Article

Semantics-Preserving RDB2RDF Data Transformation Using Hierarchical Direct Mapping

Hee-Gook Jun ¹ and Dong-Hyuk Im ^{2,*} ¹ Openub, Seoul 06097, Korea; hgjun@openub.com² School of Information Convergence, Kwangwoon University, Seoul 01890, Korea

* Correspondence: dhim@kw.ac.kr

Received: 22 September 2020; Accepted: 8 October 2020; Published: 12 October 2020

Abstract: Direct mapping is an automatic transformation method used to generate resource description framework (RDF) data from relational data. In the field of direct mapping, semantics preservation is critical to ensure that the mapping method outputs RDF data without information loss or incorrect semantic data generation. However, existing direct-mapping methods have problems that prevent semantics preservation in specific cases. For this reason, a mapping method is developed to perform a semantics-preserving transformation of relational databases (RDB) into RDF data without semantic information loss and to reduce the volume of incorrect RDF data. This research reviews cases that do not generate semantics-preserving results, and the corresponding problems into categories are arranged. This paper defines lemmas that represent the features of RDF data transformation to resolve those problems. Based on the lemmas, this work develops a hierarchical direct-mapping method to strictly abide by the definition of semantics preservation and to prevent semantic information loss, reducing the volume of incorrect RDF data generated. Experiments demonstrate the capability of the proposed method to perform semantics-preserving RDB2RDF data transformation, generating semantically accurate results. This work impacts future studies, which should involve the development of synchronization methods to achieve RDF data consistency when original RDB data are modified.

Keywords: hierarchical direct mapping; relational database; semantic web; web ontological language

1. Introduction

The transformation of relational databases (RDB) into resource-description-framework (RDF) data is a key information extraction method used to publish semantics web data [1–3]. In 1998, Tim Berners-Lee proposed the concept of mapping RDBs to the Semantic Web [4]. Since then, several approaches have been proposed to improve the mapping of RDBs to semantic data. Additionally, the World Wide Web Consortium (W3C) has organized working groups to help standardize the technologies used for transforming RDBs into RDF data.

Direct mapping is a representative mapping method recommended by the W3C to support the automatic mapping of relational data to semantic data [5]. Figure 1 illustrates an example of direct mapping, which defines mapping rules for transforming both relational schema and instance data into RDF data. In the field of direct mapping, researchers have long-studied effective automated processes that focus on semantics preservation. The semantics preservation of the direct mapping process reflects the relational integrity constraints within the mapping result [6,7]. Because integrity constraints define the semantics of a database, mapping with integrity constraints generates more semantically accurate results.

Figure 1. A simplified example of direct mapping. RDB represents relational database, RDF represents relational databases resource-description-framework.

Although the transformation of relational data using integrity constraints has been long-studied [8–16], several problems still occur during the direct-mapping processes under specific conditions (see Figure 2). If an attribute has two or more integrity constraints (e.g., *NotNull* and *Unique* in Figure 2a), the mapping result will output a single RDF graph structure that combines all of the integrity constraints (the subgraph rooted by *name* in Figure 2b). The mapping result preserves the semantics in view of the previous methods because relational data is transformed into RDF data. However, to date, no explicit transforming meta-information or well-designed hierarchical structure has been provided to trace the input relational data into the mapping result. RDF *NotNull* and *Unique* triples should have meta-information that reveals their *NotNull* and *Unique* constraints respectively, regarding the attribute name in table *Lecture*. Without this information, a new subgraph can be extracted from the merged RDF graph, which can then be misinterpreted, generating unintended constraints not found in the original data (see the primary key in Figure 2c). Thus, mapping methods based on weak definitions of semantics preservation can cause information loss and incorrect data generation. Therefore, to ensure the accuracy of the mapping process in all cases, a stringent definition is needed to quantify semantics preservation.

Figure 2. An example of problems encountered during a direct-mapping process.

This paper proposes a hierarchical direct-mapping algorithm that prevents the problem illustrated in Figure 2 and preserves the semantics based on strict logical rules. Mapping problems occur when a mapping method simply focuses on data-type transformations. To prevent this, the proposed method uses a hierarchical semantics vocabulary and advanced mapping rules to map without semantic information loss. This paper also defines an evaluation metric using the inverse-mapping phase. That is, a mapping method is said to preserve the semantics if the result of inverse-mapping is semantically identical to the original input data (Figure 3). Evaluation results confirm the effectiveness and accuracy of semantics preservation of the proposed mapping methods.

Figure 3. Overview of our direct-mapping method.

This paper makes the following contributions:

- Lemmas are defined to ensure semantics preservation and to demonstrate the soundness and completeness of direct mapping.
- Hierarchical mapping rules are defined based on lemmas to perform semantics-preserving RDB-to-RDF (RDB2RDF) transformations and to prevent the loss of semantics and incorrect semantic data generation problems.
- The scope of semantics preservation is extended, such that the inverse transformation of the output semantic data should be identical to the original input data.

The remainder of this paper is structured as follows. In the following section, the RDB2RDF mapping methods are briefly discussed. Section 3 presents preliminaries and problem descriptions of direct mapping. Section 4 describes the proposed mapping rules using logical definitions and the implementation of those rules in detail. Section 5 summarizes the results of our experiments. Finally, the conclusions and discussion of prospective future research are provided in Section 6.

2. Related Work

RDB2RDF is a mapping method that transforms relational data into semantic data represented by the RDF. The RDF data model [17] is a language that describes semantic information on the Semantic Web. The basic unit of RDF data is based on a graph structure (i.e., the triple: subject, property, and object) [18–20]. The RDF is a flexible and interoperable model used to publish information to the web relative to the relational data model. However, because ~70% of websites are backed up as RDBs [1], existing relational data must be used with the RDB2RDF methodology for the improvement of the Semantic Web.

RDB2RDF mapping approaches include mapping creation, mapping representation and accessibility, mapping implementation, query implementation, application domain, and data integration [21]. Mapping creation has been widely studied to improve the generation of mappings between relational data and semantic data, and it can be performed either automatically or manually [22] (see Table 1 for a list of previous mapping works). Domain semantics-driven mapping is a manual mapping method [23,24]. The W3C RDB2RDF Working Group has recommended the R2RML mapping language [25,26] for customizing mappings. Mapping tools, such as D2RQ [27,28], Virtuoso [29,30], Ultrawrap [31,32], etc. have also been provided to support manual mapping. On the other hand, direct mapping is an automatic mapping method that was published by the W3C RDB2RDF Working

Group in 2012 [5]. It uses RDB instances and schemas as inputs and automatically generates RDF semantic data. In the field of RDF data creation, some methods used to transform various types of data (e.g., heterogeneous data [33], object-oriented data [34], and the Web of Data [35]) to RDF data have been devised. The current paper, however, mainly focuses on direct mapping to manage large-scale data on the web.

