Generating Relational Database using Ontology Review

Issues, Challenges and Trends

Christina Khnaisser
Département d'informatique
Université de Sherbrooke
Sherbrooke, Canada

Luc Lavoie
Département d'informatique
Université de Sherbrooke
Sherbrooke, Canada

Anita Burgun
INSERM UMR 1138 team 22, CRC
Université Paris Descartes
Paris, France

Jean-Francois Ethier
Département de médecine / informatique
Université de Sherbrooke, Sherbrooke, Canada
INSERM UMR 1138 team 22, CRC
Paris, France

***Abstract*—A huge amount of data is being generated every day from different sources. Access to these data can be very valuable for decision-making. Nevertheless, the extraction of information of interest remains a major challenge given a large number of heterogeneous databases. Building shareable and (re)usable data access mechanisms including automated verification and inference mechanisms for knowledge discovery needs to use a common knowledge model with a secure, coherent, and efficient database. For this purpose, an ontology provides an interesting knowledge model and a relational database provides an interesting storage solution. Many papers propose methods for converting ontology to a relational database. This paper describes issues, challenges, and trends derived from the evaluation of 10 methods using 23 criteria. Following this study, this paper shows that none of the methods are complete as well as the conversion process does not use the full expressivity of ontology to derive a complete relational schema including advanced constraints and modification procedures. Thus, more work must be done to decrease the gap between ontologies, a relation database.**

***Keywords*—Ontology; relational database; database modeling; knowledge model; ontology to relational database**

I. Introduction

Information systems are now at the center of decisions in many areas (healthcare, economic, industrial, manufacturing and so on). A huge amount of data is being generated every day from different sources. Organizations desire to reuse this data for many kinds of analyses to enhance decision-making. The correctness of decision-making depends on the quantity but also on the quality of data collected. Nevertheless, data is stored in different sources that are structured (structural heterogeneity) and encoded (terminological heterogeneity) in different ways. On the one hand, to use data efficiently and correctly from these various heterogeneous sources, experts would ideally be able to express their queries according to a unified knowledge model that represents their domain without the need to know the structure of each database nor to manually extract data from many sources each time. The resulting data should also be available in a unified format reflecting the knowledge model used to define the query. On the other hand, data managers must be able to create, manage, and maintain data with the least possible resources while ensuring its fidelity, integrity, and traceability of its evolution. For this purpose, data integration is the mechanism used for combining data from different sources in a unique unified model offering a single access point. In the database field, the two main and widely used techniques to represent a data model (let's call them conventional techniques) are the entity-relationship model [1] and the object-oriented model [2] which are otherwise mutually convertible [3]. However, these conventional techniques do no longer provide expressivity that is sufficiently complete to semantically interpreted and widely reused data outside a restricted field of application [4]. The current trend in data integration is, therefore, the use of knowledge to enhance the process [4], [5].

In many heterogeneous environments, a knowledge model seems very useful to decipher source structure, and isolate interesting data elements to extract and combine [4]. In the early 80s, the computer science community adopted the ontologies as a knowledge model with reasoning abilities [6] to provide a shared conceptualization of some domain of interest [7]. Since then, ontologies (such as those expressed using the OWL language [8]) have been used in different ways: database modeling, data integration, data mapping, data exchange, data annotation, information retrieval, knowledge discovery, and so on [9]. In particular, ontologies are becoming an important tool for data integration because they handle semantic interoperability by describing a common understanding of data without preoccupation with the underlying layer [7], [4]. In addition, sound integration cannot be done without handling data integrity when a large amount of data sources is highly fragmented and heterogeneous. In light of the recent technology, Relational Database Management System

(RDBMS) with vertical representation [10] and in-memory databases [11] are performing significantly better than "triple stores" sometimes used in conjunction with ontologies [12]. Thus, a RDBMS is needed to allow multiple users to store, modify, and interrogate a large volume of data in a concurrent, reliable, and secure manner [13]–[15].

Yet, since ontologies offer excellent data integration support for disparate systems, a relational database derived directly from an ontology can be hypothesized to be the best way to ensure data integrity and a unique data access point for a large volume of data. First, modeling a relational database using ontologies (rather than entity-relationship) allows the reuse of ontology in many tasks facilitating queries expression as well as ensuring semantic and structural uniformity [16], [17]. Second, with the axiomatic model underlying the ontology, data storage and verification can be automated, increasing data integrity. Reasoning mechanisms can also be leveraged to allow for knowledge discovery. Finally, both models (ontological and relational) must be used together to benefit from the expressiveness of ontologies and the maturity of databases [16].

