[Karunaratna *, Vol.6 (Iss.11): November 2018] ISSN- 2350-0530(O), ISSN- 2394-3629(P)
(Received: October 28, 2018 - Accepted: November 29, 2018) DOI: 10.29121/granthaalayah.v6.i11.2018.1123

INTERNATIONAL JOURNAL OF RESEARCH – GRANTHAALAYAH
A knowledge Repository

Science

BUILDING ONTOLOGIES OVER RELATIONAL DATABASES

Damitha D Karunaratna [*1]
[*1] University of Colombo School of Computing, Colombo, Sri Lanka

Abstract

Relational Databases are typically created to fulfil the information requirements of a community of users generally belongs to a single organization. Data stored in these databases were typically accessed by using Structured Query Languages or through customized interfaces. With the popularity of the World Wide Web and the availability of large number of Relational Databases for public access there is a need for users to retrieve data from these databases by using a text-based queries, possibly by using the terms that they are familiar with. However, the inherent limitations of Structured Query Languages used to create and access data in relational Data Bases does not allow uses to access data by using text-based queries. Also, the terms used in queries should be limited to those used during the construction of the databases. This paper proposes an architecture to generated ontologies over relation databases and show how they could be enhanced semantically by using available domain-specific or top-level ontologies so that the data managed by the DBs can be accessed by using text-based queries. The feasibility of the proposed architecture was demonstrated by building a prototype system over a sample MySQL database.

Keywords: Ontologies; Databases; Reverse Engineering; SQL; World Wide Web.

Cite This Article: Damitha D Karunaratna. (2018). "BUILDING ONTOLOGIES OVER RELATIONAL DATABASES." *International Journal of Research - Granthaalayah*, 6(11), 254-265. https://doi.org/10.29121/granthaalayah.v6.i11.2018.1123.

1. Introduction

Today large volumes of data are managed in relations databases (RDBs). These RDBs are created by different organizations for different purposes and accessed through the Structured Query Language (SQL) provided by the database management systems (DBMSs) or through customized interfaces build over the databases (DBs). With the popularity of the World Wide Web (WWW), there is an increasing trend to provide Hyper Text Markup Language (HTML) form-based interfaces over DBs to provide data through web pages. When using SQL to access data in a DBs users should have a fair knowledge on the syntax and semantics of the DB content. However, complete knowledge of RDBs is usually unavailable with the DBs (Astrova, 2005), thus cannot be extracted from the DBs. When using web-based interface users are limited to the data provided by the interfaces and they cannot extract and combine data as they required from the available DBs. Thus, different approaches are proposed to link DBs with different types of ontologies to enrich the semantics so that the DB contents can be understood easily (Astrova, 2005; Cullot, Ghawi, & Yetongnon, 2007; Karunaratna, Gray, & Fiddian, 1998a; Karunaratna, Gray, & Fiddian, 1998b; Zhou, Ling, Han, & Zhang, 2010; Alonso-Calvo, et al., 2007) and can be queried by using languages other than SQL. In this context, ontologies are considered as semantically rich knowledge structures that have the potential to enrich the semantics of data and meta-data in data repositories.

In this paper, we present a mechanism to construct ontologies over RDBs in third normal form. The proposed mechanism is based on a set of rules. Also, the mechanism make use of ontologies to enrich the constructed ontologies so that the users can query the content of RDBs by using simple text-based queries. The feasibility of the proposed mechanism for building ontologies over the existing RDBs is then demonstrated by developing a prototype system.

The rest of the paper is organized as follows. Section 2 describes briefly, the major differences and limitation of databases and ontologies and then presents a selected collection of approaches reported to link databases with ontologies. In section 3 the overall architecture of the proposed system and the rules proposed to construct ontologies over RDBS were presented. Section 4 elaborates on how the porotype system was constructed to demonstrates the feasibility of the proposed architecture. Finally, section 5 presents the conclusions and future enhancements.

2. Background

2.1. Differences Between Databases and Ontologies

Database and Ontologies are constructed by using different approaches and are used by different community of users for different objectives (Martinez-Cruz, Blanco, & Vila, 2012; Sir, Bradac, & Petr, 2015; Studer, Benjamins, & Fensel, 1998). The main intension of DBs to store and manage large volumes of data efficiently while ontologies are targeted at sharing knowledge of selected domains in a formal and structural manner among large community of users (Gruber , 1993; Guarino, 1995). Also, different languages and methods were developed to construct DBs and ontologies. For example, DBs are constructed and queried by using SQL whereas ontologies are constructed and queries by using a logic based language such as description logic.