Table 1. Classification of previously reported mapping methods.

Type	Method	Authors
Manual RDB2RDF (Domain semantics-driven mapping)	R2RML	Hert et al. (2011) RDB2RDF Working Group (2012)
	D2RQ	Bizer et al. (2004, 2009)
	Virtuoso	Blakeley (2007) Erling and Mikhailov (2009)
	Ultrawrap	Sequeda et al. (2009, 2013)
Automatic RDB2RDF (Direct mapping)	Augmented mapping using OWL	Sequeda et al. (2012) Buccella et al. (2004) Astrova et al. (2007) Li et al. (2005) Lim et al. (2015) Tirmizi et al. (2008) Jun et al. (2020)
	DB2OWL	Cullot et al. (2007)
	RDBToOnto	Cerbah (2008)
	OWLED	Shen (2006)

Further research has been conducted to obtain semantic data from relational data without information loss [6,8–16]. RDF schema (RDFS) and Web Ontology Language (OWL) are used to obtain a more accurate mapping for RDB2RDF transformation. The concept of RDF data can be modeled by RDFS or OWL in a manner similar to that used for defining the relational schema using SQL data definition language (DDL). Moreover, because OWL contains a more expressive semantic vocabulary, the mapping methods can better express the semantics of relational integrity constraints.

Sequeda et al. [7] proposed an augmented direct-mapping method that generates semantic data from the integrity constraints of the SQL DDL schema. Because integrity constraints define the semantics of the RDBs, the quality of augmented direct mapping depends on the transformation of the integrity constraints of the RDBs. DB2OWL [10] and RDBToOnto [11] also provided augmented direct mapping tools. However, they were restricted to supporting only referential integrity constraints. Lim et al. [13] used the OWL to process more rules and Jun et al. [14] proposed semantics-preserving optimization of mapping multi-column key constraints. However, their method still lacked support for use in the transformation of all integrity constraints of the relational SQL syntax. Moreover, this paper has observed that the problem of incorrect semantic data generation can occur in specific cases, as described in the next section.

3. Preliminaries

3.1. Direct Mapping

Developed in 2010 and recommended in 2012 by the W3C RDB2RDF Working Group, direct mapping is an automatic map-creation method that transforms relational input data, including schema data, into RDF graph data (i.e., a direct graph). Direct mapping can be viewed as a function of transforming relational data with integrity constraints to semantic data. Figure 4 provides an example of relational input data via the direct-mapping process. The table, “Product,” contains an

attribute, pId, as its primary key, an attribute name, and an attribute production as a foreign key that references a table called “Production.” Production contains an attribute, pCd, as its primary key and an attribute name. Figure 5 presents the result of the direct-mapping process regarding the input data shown in Figure 4. The output graph comprises a set of RDF triples. Suppose that the base URI of the output data is <http://idb.snu.ac.kr/example/>. The primary key attributes with the base URI are used to generate the subject resource. Two *Product* resources and two *Production* resources are generated. Predicates are generated from the attribute names of relational tables, and objects are generated from the attribute values.

Figure 4. An example input relational data of the direct mapping.

Figure 5. An example result of the direct mapping.

3.2. Semantics Preservation of Direct Mapping

Further research has been conducted on the improvement of direct mapping to reduce information loss and ensure semantics preservation. Semantics preservation is an important feature of direct mapping as the quality of direct mapping is heavily depends on the semantics preservation. Sequeda et al. [7] provided a theoretical definition of semantics preservation. In addition to the definition, this research provides a stricter definition to quantify semantics preservation and evaluate

the accuracy of the mapping methods. This paper defines the semantics preservation of mapping methods as follows:

Semantics preservation: Suppose X is a set of relational data, F is a RDB to RDF mapping function, and G is an RDF to RDB inverse-mapping function of F , if $|X| = |G(F(X))|$ and $|G(F(X)) - X| = 0$, then F is an ideal function that satisfies semantics preserving mappings (Figure 6).

Figure 6. Graphical view of semantics preservation of direct mapping.

3.3. Limitation of Direct Mapping for Semantics Preservation

This section defines the three challenging problems encountered during the direct-mapping process for semantic preservation. This paper observes loss of semantics (Problems 1 and 2) and incorrect semantic data generation (Problem 3), which may occur in specific cases. The problems and specific conditions in which the problems occur were found during studies of existing direct-mapping methods [8–16]. To overcome these drawbacks, the problems into three categories are organized.

Problem 1 illustrates the loss of information when relational tables are transformed into semantic data using an OWL class. Because the OWL class is associated with objects generated in a single semantic model structure, every semantic resource can be inferred from the OWL class. Therefore, additional methods are needed to indicate that the output data are particularly generated from the transformed relational table.

- Problem 1: Suppose $y_a = \text{Class}(x_a)$ is an RDB2RDF mapping rule for a relational table, where $x_a \in R$, R is a set of relational tables, $y_a \in C$, C is a set of OWL classes, and $x_b = \text{Class_Inverse}(y_b)$ is an RDF2RDB inverse-mapping rule of an OWL class, where $x_b \in X$, X is a set of results generated by $\text{Class_Inverse}(y_b)$, $y_b \in C$, and C is a set of OWL classes. However, x_b is not the same as x_a , because $R \subset X$.

Problem 2 illustrates another type of information loss that occurs when semantic resources reference other resources. Each referencing object, including relational attributes and binary relations, can be transformed into the same type (i.e., OWL object property). Thus, a method of distinguishing each referencing object is discussed and provided in Section 4.

- Problem 2: Suppose $y_a = \text{ObjProp}(x_a)$ is an RDB2RDF mapping rule for a relational table, where $x_a = \{x_{a1}, x_{a2}\}$, $x_{a1} \in B$, B is a set of binary relations, $x_{a2} \in F$, F is a set of foreign keys, $y_a \in O$, O is a set of OWL object properties, and $x_b = \text{ObjProp_Inverse}(y_b)$ is an RDF2RDB inverse-mapping rule of an OWL object property, where $x_b \in X$, X is a set of results generated by $\text{ObjProp_Inverse}(y_b)$, $y_b \in O$, and O is a set of OWL object properties. However, $\text{ObjProp_Inverse}()$ does not work as intended, because it has not been given any information to determine whether y_b is generated from x_{a1} or x_{a2} .