Several methods have been proposed in the literature to convert an ontology into a relational database [5]. Nevertheless, methods differ in terms of the various ontological constructs covered to generate an enriched relational model. Therefore, 10 methods using 23 specific criteria were described and analyzed.

The rest of the paper is organized as follows: Section II describes the study methodology including conversion issues and challenges as well as criteria definition and the selection process. Section III presents the analysis of the evaluated methods. Section IV elaborates on the advantages of using ontologies and gives an overview of current trends. Section V concludes the paper. Finally, in the appendix, the results of each criterion for the 10 methods are presented in a table.

II. Study Methodology

In this section, first, some ontology and relation database constructs are briefly presented. Then, some noteworthy conversion issues and, challenges are described. Twenty-three evaluation criteria were then derived to evaluate different methods, analyze the challenges, and underline the trends. Using the criteria, the evaluation of 10 methods is presented in the form of a table in the appendix.

A. Issues and Challenges

An ontology and a relational database (relational theory) represent facts using different modeling construct [18]. An ontology is defined using classes, individuals, axioms, properties (object properties and data properties), datatypes, and annotations. For more detail please refer to [8], [19]. A relational database is defined by a set of relational variables (a.k.a *relvar* in relational theory or table in SQL), each relational variable is defined by a set of attributes (pairs of a unique name and a datatype) and constraints. For more detail please refer to the references [20], [21]. Both models also share common foundations: the set theory and the first order logic. Thus, a conversion from one to the other is possible, at least in part, but the main challenge is to maintain the richness of ontological definitions in the resulting relational database by converting uniformly and consistently ontology constructs into a relational construct. Some related challenges are now presented.

1) Preserve property cardinalities: An axiom is an expression that links classes or individuals using properties and cardinalities. A cardinality represents the number of individuals of the related entities that can participate regarding a property. A cardinality is represented as a participation with a range having a minimum value and a maximum value such as [0..1], [1..1], [0..n], [1..n], [0..*], [1..*] and [n..n] where n is a positive integer representing the exact participation value, and a "*" (star) is an undefined participation. In a relational database, an axiom can be represented as candidate keys (primary keys, unique keys), referential keys (foreign keys), and general constraints (functions and triggers). A conversion process that handles cardinality constraints increases data integrity and automatic inconsistency detection.

2) Handle missing information: In an ontology, properties of some class can be optional (as attributes in a relational database). That is, the value can be unknown (in some cases) or inapplicable (in some other cases). This can be implemented by the use of null values, but this implementation is not semantically complete. Moreover, null values can introduce data inconsistency in query results. Thus, it is recommended to avoid them by using other modeling techniques as described in [22], [23].

3) Preserve axioms expressivity: Axioms can be defined using different forms: simple or composite. A simple axiom is defined by atomic entities including a property, one or two classes, and a datatype. Several simple axioms can be combined by set operators (intersection and union) and logical operators (conjunction and disjunction) thus forming composite axioms. Let be the following composite axiom **A OP qt (B ∩ C)** where **A**, **B**, and **C** are classes, **OP** is an object property, and **qt** is a quantifier: some, only, min, max, exactly; it is possible to express it in 3 simple axioms: **A OP qt Z ; Z isa B ∧ Z isa C** where **Z** is a new class subclass of Thing. In a relational database, such operators are defined in the conceptual model and may have different transformations [3] generating different structures and constraints. This may be an issue when using popular medical and biomedical ontologies (see ontologies in OBO foundry repositories [24]) because composite axioms are frequently used. There is a need for a conversion process that uniformly converts such axioms by preserving the complete semantic.

4) Maintain Ontology-SQL type compatibility: A type (datatype) is a set of values. On the one hand, in computing, these sets are necessarily finite but this restriction is not automatically applicable on a data property. On the other hand, for some ontology-type (e.g. owl:rational, xsd:positiveInteger) there is no direct mapping to an SQL-type. Furthermore, SQL-types compatibility and exact handling depend on the target RDBMS. This must also be considered when converting types.

