Krishna, M., *Retaining Semantics in Relational Databases by Mapping them to RDF*, Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology,2006.

[12] Kyawt Kyawt San, Khin Nweni Tun, *Automatically Generating RDF Instances from Relational Data Sources*, ICCA 2013.

[13] Kyawt Kyawt San, Khin Nweni Tun, *Formalization of Direct Mapping for Relational Data to Semantic Web Representation*, ICCA 2012.

[14] Levshin, D., *Mapping Relational Databases to the Semantic Web with Original Meaning*. Knowledge Science, Engineering and Management, 2009: p. 5-16.

[15] W3C Working Draft, *A Direct Mapping of Relational Data to RDF*, 29 May 2012, http://www.w3.org/TR/2012/WD-rdb-direct-mapping-20120529/.