Problem 3 illustrates incorrect semantic data generation when integrity constraints are transformed without considering that every subgraph having a specific identical root node can be merged into a single graph.

- Problem 3: Assume that the mapping rules for integrity constraints of relational data are described in Figure 7. Here, the predicates on the right-hand side are used to verify the integrity

constraints. $\text{DefaultCondition}(p, v)$ is a function that assigns v as a default value of predicate p . $\text{CheckCondition}(p, c)$ is a function that assigns c as a check condition of predicate p , and the other predicates on the left-hand side are defined in the Appendix A.

$$\begin{aligned} & \text{FP}(p) \wedge \text{Card}(p, 1) \rightarrow \text{NotNull}(p) \\ & \text{FP}(p) \wedge \text{IFP}(p) \wedge \text{MinCard}(p, 0) \wedge \text{MaxCard}(p, 1) \rightarrow \text{Unique}(p) \\ & \text{FP}(p) \wedge \text{IFP}(p) \wedge \text{Card}(p, 1) \rightarrow \text{PK}(p) \\ & \text{FP}(p) \wedge \text{MinCard}(p, 0) \wedge \text{MaxCard}(p, 1) \rightarrow \text{FK}(p) \\ & \text{FP}(p) \wedge \text{MinCard}(p, 0) \wedge \text{MaxCard}(p, 1) \wedge \text{DefaultCondition}(p, v) \rightarrow \text{Default}(p, v) \\ & \text{FP}(p) \wedge \text{MinCard}(p, 0) \wedge \text{MaxCard}(p, 1) \wedge \text{CheckCondition}(p, c) \rightarrow \text{Check}(p, c) \end{aligned}$$

Figure 7. Simple mapping rules of integrity constraints.

The subset relationships can be inferred, as shown in Figure 8. However, semantic data generated by the above rules can be misinterpreted. For example, assume that a relational attribute, x , has integrity constraints, “primary key” and “check,” F is a mapping function that contains the rules of Figure 7, and G is an inverse mapping function of F . Then, the integrity constraints of $G(F(x))$ are “primary key,” “check,” “foreign key,” and “unique,” because $\text{FK}(p) \subseteq \text{Check}(p)$, and $\text{Unique}(p) \subseteq \text{PK}(p) \cup \text{Check}(p)$. Therefore, developing a method to avoid incorrect semantic data generation is a challenge associated with the semantics-preserving RDB2RDF transformation (the detailed example is provided in the Appendix B).

$\text{FK}(p) \subseteq \text{Unique}(p)$ $\text{FK}(p) \subseteq \text{Default}(p)$ $\text{FK}(p) \subseteq \text{Check}(p)$ $\text{PK}(p) \subseteq \text{NotNull}(p) \cup \text{Unique}(p)$	$\text{NotNull}(p) \subseteq \text{PK}(p)$ $\text{Unique}(p) \subseteq \text{PK}(p) \cup \text{FK}(p)$ $\text{Unique}(p) \subseteq \text{PK}(p) \cup \text{Default}(p)$ $\text{Unique}(p) \subseteq \text{PK}(p) \cup \text{Check}(p)$
---	---

Figure 8. Subset relationships by mapping rules of integrity constraints.

4. Hierarchical Mapping Rules

This section provides the hierarchical rules for learning general relational schemas and integrity constraints. Each rule is based on lemmas that are valid within the semantics domain. Then, this section then explains how the problems described in Section 3.3 can be prevented by using proposed rules via lemmas. This work uses predicate logic to define rules and add graphical examples for better understanding. Then, the hierarchically structured semantic vocabularies are provided in order to generate sound and precise semantic data. The relationships among the lemmas, rules, and problems are described in Figure 9 to clarify the concept of the rules.

Lemmas	Rules		Related data	Related problems
Lemma1	General mapping rules	Rule 1	Relational tables	Problem 1
Lemma2		Rule 2 Rule 3 Rule 4 Rule 5	Relational attributes Binary relations Identifying relationships Non-identifying relationships	Problem 2
Lemma3		Rule 6 Rule 7 Rule 8 Rule 9 Rule 10 Rule 11	Integrity constraint: not null Integrity constraint: unique Integrity constraint: primary key Integrity constraint: foreign key Integrity constraint: default Integrity constraint: check	Problem 3

Figure 9. Relationships among the lemmas, rules, and problems.

4.1. Rules for General Relational Schemas

This section defines the rules using Lemmas 1 and 2 to generate accurate RDF data from relational data without information loss (proofs are provided in the Appendices C and D). Lemma 1 describes the feature of the OWL class during the mapping process.

- Lemma 1: Suppose R is a relational table set, A is an attribute set, K is an integrity constraint set, I is a relational instance set, X is a set where $X \subset (R \cup A \cup K \cup I)$, and F is a direct mapping function. Then, every $y \in F(X)$ by inference from `owl:Class` can be retrieved.

Thus, to avoid Problem 1 described in Section 3.3, Rule 1 for mapping relational tables based on Lemma 1 is defined as follows.

- Rule 1: $\text{Rel}(r) \wedge \neg \text{BinRel}(r, a_1, \dots, a_m, s, b_1, \dots, b_n, t) \rightarrow \text{Relation}(r)$, where the predicates used on the left-hand side are defined in the Appendix A, and $\text{Relation}(r)$ is a predicate that verifies that r is a relational table and not a binary relation.

By Rule 1, relational tables are transformed into semantic resources using `Relation` (typed OWL class), which is a semantic vocabulary to notate relational tables. For example, if a relational table, “Student,” is transformed by a naive rule, “ $\text{Rel}(\text{Student}) \rightarrow \text{Class}(\text{Student})$,” then the transformed output *Student* loses the explicit information indicating that it is a relational table. This loss happens, because all semantic resources are typed only by the OWL class (Figure 10a). On the other hand, *Student* will not lose the information that it is a relational table if Rule 1 is implemented. *Relation* is defined as a type of *Student* using Rule 1, which provides explicit information that the RDF data is transformed from a relational table (Figure 10b).

Figure 10. A comparative example of rules for mapping relational tables (Rule 1).

Lemma 2 illustrates the feature of the OWL object property used to express the semantics of relationships between semantic resources.

- Lemma 2: For any X in relational data, if $x \in X$ references another $y \in X$, then x can be transformed into a semantic resource, which has a type of `owl:ObjectProperty`.