5) Enable structural and tuple reversibility: One of the objectives of using ontologies is the reuse of the model (described entities and axioms) in several tasks. Once data is integrated, a sound mechanism for linking tuples to entities can be very valuable for knowledge discovery and ontology-based data access [7]. In this context, the reversibility between the components of an ontology and those of a relation schema is an essential point to take into account when defining the conversion process. Two reversibility features can be defined: the structure reversibility, the ability to reconstruct or identify an ontological construct from a relational construct; the tuples reversibility, the ability to reconstruct individuals (i.e. as RDF triples) from tuples.

6) Handle knowledge and schema evolution: Knowledge is in constant evolution. This implies changes to ontologies with ensuing repercussions on the related relational schema. The schema must, therefore, cope with it while maintaining earlier knowledge interpretations and preserving coherent data [17]. Moreover, with the opportunity to easily access data, new needs will emerge, and existing needs may change. The impact when integrating knowledge change that implies schema evolution can be very large. Consequently, the conversion process needs to be defined in a way to facilitate structure modification and extensibility.

7) Maintain schema documentation: Relational schema documentation is often ignored despite the added value in many tasks such as data querying, mapping definition, application development and so on [25]. Without a clear definition of the data or the database structure, several tasks cannot be verified or even done [4]. Part of the semantics of the class is carried by the axioms but complementary information can be found in annotations. An annotation in the ontology represents a text in a specific language that defines various aspects of an entity. Using annotations can be beneficial for schema generation and documentation. Examples of interesting types of annotations are: labels (e.g. rdf:label) and comments (e.g. rdf:comment), and definitions (e.g. definition from IAO ontology[1]) which propose a common language description of classes, individuals, and properties.

B. Criteria Definition

Based on the issues and challenges described above, criteria were defined and grouped into four categories: the ontology criteria, the relational schema criteria, the conversion process criteria and tool implementation criteria.

1) Ontology criteria

- Ontology language – the ontology language supported by the method: OWL-DL, OWL-QL, OWL-RL, OWL-EL, RDF(S), DAML, etc.

2) Relational schema criteria

- Structure normalization – the relational schema normal form? : 3NF, BCNF, 5NF or 6NF. This criterion can be deduced from the conversion rules.
- Structure scope – the form of the predicate represented by the relvars of the generated relational: schema generic [G] (*RDF generic style* <predicate, subject, object>) or specific [S] (a specific predicate per class or association).
- Domains[2] – does the method convert the ontology data types and their constraints into domains (e.g. CREATE DOMAIN in PostgreSQL)?
- Primary keys – does the method generate [G] or calculate [C] the primary keys? A generated key is an artificial key defined independently of the set of axioms. A calculated key is a key deduced from the set of axioms.
- Secondary keys – does the method generate the secondary key from the set of axioms?
- Foreign keys – does the method convert the appropriate axioms into foreign keys?
- General constraints – does the method convert cardinalities into general constraints?
- Modification procedures – does the method define the procedures for modifying the data (insert, delete, and update triggers)?
- Supported target RDBMS – such as PostgreSQL, MySQL, Oracle, MSSQL, etc.

3) Conversion process criteria

- Axiom normalization – does the conversion process deal with composite axiom?
- Intermediate structure – the intermediate data structure used for the conversion of OWL into a relational schema: MOF (Meta-Object Facility) FOL (First order logic), RDF, Jena model, etc.
- Type conversion – does the conversion process specify or configure the conversion rules between ontology types and SQL types?
- Restriction conversion – does the conversion process convert the restrictions to general constraints? If yes: is explicit conversion [E] or metadata (implicit conversion [I]).
- Annotation conversion – does the conversion process convert the annotations to document the relational schema?

[1] http://www.obofoundry.org/ontology/iao.html

[2] A domain as defined by the type theory is a finite set of values and a type is a constrained domain that restricts the accepted values.

- Structural reversibility – does the conversion process make it possible to refer to the ontology construct?
- Structural reversibility algorithm – does the method describe the algorithm and propose an implementation of structural reversibility?
- Tuples reversibility – does the conversion process make it possible to import tuples stored in the DB in their full ontological expression?
- Tuples reversibility algorithm – does the method describe the algorithm and propose an implementation of tuples reversibility?

4) Implementation criteria

- Existence of an implementation – has the method been implemented?
- Tool availability – is the tool publicly available?