It is important to understand the evolution of databases in order to understand the limitations of relational database technology. The Database Management Systems (DBMSs) available in late sixties and early sentries did not provide a sharp distinction between the logical view of the data and the physical representation of data in storage. As a result, application programmers were burdened with knowing irrelevant details on how data are organized and stored to develop applications over the DBs. This requirement sharply reduces the productivity of application programmers. The Relational Database Management Systems (RDBMS) was proposed by E. F Code (Codd, 1982) as a solution to this problem. Thus, the main objective of the RDBMS is to increase the productivity of application programmers within an organization who uses a RDB to develop applications. Consequently, much emphasis was not placed on the semantic expressiveness of data elements managed in data dictionaries of RDBMSs since one of the main premises made during the DB construction is that its usage is limited to a user community within a single organization. During the DB design stage conceptual schemas such as ER diagrams or

UML are typically constructed to model the real world. These models are then used to create DB schemas by using SQL provided by the RDBMSs. The models like ER models are considered as intensional models (Guarino, 1998) at the conceptual level whereas DBs are considered as extensional models at the implementation level. In the process of converting a model at the intensional level to a model at the extensional level some explicit semantics associated with conceptual models would become implicit are some are lost. For example, the cardinality between entities in a ER diagram become implicit and name of relationships between entities are lost. Thus, the extensional models and the corresponding intensional models do not encode the same semantics. Also, there is no standard in assigning names for DB components. As a result, DB creators were free to assign their own invented names for tables and attributes in the DB schemas, which may not reflect the actual real-world entities and their properties they represent. In addition, new tables may be constructed to represent many-to-many relationships among entities in the ER models and such tables do not correspond to any real-world entities. These problems would cause serious limitations on usability, with respect to the semantics of DB elements, when the DBs are exposed to a larger community of users.

The study of ontologies initially started as a discipline in philosophy and integrated gradually into information technology during the last few decades (Sir, Bradac, & Petr, 2015). The word ontology has two different meanings based on the context on which it is used (Gómez-Pérez, Fernández-López, & Corcho, 2004; Guarino, 1995). In the first context, the term ontology refers to the study of things and their attributes and how the attributes belong to the things because of their very nature (Corazzon, 2017; Guizzardi, 2005). In the second context it is used to model the structure of a system formally. The system could be any area of interest that needs to be represented in a structural manner. In Computer Science the concept ontology is used in this latter context.

The main components of an ontology are concepts. A concept may be assigned with one or more names and may have linked with each other through relationships. The relationships give a structure for an ontology. Also, axioms may be assigned with both concepts and relationships. The names, relationships and axioms assigned for concepts together define the intended meaning of concepts in an ontology. Ability to represent data in languages such as OWL and RDF opens up the possibilities and reduces the cost and complexity associated with building ontology from scratch (Yu, 2007).

Since a DB schema is an explicit specification of a conceptualization it also can be considered as an ontology according to the definition given by Gruber (Gruber , 1993). However, a schema defines mostly the structure of a DB but not the semantics of the DB content formally as required by a typical ontology. Thus, a DB schema by itself cannot be used as an ontology to share knowledge of the DB content in a formal and structural manner among large community of users. This means that a DB schema is a primitive ontology with capability to define only some of the semantics of the DB content. Thus, if a RDB has to be shared among a large community of users with the intension of allowing them to query data by using their own queries two components must be provided. A knowledge layer such as ontologies on top of the BDs to expose the semantics of DB content to the intended user community and a user query to SQL query engine to convert the user queries to native SQL queries understood by the RDBMSs.

2.2. Linking Databases with Ontologies

The approaches reported in the literature aiming at linking databases with ontologies generally take two different approaches. In the first approach databases are mapped with existing ontologies whereas in the second approach a new skeletal ontology is built for the DB by constructing a data model for the DB through reverse engineering techniques and then enrich the data model into a new ontology. Some research adopted this approach were based only on the schema information while others have used both schema information as well as properties of data for this process. The former approach has several drawbacks. One of them is the semantic mismatch between available ontologies and DBs. Since different ontologies and DBs may have constructed by different communities for different purposes with different assumption linking them together would result in unacceptable semantic inconsistencies. The latter approach provides many benefits. Firstly, it allows meta-data extracted from the DBs to be semantically enriched prior to the construction of the ontology. Secondly, the ontology constructed contains only those concepts required to describe the semantics of the associated DB. Thus, many research work on linking DBs to ontologies had taken the second approach.