Based on Lemma 2, if a direct-mapping method does not manage the feature of an object property accurately, then Problem 2 described in Section 3.3 can occur during the mapping process. Thus, Rules 2–5 based on Lemma 2 for mapping the semantics of relationships are defined. Rule 2 is composed of five sub rules to specify the attributes within the hierarchical structure:

- Rule 2: $\text{Prop}(a, r, _) \wedge \text{FP}(a) \rightarrow \text{Attr}(a, r)$
 $\text{ObjProp}(a, r, s) \rightarrow \text{FKeyAttr}(a, r, s)$
 $\text{DataProp}(a, r, \text{type}(a)) \rightarrow \text{NonFKeyAttr}(a, r)$
 $\forall a \forall r \text{ FKeyAttr}(a, r, s) \subseteq \forall a \forall r \text{ Attr}(a, r)$
 $\forall a \forall r \text{ NonFKeyAttr}(a, r) \subseteq \forall a \forall r \text{ Attr}(a, r),$

where the predicates on the left-hand side are defined in the Appendix A, and the predicates on the right-hand side represent the transforms of relational attribute a.

With these predicates, Rule 2 has a distinct advantage over previous approaches that simply used the OWL object property and the datatype property to map relational attributes. Because the OWL properties are provided to describe any resource with referencing semantics (not just for relational attributes), using only the OWL properties will not always guarantee that the output data was originally attributed data. As a result, previous approaches cannot avoid semantic information loss during mapping attributes. However, Rule 2 adopts hierarchical structured semantic vocabularies on attributes (Figure 11). The vocabularies describe various types of attributes, and each input attribute can be transformed into RDF data with detailed information.

Figure 11. Set of attributes as a hierarchical structured semantic vocabulary (Rule 2).

Rule 3 is defined for mapping binary relations as follows:

- Rule 3: $\text{BinRel}(r, a_1, \dots, a_m, s, b_1, \dots, b_n, t) \wedge \neg \text{BinRel}(s, _ _ _ _) \wedge \neg \text{BinRel}(t, _ _ _ _) \rightarrow \text{BinaryRelation}(r, s, t),$

where the predicates on the left-hand side are defined in the Appendix A, and $\text{BinaryRelation}(r, s, t)$ is a predicate that verifies whether a binary relation, r , can be transformed into semantic resource BinaryRelation (typed OWL object property), which is a semantic vocabulary that notates binary relations.

Although both Rules 2 and 3 use $\text{owl:ObjectProperty}$ during the mapping process, the mapping results of the two rules can be readily distinguished. The semantics of type $\text{owl:ObjectProperty}$ are encapsulated by the mapping resource, FKeyAttr , in Rule 2 (Figure 11) and BinaryRelation in Rule 3 (Figure 12). Therefore, a mapping result having FKeyAttr implies that it was originally an attribute of relational data. Thus, from a result having BinaryRelation , we can infer that it was a binary relation before the mapping process.

Figure 12. Semantic vocabulary of a binary relation (Rule 3).

Rules 4 and 5 indicate the relationships between relational tables:

- Rule 4: $\text{Rel}(s) \wedge \text{Rel}(t) \wedge \text{PK}(a, s) \wedge \text{FK}(a, s, _ t) \wedge \text{ObjProp}(r, s, t) \wedge \text{FP}(r) \rightarrow \text{IdentifyingRelationship}(r, s, t)$
- Rule 5: $\text{Rel}(s) \wedge \text{Rel}(t) \wedge \text{PK}(a, s) \wedge \neg \text{FK}(a, s, _ t) \wedge \text{ObjProp}(r, s, t) \wedge \text{FP}(r) \rightarrow \text{NonIdentifyingRelationship}(r, s, t),$

where the predicates on the left-hand side are defined in the Appendix A, $\text{IdentifyingRelationship}(r, s, t)$ is a predicate that verifies identifying relationships, and $\text{NonIdentifyingRelationship}(r, s, t)$ is a predicate that verifies nonidentifying relationships.

Figure 13 shows an example of mapping an identifying relationship. Because a primary key of *Professor* contains a foreign key referencing *Person*, the relation *Professor* is dependent on the relation *Person*. In such a case, relationships between *Professor* and *Person* can be mapped using $\text{IdentifyingRelationship}()$, as defined by Rule 4.

Figure 13. An example of mapping an identifying relationship (Rule 4).

Figure 14 is an example of mapping a nonidentifying relationship. Because the foreign key of *Student* referencing *Major* is not an attribute for the primary key of *Student*, the relations, *Student* and *Major*, are independent. In such a case, relationships between *Student* and *Major* can be mapped using $\text{NonIdentifyingRelationship}()$ by Rule 5.

Figure 14. An example of mapping a nonidentifying relationship (Rule 5).

4.2. Rules for Relational Integrity Constraints

This section provides additional rules for transforming relational integrity constraints to prevent incorrect RDF data generation problems. Lemma 3 illustrates the feature of RDF data, using a linked graph structure. This feature acts as a major factor that prevents the generation of incorrect RDF data. Thus, this section defines Rules 6–11 for the mapping integrity constraints based on Lemma 3 (proofs are provided in the Appendix E).

- Lemma 3: Suppose G is an RDF graph, G_1 and G_2 are the components of G , there is no edge between G_1 and G_2 , G_1 is rooted at $x \in R$, and G_2 is rooted at $y \in R$, where R is a set of semantic resources. Then,
 1. If x and y have the same uniform resource identifier (URI) [36], then x is identical to y . Thus, G_1 and G_2 can be merged into one graph.
 2. If x and y have different URIs, x has a property, p_1 , and y has a property, p_2 , that has the same URI as p_1 , then G_1 and G_2 cannot be merged into one graph, and p_1 can be distinguished from p_2 using x and y .
- Rule 6: $\text{NonFKKeyAttr}(a, r) \wedge \text{subClassOf}(r, _b) \wedge \text{Card}(a, _b, 1) \rightarrow \text{NotNull}(a, r)$
- Rule 7: $\text{NonFKKeyAttr}(a, r) \wedge \text{IFP}(a) \wedge \text{subClassOf}(r, _b) \wedge \text{MaxCard}(a, _b, 1) \wedge (\exists!v) a(r, v) \rightarrow \text{Unique}(a, r)$