C. Literature Selection Process for the Evaluated Methods

The search process started in February 2018. The first intent of the search was to find a publicly available tool. To identify methods implemented or updated with current technologies, only papers from the year 2010 and up were retained. The search was conducted with Google Scholar and Engineering Village using the following keywords: "Ontology to relational schema", "Ontology to relational database", "Relation schema from ontology". From 178 papers, 10 were selected and evaluated. The selection was based on the completeness of the paper. The completeness was defined regarding the number of criteria. A paper was selected if at least 15 (over 23) criteria can be evaluated. Moreover, authors of the papers were contacted to validate the evaluation or complete the missing values in the evaluation table and 6 out of 10 responded.

III. Results

This section presents the analysis of the results. The analysis is divided into two categories: general observation and, criteria observation. The Appendix I presents specific criteria results for each method.

A. General Observation

The OWL language is the most popular ontology language. However, the methods differ from one another according to the ontology constructs taken into account in the conversion process. The common ontology constructs used are: classes, objects properties, data properties, subclass axioms with simple restriction (some, exactly, min and max), functional properties characteristic as well as the domain and the range of properties. The methods also differ from one another in several other aspects: the conversion rules, the conversion steps, the quality of the relational schema generated, the availability of the tool, etc.

First, there seems to be a consensus on some conversions rules, especially the class conversion rule is the same for all methods, a class is transformed into a relation. But, properties and axioms are generated differently, i.e. a property can be transformed into an attribute or into a relation depending on the granularity of the conversion rules. In addition, several conversions are suggested to increase the integrity of the data, including the generation of secondary keys, general constraints, modification procedures, and so on. Furthermore, no tool supporting multiple ontologies with an advanced conversion process that was publicly available has been identified. It is thus very difficult to reproduce or share the results, let alone reuse it.

B. Criteria Observation

1) Ontology criteria: OWL defines four profiles: EL, RL, QL, and DL. The DL profile being the superset of the three others [26]. It is important to use DL profile to benefit from a higher degree of expressivity, and to cover more general modeling construct (i.e. universal and existential quantification, cardinality restrictions, functional properties, etc.). While some reasoning operations may be undecidable [26] the gain in expressivity is notable. On the other sides, optimized translation schema may be derived taking the restrictions of EL, RL, or QL into account.

DL is the most supported (7/10) and just one method supports specifically EL, QL and RL profiles, but no indication is given on the available optimizations, if any.

2) *Relational schema criteria:* The generated relational databases differ with respect to their structure and their constraints. On the one hand, all methods generate an ontology-specific relational schema except the method presented in [27]. The latter uses generic representations as a "triple store" where all the data are stored in a single large table (subject, predicate, object). This representation accepts any combination of "fact" but makes it difficult to verify data integrity and to process a large volume of data [13]. In addition, an object property is often converted to a join table, and a data property is always converted to an attribute. These conversions rarely take into account the value of the cardinality. On the other hand, regarding the schema constraint, primary key constraints are generated by most of the methods (7/10), and secondary keys are calculated only by few of them (3/10) using property characteristics (functional and inverse functional). In addition, only two methods generate general constraints, yet this generation is limited to an enumeration restriction (i.e. value enumeration inside a check constraint).

Furthermore, modification procedures are only generated by one method [27] and are limited to the insertion procedure. The generation is based on cardinality restrictions and on all the property characteristics (functional, inverse functional, transitive, symmetric, asymmetric, reflexive and, irreflexive).

Finally, the main RDBMS targeted are PostgreSQL, Oracle, MySQL, and MSSQL. Method [28] supports multiple RDBMS and method [29] uses an ontology database system called OntoDB.

3) Conversion process criteria: The conversion process is unique to each method. The methods differ in terms of the sequence of steps or conversion rules. A noteworthy point is

that all methods only deal with simple axioms or do not give an explicit indication of the treatment of composite axioms.

Type conversion: few methods define conversion of ontology types to SQL types (4/10). In the case where a definition is presented, the conversion often depends on an internal configuration or a specific RDBMS. Thus, manual adaptation is needed to reuse the generated relational database in different RDBMS. In addition, some ontology-type does not have a direct mapping to SQL-type, a more advanced mapping mechanism is required.