2.3. Similar Work

OntoFusion (Alonso-Calvo, et al., 2007), is a system based on the second approach. It was designed to provide unified access to multiple, heterogeneous biological and medical data sources that are publicly available over Internet by constructing ontologies over data sources by using terms from already available domain ontologies, and then unifying similar ontologies. Concepts in the domain ontologies are mapped with the data items in the data source so that data sources can be queried by using the concepts defined in the domain ontologies.

Datagenie (Zhou, Ling, Han, & Zhang, 2010) , is one of the earliest attempts reported to integrate a database with an Ontology. The ontologies of Datagenie are constructed by using the ontology construction toolkit Protégé (PROTÉGÉ, 2018). Consequently, terms used to describe the system are tightly coupled with the Protégé vocabulary. Datagenie was built as a plug-in and has the following simple rules to link DB elements with the components in an ontology.

Table 2.1: Mappings in DataGenie System

Relational database	**Ontology**
Table	Class
Column	Class property
Row	instance
Foreign Keys	Protégé instance pointers

The Datagenie is too simplistic since the generated ontology is very primitive with respect to its semantics and the names in ontology may not make sense to a larger community as the names of ontology items are taken directly from the names of schema items. Also, it is not realistic to map tables representing many-to-many relations to classes as representing real-world concepts. The project qualegDB (Astrova, Korda, & Kalja, 2007) reported a similar approach but it has additional rules to extract inheritance relationships between tables in relational databases.

The research reported by Karunaratna (Karunaratna, Gray, & Fiddian, 1998a; Karunaratna, Gray, & Fiddian, 1998b) has taken the latter approach. The research was aimed at extracting metadata from relation database schemas in a federation of DBs (Sheth & Larson A, 1990) by using SQL and convert and unify that data into a single Prolog knowledge base(KB) with predicates of arity 1 and 3. The predicates with arity 1 were used to represent the concepts defined by the tables in the relational databases and arity 3 predicates to represent relationships between tables in the DB. Prolog rules were used to represent domain constraints of the DBs. Then the resulting KB is manually enriched with the assistance of domain experts and exposed to the public as an unifying ontology over the DBs in the federation. The research reported in (Banu, Fatima, & Khan, 2011) also took a similar approach. However, in that research the final ontology generated was represented in Resource Description Framework (RDF).

Also, DB2OWL (Cullot, Ghawi, & Yetongnon, 2007) is an attempt to transform Database schemas to Ontologies. It categorizes the tables in the relational DB schema into three classes and then builds the appropriate component/s in the Ontology based on the classification as described below. Case 1: deals with tables which represent associations of rows in different tables. Such tables are ignored in constructing Ontology classes.

Case 2: focuses on tables which represent specialization. These tables are considered as subclasses in the Ontology of the tables from which they derive.

Case 3: is the default case, where the tables which do not meet the above two are converted into Ontology classes unconditionally.

The process of conversion is first done with tables belong to case 3 followed by case 2, and then in case 1. Keys (except keys involved in case 2) are converted to object type properties in the Ontology and the other columns to data type properties. This approach also assume that the vocabulary used by the DB constructors are sufficient enough to expose the semantics of related items. Also, many of the research reported ignore the fact that primary keys and foreign keys defined in a RDB may comprises of multiple attributes.

Most of the approaches reported in the literature do not specify the rules formally, that they have used to construct ontologies and the structure and the composition of the ontologies built. Also, no clear explanation is reported on how the mapping between the components of the ontologies and components of DBs are recorded so that they can be used in subsequent SQL generation to access data in the DBs. Thus, it is not easy to identify their strengths and weaknesses easily.

3. Methodology

The approach we propose to build an Ontology comprises of two phases namely Meta-data Extraction phase and Ontology Generation Phase as depicted in figure 4.1.