- Rule 8: $\text{NonFKeyAttr}(a, r) \wedge \text{IFP}(a) \wedge \text{subClassOf}(r, _b) \wedge \text{Card}(a, _b, 1) \wedge (\exists!v) a(r, v) \rightarrow \text{PK}(a, r)$
- Rule 9: $\text{FKeyAttr}(a, r, s) \wedge \text{subClassOf}(r, _b) \wedge \text{MinCard}(a, _b, 1) \rightarrow \text{FK}(a, r, s)$
- Rule 10: $\text{NonFKeyAttr}(a, r) \wedge \text{subClassOf}(r, _b) \wedge \text{MaxCard}(a, _b, 1) \wedge \text{DefVal}(a, _b, v) \rightarrow \text{Default}(a, r)$
- Rule 11: $\text{NonFKeyAttr}(a, r) \wedge \text{subClassOf}(r, _b) \wedge \text{MaxCard}(a, _b, 1) \wedge \text{CheckCond}(a, _b, v) \rightarrow \text{Check}(a, r),$

where a is an attribute of relational table r , $_b$ is a blank node [37], v in Rules 7 and 8 is an attribute value, v in Rule 10 is a default attribute value, and v in Rule 11 is a check condition. The predicates on the left-hand side are defined in the Appendix A, and the predicates on the right-hand side preserve the integrity constraints: not null, unique, primary key, foreign key, default, and check.

Rule 6 describes the *NotNull* constraint. It also defines a predicate, $\text{Card}(a, _b, 1)$, that restricts the cardinality of an attribute to be exactly one (Figure 15a). Rule 7 specifies a unique constraint and defines a predicate with a unique existential quantifier, $(\exists!v) a(r, v)$, such that there is only one attribute value, v , contained in the domain of $a(r, v)$ (Figure 15b).

Figure 15. Examples of mapping integrity constraints (Rule 6,7).

Rule 8 specifies the primary key defined by $\text{Card}(a, _b, 1)$ and $(\exists!v) a(r, v)$, to assign an attribute, a , with a primary key (Figure 16a). To define a foreign-key constraint, Rule 9 specifies a lower bound of the cardinality using $\text{MinCard}(a, _b, 1)$, because the relational tables can reference more than one other table. Rule 9 also uses $\text{FKeyAttr}(a, r, s)$ to describe the semantics that the type of attribute a is an OWL object property with domain r and range s (Figure 16b).

Figure 16. Examples of mapping integrity constraints (Rule 8,9).

Rule 10 specifies the default constraint, and it uses a function $\text{DefVal}(a, _b, v)$ that returns a default value, v , if a value of the attribute, a , is omitted (Figure 17a). In Rule 11, a function $\text{CheckCond}(a, _b, v)$ is used to restrict the value range for the check constraint. For example, $\text{CheckCond}(\text{quantity}, _b, \text{'quantity'} > 0)$ means that a value of the attribute quantity must be greater than zero (Figure 17b).

Figure 17. Examples of mapping integrity constraints (Rule 10,11).

4.3. Soundness and Completeness of the Rules

Lemmas describe the features of semantic resources during the mapping process. Lemma 1 states that every semantic resource can be inferred from the OWL class. Lemma 2 states that every semantic resource referencing other resources can be typed by the OWL object property. Lemma 3 states that every subgraph, which has the same semantic resource as a root node, can be merged into a single graph. On the other hand, the problems described in Section 3.3 are specific cases of violation of semantics preservation. Problem 1 illustrates the loss of information when the relational tables are transformed without considering Lemma 1. Problem 2 illustrates another case of loss of information when attributes, binary relations, or other referencing objects are transformed without considering Lemma 2. Problem 3 illustrates incorrect RDF data generation when integrity constraints are transformed without considering Lemma 3. Therefore, mapping rules are defined based on the lemmas to perform semantics-preserving transformation of RDBs to RDF data and to avoid the loss of semantics or incorrect RDF data generation. Lemma 4 demonstrates the soundness and completeness of the provided RDB2RDF data transformation methods (see the Appendix F for proof).

- Lemma 4: Consider that X is a set of relational data, F is an RDB2RDF mapping function, and G is an RDF2RDB inverse-mapping function of F . If the mapping rules are defined based on Lemmas 1, 2, and 3, then,
 1. Soundness: the mapping rules are sound if the rules generate only semantics in RDB data ($X \supseteq G(F(X))$).
 2. Completeness: the mapping rules are complete if the rules generate all semantics in RDB data ($X \subseteq G(F(X))$).

5. Experimental Results

5.1. Environments

Experiments were conducted using five real datasets and one synthetic dataset on a cluster of 12 nodes using a 3.1-GHz quad-core processor, 4-GB memory, and a 2-TB hard disk. Each real dataset contains relational schema information with integrity constraints: Ensembl-compara (www.ensembl.org/info/docs/api/compara), Ensembl (www.ensembl.org), PHPmyadmin (<https://www.phpmyadmin.net>), and MusicBrainz (<https://musicbrainz.org>). The DBT2 (<http://osdltdb.sourceforge.net>) benchmark was used for the synthetic dataset. This work generated warehouse data using DBT2 and restructured the schema by adding integrity constraints to evaluate the semantics preservation of the mapping methods.

5.2. Analysis

Figure 18a presents the results for the number of triples transformed from the relational data. To perform a comparative analysis of cost efficiency of the mapping rules, this work employed OWL

ontology-based augmented direct mapping [13], which provides implementation details of the mapping algorithm and demonstrates the improvement over other previous methods. The horizontal axis represents the relational data size of the input as mapping methods, and the vertical axis represents the number of semantic triples as output data. As viewed in Figure 18a, our approach generates fewer triples compared with the previous method. Figure 18b shows the average number of triples that result from each transformation method. The horizontal axis represents each relational dataset, and the vertical axis represents the average number of triples generated from the transformation of a single relational element. Assuming that two output results are identical in terms of semantics, the method that generates a smaller-sized result is better with regard to both space and computation. On one hand, when the input data are transformed into RDF data, the mapping rule uses the hierarchical RDF data model. On the other hand, previous methods generated output data without a predefined RDF data model, and repetitive RDF data were generated using primitive semantic model languages when the input data had several referential relationships or constraints. Thus, the results show that the proposed approach generates more compact RDF data that express the same information with fewer resources.

Figure 18. Comparative results of the mapping methods.

Figure 19 shows the failure rate of the mapping methods in each database. The horizontal axis represents each relational dataset, and the vertical axis represents the failure rate during the transformation of relational data into RDF data. The mapping failures of the previous approach result in the incorrect RDF data generation problem was discussed in Section 3. These failures occurred because the previous methods lacked support for handling the integrity constraints. Our approach followed Lemma 4 and guaranteed that the hierarchical mapping rules generated fewer false mapping results. By our method, the false results could occur when the input data are defined based on the practical SQL statements that are not included in standard SQL.