Restriction conversion: for the majority of the methods (9/10) the participation applicable to an object property in an axiom are converted into referential keys. In addition, a few methods (2/10) generate general constraints (explicit conversion) and some of the methods (4/10) keep information about the participation in metadata tables (implicit conversion). The latter case implies that cardinalities are not verifiable by the RDBMS unless there are constraints or automatisms that take advantage of them. However, these constraints or automatisms are not generated by these methods. In this case, the RDBMS cannot guarantee intrinsically the integrity of the data as described by the ontology. Yet, the integrity of the database can be evaluated externally if the information is present in the metadata, and internally if explicit constraints were generated.

Individual conversion: individuals are converted into tuples by five methods (5/10) so the initial database schema can contain tuples. This is an interesting feature as the database can be ready for querying. In addition, to benefit from it tuples reversibility can be valuable.

Annotation conversion: annotations are rarely used by methods (2/10) to document the database or the automate some conversion rules. Thus, a complementary mechanism is needed to fetch information from the ontology and the relational database to take full advantage of semantics.

Structure reversibility: none of the methods defines the reversibility explicitly. But, according to the overall conversion process, structure reversibility can be easily defined by six of the methods. Only two of the six methods present algorithms for this use.

Tuples reversibility: none of the methods defines the reversibility explicitly. But, according to the overall conversion process, tuples reversibility can be easily defined by five of the methods. Three over five methods present algorithms for this use.

4) Implementation criteria: Most methods have an implementation (7/10) but unfortunately only one is publicly available [30] and its use is limited to one ontology. Furthermore, the tool evaluation is rarely detailed which made it difficult to have a clear view of the evaluation scale.

IV. Discussion

Ontologies are used as knowledge models to define axiomatically the application domain while the relational schema is used as a logical data model to store, modify, and retrieve in a secure way a large amount of data. In the context of data integration, the role of ontologies is twofold. First, an ontology presents a consensual knowledge model [31], thus more suitable for semantic interoperability and for defining a unique access point. Secondly, an ontology offers a formal definition of the database enabling automatic structure and data consistency verification (that can be done automatically by most of the reasoners). Both the ontological and the relational model must be used together to take advantage of the abstraction and expressiveness of ontologies as well as the operational functionalities of databases built using relational schema [16].

As illustrated here, more work is still needed to improve the overall database quality especially to consolidate the schema integrity. First, the relational schema must be normalized and general constraints must be defined based on the cardinality in the axiom. This implies better management of missing information, and participation of individuals according to the property. In addition, the normalization is even more important in the context of physical data warehousing (especially for temporal database and big data) or virtual data warehousing (mediation) where the data extracted from multiple sources are heterogeneous, highly fragmented and context dependent [17], [32]. A “high” normal form (like 5NF and 6NF) reduce uncontrolled redundancy and facilitate schema extension as every part of the schema represents one predicate [33]. Even more, the resulting structure (after normalization) is closer to the set-theory foundation of ontologies and databases, thus facilitating query formulation [34] as the query formulated using ontology entities can have a more direct mapping. Second, preserving axiom expressivity is important to guarantee lossless conversions. Some ontologies may be built with simple axioms definitions but in the biomedical field, this hypothesis is not guaranteed, and as mentioned earlier, multiple ontologies in the OBO foundry use this approach. Describing conversion rules according to axioms definition is, therefore, an important aspect to cover ontologies produced in a large number of domains. Third, the structure of relations in a database differs from the structure of ontological entities. "Ontological" views can be generated to provide a view of the ontology structure. Even more, through these views, modification procedures can be generated to provide better access and standardize data manipulation. Therefore, a method that allows the generation of views and modification procedures can be very beneficial for application development in the sense that data access and modification can be handled at the database level. Fourth, another interesting advantage of using ontologies is the capability to document entities using annotation. However, the methods do not use annotations to document the relational schema (actually documentation is rarely available). For example, ontology label annotation can be used to define different views with different languages increasing the schema accessibility. Finally, regarding ontology-SQL type compatibility, when the RDBMS allows it, domains must be created for ontology types to make it easier to change types during the life cycle of the database.

Maybe more importantly, knowledge and schema evolution are not mentioned in any paper. This subject may be out of the scope of the selected paper, but it is an important one to ensure a long-term solution. Sustainability is a challenge faced by

every system, and explicit approaches to address this challenge are required.

Regarding structure reversibility and tuples reversibility, interesting work has been done in the reversibility aspect. That means that the methods are used in a more global context such as ontology data access, data acquisition, and data extraction.