Figure 19. RDB2RDF failure rate of the mapping methods.

Figures 20 and 21 illustrate the soundness and completeness of mapping rules when the size of the relational input data varies from 20 to 100. This experiment duplicated the experiments of Astrova et al. [8], Buccella et al. [9], Li et al. [12], Lim et al. [13], Shen et al. [15], and Tirmizi et al. [16]. The results show that the proposed method adheres to the definition of the semantics-preserving direct-mapping rule. The mapping rules generate only semantics in RDB data and generate most of the semantics of the RDF data.

Figure 20. Soundness of RDB2RDF data transformation methods.

Figure 21. Completeness of RDB2RDF data transformation methods.

6. Conclusions

The paper focuses on the problem of the existing direct-mapping methods that they do not fully support mapping-integrity constraints. The problems are observed in specific cases in which semantic information loss or incorrect RDF data generation occurred. In this paper, the improved definition of semantics preservation is provided to solve the problems and augment the RDB2RDF mapping methods.

Three lemmas are defined to describe the features of semantic resources during the mapping process. Lemma 1 stated that each semantic resource could be inferred from an OWL class (i.e., semantic resource). Lemma 2 stated that each semantic resource referencing other resources could be typed by an OWL object property (i.e., referential relationship). Lemma 3 stated that each subgraph having the same semantic resource as a root node could be merged into a single graph (i.e., union of semantic resources). A hierarchical structured semantic vocabulary was also defined for use in direct-mapping rules.

Rule sets are defined based on the lemmas to transform relational tables and attributes. The mapping rules comprised general- and constraint-mapping rules. The general-mapping rules are used for mapping relations, attributes, and other general relational objects and are defined to avoid semantic information loss during the transformation of general relational objects. Constraint mapping rules are used for mapping the integrity constraints and are defined to reduce the volume of incorrect RDF data generated.

Finally, the semantics-preserving direct-mapping method was implemented, and a comparative experimental study was performed with both synthetic and real datasets. The experiments demonstrated that the proposed mapping method performs semantics-preserving RDB2RDF transformation and generates semantically accurate results. In the future, we will study the methods of synchronization to achieve RDF data consistency when the original relational data are modified [38]. We will also build a cost-benefit model that reduces the number of repetitive processes.

Author Contributions: H.-G.J. conceive the problem and implemented the algorithm and performed the experiments; D.-H.I. supervised the whole research work and revised the algorithm and the theorems; H.-G.J. and D.-H.I. wrote the paper. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the Ministry of Science and ICT (MSIT), Korea, under the Information Technology Research Center (ITRC) support program (IITP-2020-2018-08-01417) supervised by the Institute for Information & communications Technology Promotion (IITP).

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Definitions of Predicates Used in Mapping rules

The predicates shown in Table A1 are used for the verification of OWL ontology and to be used by the mapping rules.

Table A1. List of predicates used in mapping rules.

Predicates	Conditions of Predicates to Return True
Class(<i>r</i>)	<i>r</i> is an OWL class
Prop(<i>p</i> , <i>d</i> , <i>r</i>)	<i>p</i> is an RDF property with domain <i>d</i> and range <i>r</i>
ObjProp(<i>p</i> , <i>d</i> , <i>r</i>)	<i>p</i> is an OWL object property (ObjProp) with domain
DataProp(<i>p</i> , <i>d</i> , <i>t</i>)	<i>d</i> and range <i>r</i>
FP(<i>p</i>)	<i>p</i> is and OWL datatype property (DataProp) with
IFP(<i>p</i>)	domain <i>d</i> and datatype <i>t</i>
Card(<i>p</i> , <i>v</i>)	<i>p</i> is an OWL functional property (FP)
MinCard(<i>p</i> , <i>v</i>)	<i>p</i> is an OWL inverse functional property (IFP)
MaxCard(<i>p</i> , <i>v</i>)	cardinality of property <i>p</i> is <i>v</i>
type(<i>x</i> , <i>t</i>)	minimum cardinality of property <i>p</i> is <i>v</i>
subClassOf(<i>x</i> , <i>y</i>)	maximum cardinality of property <i>p</i> is <i>v</i>
Rel(<i>r</i>)	datatype of <i>x</i> is <i>t</i>
BinRel(<i>r</i> , <i>a</i> ₁ , ..., <i>a</i> _{<i>m</i>} , <i>s</i> , <i>b</i> ₁ , ..., <i>b</i> _{<i>n</i>} , <i>t</i>)	<i>x</i> is a subclass of <i>y</i>
Attr(<i>a</i> , <i>r</i>)	<i>r</i> is a relation
FKeyAttr(<i>a</i> , <i>r</i>)	<i>r</i> is a binary relation between relation <i>s</i> with primary
NonFKeyAttr(<i>a</i> , <i>r</i>)	key columns <i>a</i> ₁ , ..., <i>a</i> _{<i>m</i>} and <i>t</i> with primary key
	columns <i>b</i> ₁ , ..., <i>b</i> _{<i>m</i>}
	<i>a</i> is an attribute of relation <i>r</i>
	<i>a</i> is a foreign key
	<i>a</i> is not a foreign key

The properties in Table A1 (words in bold) are OWL properties. For better understanding, we provide the relationships among triples, OWL properties, and their semantics in Table A2.

Table A2. Examples of semantic triple data using the OWL notation.

(Subject, Property, Object)	Types of Property	Semantics
(Professor, teaches, Student)	Object property	Relation between resources
(Person, age, "23")	Data type property	Subject has a data type value
(Product, produced, ProductLine)	Functional property (FP)	N:1 relationship
(Shakespeare, writes, Book)	Inverse functional property (IFP)	1:N relationship
(Husband, marriage, Wife)	FP and IFP	1:1 relationship

Appendix B. An Example of Problem 3

Figure A1 shows an example of Problem3, assume that a relational attribute x has integrity constraints “unique” and “not null”, F is a mapping function that contains the rules in Figure 7, and G is an inverse-mapping function of F , then the integrity constraints of $G(F(x))$ are “unique”, “not null”, and “primary key” because $FK(p) \subseteq Unique(p)$, $NotNull(p) \subseteq Unique(p)$, and $PK(p) \subseteq NotNull(p) \cup Unique(p)$.

Figure A1. An example of Problem 3.

Appendix C. Proof of Lemma 1

If $x \in R$, then x can be transformed directly into `owl:Class`. If $x \in A$, then x can be transformed into either `owl:ObjectProperty` or `owl:DatatypeProperty`, which are types of `rdfs:Class`. If $x \in A \cup K$, then x can be transformed into `owl:FunctionalProperty` or `owl:InverseFunctionalProperty`, which are types of `rdfs:Class`. If $x \in K$, then x can be transformed into `owl:onProperty`, `owl:minCardinality`, `owl:maxCardinality`, or `owl:cardinality`, all of which are types of `rdf:Property`, and the type of `rdf:Property` is `rdfs:Class`. As `rdfs:Class` is a subclass of `owl:Class`, semantic resources used in transformation are directly or indirectly assigned to `owl:Class` type. Therefore, transformed RDF data $F(x)$ can be retrieved by inference using `owl:Class`.

Appendix D. Proof of Lemma 2

Let a be an attribute and b a relational table. If a is a foreign key of table x referencing a primary key attribute in table y , then a can be transformed into `owl:ObjectProperty` with domain x and range y . If b is a binary relation over table t and u , then b can be transformed into `owl:ObjectProperty` with domain t and range u . Even if a bigger relationship exists, the relation can be transformed using `owl:ObjectProperty`. Therefore, `owl:ObjectProperty` can describe any referencing relationship among relational data sources.

Appendix E. Proof of Lemma 3

First, we prove (1). Assume that G_1 and G_2 cannot be merged into one graph, G_1 has only one triple $t(x, p_1, o_1)$, G_2 has only one triple $t(y, p_2, o_2)$ that is identical to $t(x, p_2, o_2)$, and $q(s, p, o)$ is a query function to find triples. If $q(x, ?p, ?o)$ is the input of the query function, then the result is $\{t(x, p_1, o_1), (x, p_2, o_2)\}$. This contradicts our assumption that G_1 and G_2 cannot be merged into one graph. Therefore, if x and y have the same URI, then G_1 and G_2 can be merged into one graph. Second, we prove (2). Assume that G_1 and G_2 can be merged into one graph, p is a property that has the same URI as p_1 and p_2 , $p = p_1 = p_2$, G_1 has only one triple $t(x, p, o_1)$, G_2 has only one triple $t(y, p, o_2)$, and $q(s, p, o)$ is a query function to find triples. If $q(y, p, o_2)$ is the input of the query function, then there is no matching result. If $q(x, p, o_1)$ is the input for the query function, then the result is also empty. This contradicts our assumption that G_1 and G_2 can be merged into one graph. Therefore, if x and y have different URIs, then G_1 and G_2 cannot be merged into one graph.

Appendix F. Proof of Lemma 4

First, we prove the completeness of the mapping rules by induction. Suppose S is an RDB schema set, S' is the semantic graph data that represents every schema in S , F is a mapping function and G is an inverse-function of F , and X is input data, where $X \subset S$. (1) Base: Assume $X = \{\}$, then $F(X) = \{\}$ and $G(F(X)) = \{\}$, thus $X \subseteq G(F(X))$. (2) Inductive hypothesis: Given a set $X = \{x_1, x_2, \dots, x_k\}$ and $|X| = k$. Assume $F(X)$ is a mapping function for all $X \subset S$, which transforms X to RDF data that $F(X) \subset S'$. (3) Inductive step: Given a set of RDB data $\{x_1, x_2, \dots, x_{k+1}\}$ and $|X| = k+1$. Consider the set X' without x , where x is any element of X , and $|X'| = k$. We have X' and x that are excluded from X . Now, we can apply the inductive hypothesis to X' , mapping them all to be transformed into RDF data $F(X') \subset Y$ and $G(F(X')) \subset S$. We can also apply the inductive hypothesis to x , which is identical to X when k is one. Therefore, by the mapping rules, $X \subseteq G(F(X))$. Second, we prove the soundness of the mapping rules. Assume S is an RDB schema set, S' is the semantic graph data that represents every schema in S , F is a mapping function that generates a directed graph with a root node containing a unique URI value, and G is an inverse-mapping function of F . If input data X is a disjoint set, $|X| = k$, $X = \{x_1, x_2, \dots, x_k\}$, then $F(X) = y_k$ is a graph data and $F(X)$ is a disjoint set by Lemma 3.

References

1. He, B.; Patel, M.; Zhang, Z.; Chang, K.C.C. Accessing the deep web. *Commun. ACM* **2007**, *50*, 94–101. [CrossRef]
2. De Laborda, C.P.; Conrad, S. Database to Semantic Web mapping using RDF query languages. In *International Conference on Conceptual Modeling*; Springer: Berlin/Heidelberg, Germany, 2006; pp. 241–254.
3. Spanos, D.E.; Stavrou, P.; Mitrou, N. Bringing relational databases into the semantic web: A survey. *Semant. Web* **2012**, *3*, 169–209. [CrossRef]
4. Relational Databases on the Semantic Web. Available online: <http://www.w3.org/DesignIssues/RDB-RDF.html> (accessed on 28 August 2019).
5. Arenas, M.; Bertails, A.; Prud'hommeaux, E.; Sequeda, J. A direct mapping of relational data to RDF. *W3C Recomm.* **2012**, *27*, 1–11.
6. Sequeda, J.F.; Arenas, M.; Miranker, D.P. A completely automatic direct mapping of relational databases to RDF and OWL. In *Proceedings of the 10th International Semantic Web Conference (ISWC2011)*, Bonn, Germany, 23–27 October 2011.
7. Sequeda, J.F.; Arenas, M.; Miranker, D.P. On directly mapping relational databases to RDF and OWL. In *Proceedings of the 21st international conference on World Wide Web*, Lyon, France, 16–20 April 2012; pp. 649–658.
8. Astrova, I.; Korda, N.; Kalja, A. Rule-based transformation of SQL relational databases to OWL ontologies. In *Proceedings of the 2nd International Conference on Metadata & Semantics Research*, Corfu, Greece, 2–11 October 2007; pp. 415–424.
9. Buccella, A.; Penabad, M.R.; Rodriguez, F.J.; Farina, A.; Cechich, A. From relational databases to OWL ontologies. In *Proceedings of the 6th National Russian Research Conference*, 29 September–1 October 2004.