Finally, what slowed down the adoption of ontologies is the lack of publicly accessible tools. Ontologies are by their very nature (shared understanding of a domain) public and shared resources. Thus, to encourage the reuse of ontologies or the use of common database schemas, tools must be publicly accessible and usable with multiple ontologies. Once this is the case, further work to assert that the method is usable at a certain scale will require several evaluations (tests) that are accepted by the community, easily used and reproducible.

V. Conclusion

Ontologies are definitely starting to be "popular" and they are now used in different forms. They are indeed a very promising approach to bring formal semantic to relational databases in order to enhance data interoperability. Nevertheless, in the context of using ontologies to generate a relational database, many issues remain to maintain the full expressivity of ontologies, among them: preserving property cardinalities and axiom expressivity, handling knowledge and schema evolution, handling missing information, and maintaining schema documentation. Complementary, relational approaches have much potential for bringing performance and power to ontology-based data operations. Finally, more work must be done to decreases the gap between ontologies and relation databases but the work presented is a step in this direction.

Acknowledgment

We thank the authors of the selected papers that revised the evaluation table, especially colleagues who took the time to reply and greatly enhance the scientific value of this work. We also thank master student Samuel Dussault for his help to fill the evaluation table. Finally, we would like to acknowledge the important support of the Unité de Soutien SRAP du Québec.

References

[1] P. P.-S. Chen, "The Entity-relationship Model—Toward a Unified View of Data," ACM Trans Database Syst, vol. 1, no. 1, pp. 9–36, 1976.

[2] OMG, "Unified Modeling Language Specification Version 2.0," Object Management Group, 2005. [Online]. Available: http://www.omg.org/spec/UML/2.0/. [Accessed: 31-Jan-2018].

[3] R. Elmasri and S. Navathe, Fundamentals of database systems, 6th ed. Boston: Addison-Wesley, 2011.

[4] D. Calvanese, G. D. Giacomo, D. Lembo, M. Lenzerini, R. Rosati, and M. Ruzzi, "Using OWL in data integration," pp. 397–424, 2010.

[5] G. D. Giacomo, D. Lembo, M. Lenzerini, A. Poggi, and R. Rosati, "Using Ontologies for Semantic Data Integration," in A Comprehensive Guide Through the Italian Database Research Over the Last 25 Years, Springer, Cham, 2018, pp. 187–202.

[6] T. Gruber, "Ontology," Encyclopedia of Database Systems. Springer US, Boston, MA, pp. 1963–1965, 2009.

[7] A. Poggi, D. Lembo, D. Calvanese, G. De Giacomo, M. Lenzerini, and R. Rosati, "Linking Data to Ontologies," J. Data Semant. X, vol. LNCS, no. 4900, pp. 133–173, 2008.

[8] B. Motik, P. F. Patel-Schneider, and B. Parsia, "OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax (Second Edition)," 2012. [Online]. Available: https://www.w3.org/TR/2012/REC-owl2-syntax-20121211/. [Accessed: 03-Apr-2016].

[9] A. Gali, C. X. Chen, K. T. Claypool, and R. Uceda-Sosa, "From Ontology to Relational Databases," in Conceptual Modeling for Advanced Application Domains, S. Wang, K. Tanaka, S. Zhou, T.-W. Ling, J. Guan, D. Yang, F. Grandi, E. E. Mangina, I.-Y. Song, and H. C. Mayr, Eds. Springer Berlin Heidelberg, 2004, pp. 278–289.

[10] D. Krneta, V. Jovanovic, and Z. Marjanovic, "A direct approach to physical Data Vault design," Comput. Sci. Inf. Syst., vol. 11, no. 2, pp. 569–599, 2014.

[11] J. Lee et al., "SAP HANA distributed in-memory database system: Transaction, session, and metadata management," 2013, pp. 1165–1173.

[12] D. J. Abadi, A. Marcus, S. R. Madden, and K. Hollenbach, "SW-Store: a vertically partitioned DBMS for Semantic Web data management," VLDB J., vol. 18, no. 2, pp. 385–406, 2009.

[13] A. A. Tzacheva, T. S. Toland, P. H. Poole, and D. J. Barnes, "Ontology Database System and Triggers," in Advances in Intelligent Data Analysis XII, 2013, pp. 416–426.

[14] I. Astrova, N. Korda, and A. Kalja, "Storing OWL ontologies in SQL relational databases," Int. J. Electr. Comput. Syst. Eng., vol. 1, no. 4, pp. 242–247, 2007.

[15] L. Al-Jadir, C. Parent, and S. Spaccapietra, "Reasoning with large ontologies stored in relational databases: The OntoMinD approach," Data Knowl. Eng., vol. 69, no. 11, pp. 1158–1180, 2010.

[16] D.-E. Spanos, P. Stavrou, and N. Mitrou, "Bringing Relational Databases into the Semantic Web: A Survey," Semant Web, vol. 3, no. 2, pp. 169–209, 2012.

[17] C. Khnaisser, L. Lavoie, H. Diab, and J.-F. Ethier, "Data Warehouse Design Methods Review: Trends, Challenges and Future Directions for the Healthcare Domain," in New Trends in Databases and Information Systems, T. Morzy, P. Valduriez, and L. Bellatreche, Eds. Springer International Publishing, 2015, pp. 76–87.

[18] C. Pinkel et al., "RODI: A Benchmark for Automatic Mapping Generation in Relational-to-Ontology Data Integration," in The Semantic Web. Latest Advances and New Domains, F. Gandon, M. Sabou, H. Sack, C. d'Amato, P. Cudré-Mauroux, and A. Zimmermann, Eds. Springer International Publishing, 2015, pp. 21–37.

[19] F. Baader, Ed., The description logic handbook: theory, implementation, and applications, 2. ed., paperback ed. Cambridge: Cambridge Univ. Press, 2010.

[20] E. F. Codd, "A Relational Model of Data for Large Shared Data Banks," Commun. ACM, vol. 13, no. 6, pp. 377–387, 1970.

[21] H. Darwen and C. J. Date, "The Third Manifesto," SIGMOD Rec, vol. 24, no. 1, pp. 39–49, 1995.

[22] H. Darwen, "How to Handle Missing Information without Using NULL," Warwick University, 2005.

[23] C. J. Date and H. Darwen, Database Explorations: essays on the Third Manifesto and related topics. Trafford Publishing, 2010.

[24] OBO, "The Open Biological and Biomedical Ontology Foundry," The OBO Foundry, 2018. [Online]. Available: http://www.obofoundry.org/. [Accessed: 06-May-2018].

[25] C. Curino, G. Orsi, E. Panigati, and L. Tanca, "Accessing and Documenting Relational Databases through OWL Ontologies," in Flexible Query Answering Systems, 2009, pp. 431–442.

[26] B. Motik, B. Cuenca Grau, I. Horrocks, Z. Wu, A. Fokoue, and C. Lutz, "OWL 2 Web Ontology Language Profiles (Second Edition)." [Online]. Available: https://www.w3.org/TR/owl2-profiles/#OWL_2_RL. [Accessed: 10-May-2018].

[27] S. Achpal, V. Bannihatti Kumar, and K. Mahesh, "Modeling Ontology Semantic Constraints in Relational Database Management System," presented at the IMECS International Multiconference of Engineers and Computer Scientists, Hong Kong, 2016.

[28] T. Podsiadły-Marczykowska, T. Gambin, and R. Zawiślak, "Rule-Based Algorithm Transforming OWL Ontology Into Relational Database," in Beyond Databases, Architectures, and Structures, 2014, pp. 148–159.

[29] L. Bellatreche, Y. Ait-Ameur, and C. Chakroun, "A design methodology of ontology based database applications," Log. J. IGPL, vol. 19, no. 5, pp. 648–665, 2010.

[30] T. Hornung and W. May, "Experiences from a TBox Reasoning Application: Deriving a Relational Model by OWL Schema Analysis," in Proceedings of the 10th International Workshop on OWL: Experiences and Directions (OWLED 2013) co-located with 10th Extended Semantic Web Conference (ESWC 2013), Montpellier, France, May 26-27, 2013, 2013, vol. 1080.

[31] G. Pierra, "Context Representation in Domain Ontologies and Its Use for Semantic Integration of Data," in Journal on Data Semantics X, Springer, Berlin, Heidelberg, 2008, pp. 174–211.

[32] N. Golov and L. Rönnbäck, "Big Data normalization for massively parallel processing databases," Comput. Stand. Interfaces, vol. 54, Part 2, pp. 86–93, Nov. 2017.

[33] C. J. Date, Database Design & Relational Theory. Sebastopol, Calif.: O'Reilly Media, 2012.