10. Cerbah, F. Learning highly structured semantic repositories from relational databases. In *European Semantic Web Conference*; Springer: Berlin/Heidelberg, Germany, 2008; pp. 777–781.
11. Cullot, N.; Ghawi, R.; Yétongnon, K. DB2OWL: A tool for automatic database-to-ontology mapping. *SEBD* **2007**, *7*, 491–494.
12. Li, M.; Du, X.Y.; Wang, S. Learning ontology from relational database. In *Proceedings of the 2005 International Conference on Machine Learning and Cybernetics*, Guangzhou, China, 18–21 August 2005; Volume 6, pp. 3410–3415.
13. Lim, K.B.; Jun, H.G.; Kim, H.J. Semantics preserving MapReduce process for RDB to RDF transformation. *Int. J. Metadata Semant. Ontol.* **2015**, *10*, 229–239. [[CrossRef](#)]
14. Jun, H.G.; Im, D.H.; Kim, H.J. Semantics preserving optimisation of mapping multi-column key constraints for RDB to RDF transformation. *J. Inf. Sci.* **2020**. [[CrossRef](#)]
15. Shen, G.; Huang, Z.; Zhu, X.; Zhao, X. Research on the rules of mapping from relational model to OWL. *OWLED* **2006**, 216. Available online: <http://ceur-ws.org/Vol-216/> (accessed on 11 October 2020).
16. Tirmizi, S.H.; Sequeda, J.; Miranker, D. Translating SQL applications to the semantic web. In *International Conference on Database and Expert Systems Applications*; Springer: Berlin/Heidelberg, Germany, 2008; pp. 450–464.
17. Lassila, O.; Swick, R.R. Resource Description Framework (RDF) Model and Syntax Specification. 1998. Available online: <https://www.w3.org/TR/1998/WD-rdf-syntax-19980720/> (accessed on 11 October 2020).
18. Lee, T.B.; Connolly, D. Delta: An Ontology for the Distribution of Differences between RDF Graphs. Available online: <https://www.w3.org/DesignIssues/Incs04/Diff.pdf> (accessed on 11 October 2020).
19. Dau, F. RDF as graph-based, diagrammatic logic. In *International Symposium on Methodologies for Intelligent Systems*; Springer: Berlin/Heidelberg, Germany, 2006; pp. 332–337.
20. Hayes, J. A Graph Model for RDF. Darmstadt University of Technology/University of Chile. 2004. Available online: <http://users.dcc.uchile.cl/cgutierrez/papers/rdfgraphmodel.pdf> (accessed on 11 October 2020).
21. Sahoo, S.S.; Halb, W.; Hellmann, S.; Idehen, K.; Thibodeau, T., Jr.; Auer, S.; Sequeda, J.; Ezzat, A. A survey of current approaches for mapping of relational databases to RDF. *W3C RDB2RDF Incubator Group Rep.* **2009**, *1*, 113–130.
22. Michel, F.; Montagnat, J.; Faron-Zucker, C. A Survey of RDB to RDF Translation Approaches and Tools; [Research Report] I3S, 2014, fhal-00903568v2f. Available online: https://hal.archives-ouvertes.fr/hal-00903568/file/Rapport_Rech_I3S_v2_-_Michel_et_al_2013__A_survey_of_RDB_to_RDF_translation_approaches_and_tools.pdf (accessed on 11 October 2020).
23. Byrne, K. Having triplets-holding cultural data as RDF. In *Proceedings of the ECDL 2008 Workshop on Information Access to Cultural Heritage*, Aarhus, Denmark, 18 September 2008.
24. Green, J.; Dolbear, C.; Hart, G.; Goodwin, J.; Engelbrecht, P. Creating a semantic integration system using spatial data. In *Proceedings of the 2007 International Conference on Posters and Demonstration Session at the 7th International Semantic Web Conference (ISWC2008)*, Karlsruhe, Germany, 26–30 October 2008; Volume 401, pp. 70–71.
25. Hert, M.; Reif, G.; Gall, H.C. A comparison of RDB-to-RDF mapping languages. In *Proceedings of the 7th International Conference on Semantic Systems*, Graz, Austria, 7–9 September 2011; pp. 25–32.
26. R2RML: RDB to RDF Mapping Language. Available online: <https://www.w3.org/TR/r2rml/> (accessed on 9 October 2019).
27. Bizer, C.; Seaborne, A. D2RQ-treating non-RDF databases as virtual RDF graphs. In *Proceedings of the 3rd International Semantic Web Conference (ISWC2004)*, Hiroshima, Japan, 7–11 November 2004.
28. The D2RQ Platform v0.7-Treating Non-RDF Relational Databases as Virtual RDF Graphs. User Manual and Language Specification. Available online: <http://wifo5-03.informatik.uni-mannheim.de/bizer/d2rq/spec/20090810/> (accessed on 30 January 2018).
29. Blakeley, C. Mapping Relational Data to RDF with Virtuoso’s RDF Views. OpenLink Software. 2007. Available online: <http://virtuoso.openlinksw.com/dataspace/doc/dav/wiki/Main/VOSSQLRDF> (accessed on 11 October 2020).
30. Erling, O.; Mikhailov, I. RDF Support in the Virtuoso DBMS. In *Networked Knowledge-Networked Media*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 7–24.
31. Sequeda, J.F.; Depena, R.; Miranker, D.P. *Ultrawrap: Using SQL Views for RDB2RDF*; ISWC 2009 Posters & Demonstrations Track: Seattle, WA, USA, 2009.

32. Sequeda, J.F.; Miranker, D.P. Ultrawrap: SPARQL execution on relational data. *J. Web Semant.* **2013**, *22*, 19–39. [[CrossRef](#)]
33. Malik, K.R.; Ahmad, T.; Farhan, M.; Aslam, M.; Jabbar, S.; Khalid, S.; Kim, M. Big-data: Transformation from heterogeneous data to semantically-enriched simplified data. *Multimed. Tools Appl.* **2016**, *75*, 12727–12747. [[CrossRef](#)]
34. Sharma, K.; Marjit, U.; Biswas, U. RDF link generation by exploring related links on the Web of data. *Int. J. Inf. Technol. Comput. Sci.* **2018**, *10*, 62–68. [[CrossRef](#)]
35. Tong, Q. Mapping object-oriented database models into RDF (S). *IEEE Access* **2018**, *6*, 47125–47130. [[CrossRef](#)]
36. URIs, URLs, and URNs: Clarifications and Recommendations 1.0. Available online: <https://www.w3.org/TR/uri-clarification/> (accessed on 9 October 2019).
37. Mallea, A.; Arenas, M.; Hogan, A.; Polleres, A. On blank nodes. In *International Semantic Web Conference*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 421–437.
38. Im, D.H.; Lee, S.W.; Kim, H.J. A version management framework for RDF triple stores. *Int. J. Softw. Eng. Knowl. Eng.* **2012**, *22*, 85–106. [[CrossRef](#)]

© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).