[34] P. LePendu, D. Dou, Z. M. Ariola, and C. Wilson, Ontology-based Relational Databases. 2007.

[35] D. Dou, H. Qin, and P. Lependu, "OntoGrate : Towards Automatic Integration For Relational Databases And The Semantic Web Through An Ontology-Based Framework," Int. J. Semantic Comput., vol. 04, no. 01, pp. 123–151, 2010.

[36] D. d B. Saccol, T. d C. Andrade, and E. K. Piveta, "Mapping OWL ontologies to relational schemas," in 2011 IEEE International Conference on Information Reuse Integration, 2011, pp. 71–76.

[37] E. Vyšniauskas, L. Nemuraitė, and B. Paradauskas, "Preserving Semantics of Owl 2 Ontologies in Relational Databases Using Hybrid Approach," Inf. Technol. Control, vol. 41, no. 2, pp. 103–115, 2012.

[38] E. Jiménez-Ruiz et al., "BootOX: Practical Mapping of RDBs to OWL 2," in The Semantic Web - ISWC 2015, 2015, pp. 113–132.

[39] L. T. T. Ho, C. P. T. Tran, and Q. Hoang, "An Approach of Transforming Ontologies into Relational Databases," in Intelligent Information and Database Systems, 2015, pp. 149–158.

[40] H. Afzal, M. Waqas, and T. Naz, "OWLMap: Fully Automatic Mapping of Ontology into Relational Database Schema," Int. J. Adv. Comput. Sci. Appl. IJACSA, vol. 7, no. 11, 2016.

APPENDIX I

The table below (Table I) presents the evaluation result of the criteria and of the requirement for the selected paper respectively. The value "?" means that the information was not found in the article. The evaluated papers are:

A1. Dou et al. [35]
A2. Bellatreche et al. [29]
A3. Saccol et al. [36]
A4. Vyšniauskas et al. [37]
A5. Hornung and May [30]
A6. Podsiadły-Marczykowska et al [28]
A7. Jiménez-Ruiz et al. [38]
A8. Ho et al. [39]
A9. Afzal et al. [40]
A10. Achpal et al. [27]

TABLE I. CRITERIA EVALUATION TABLE

	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
Ontology										
Ontology language	OWL ?	OWL ?	OWL ?	OWL ?	OWL DL	OWL DL	OWL QL,RL,EL	OWL ?	OWL DL	OWL DL
Schema										
Structure	? S	BCNF S	3NF S	BCNF S	BCNF S	3NF S	BCNF S	? S	? S	? G
Domains	No	No	No	No	No	No	No	No	No	No
Primary Keys	G	C	G	G	C	G	C	G	G	G
Secondary Key	No	?	No	Yes	?	?	Yes	?	?	Yes
Foreign keys	?	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Participation constraints	?	?	No	No	No	No	Yes	No	No	Yes
General constraints	?	?	Yes	Yes	?	Enum	Yes	Enum	No	Yes
Modification procedure	No	No	No	No	No	No	No	No	No	Yes
Target DBMS	?	OntoDB	PostgreSQL	?	?	Multi	?	MySQL	MSSQL	?
Process										
Axiom normalization	?	?	No	?	?	No	No	?	?	?
Intermediate Structure	FOL	MOF	No	OWL W3C	RDF	No	?	Jena	Jena	?
Type conversion	?	?	No	?	?	?	Yes	Yes	Yes	?
Restriction conversion	?	I	No	I	No	No	E*	I	I	E
Individual conversion	Yes	No	No	No	Yes	No	Yes	Yes	No	Yes
Annotation conversion	No	No	No	Yes	Yes	No	No	No	No	No
Structural reversibility	Yes Yes	Yes ?	No No	Yes ?	No No	? ?	No No	Yes No	Yes ?	? ?
Tuples reversibility	Yes Yes	Yes ?	No No	Yes ?	Yes Yes	? ?	Yes Yes	? ?	? ?	? ?
Tool										
Existence	Yes	Yes	Yes	?	No	Yes	Yes	Yes	Yes	?
Availability	No	?	?	?	Yes*	?	?	?	?	?

A5. Tool availability: The tool is a web application and works with Mondial ontology: http://www.semwebtech.org/rdf2sql/

A7. Restriction conversion: CHECK constraints are generated according to individual values