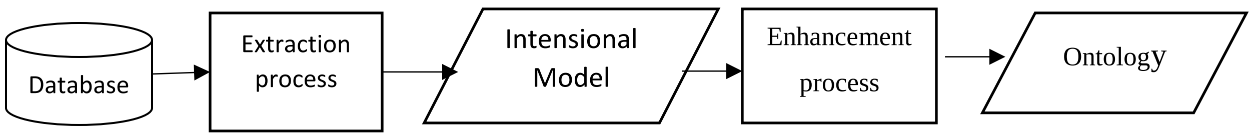

Figure 3.1: The Ontology building process

The extraction process is responsible for extracting the metadata from the relational database schema and to build a database model to represent the data in the DB. This process can be viewed as an attempt to build an intensional model, an intermediate ontology, from the extensional model described by the data in the DB. This process uses a set of rules to build an intensional model for a DB. In the next step the intermediate ontology created is enhanced with semantics by using an existing domain or a top-level ontology. The purpose of this stage is to ensure that the final ontology constructed is semantically rich enough to cater for a wide range of search words (to be entered by the user) than the meta-data items found in the database schema.

3.1. Construction of the Intentional Model

In our research an ontology O, is defined as below.

O = $\{C_1, C_2, \ldots, C_m\}$, where $C_1, C_2, \ldots, C_m$ are the concepts in the ontology. Each concept Ci is assigned with one or more names and may have zero or more attributes. Each attribute of a concept is assigned with one or more names and has a domain and a range. Concepts may be related with each other by using sub_class/super_class or named relationships. An ontology can be viewed as Figure 1.

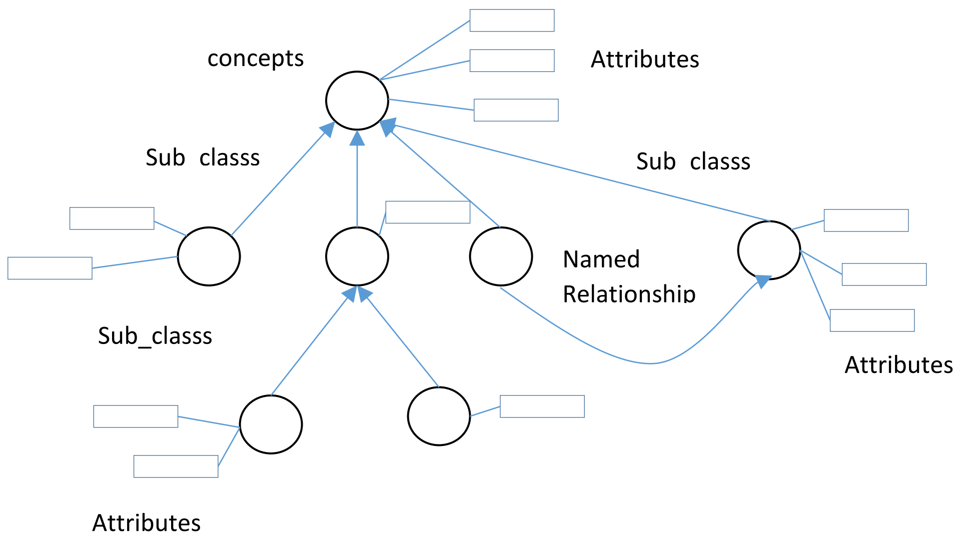

Figure 3.2: Representation of an ontology

The rules propose to construct the primitive ontology over a give database D are based on the following definitions and functions. The rules are coupled with a set of functions to record the mappings between generated components of the ontology with the corresponding components of the database. Such information is essential in generating SQL statements to retrieve data from the databases at a latter stage. The mapping information can be stored either in the DB or in a separate file in a variety of formats.

D = $\{T_1, T_2, \ldots\ldots, T_k\}$ be a database comprising tables with names $T_1, T_2, \ldots, T_k$.

$T_i = \{A_{i1}, Ai_2, \ldots\ldots, A_{ik}\}$ where A_{ij} is an attribute(or column) of table T_i.

- Name (A_{ij}) = Name of the attribute A_{ij}
- Domain (A_{ij}) = Domain of the attribute A_{ij}
- Range (A_{ij}) = Data type of the attribute A_{ij}
- PK (T_i) = $\{K_1, K_2,……..,K_p\} \subseteq \{A_{i1},A_{i2},……..,A_{ik}\}$
- FK (T_i) = $\{F_1, F_2,……..,F_q\}$ where $F_i \subseteq \{A_{i1},A_{i2},……..,A_{ir}\}$ and $\ni T_u$ such that $F_i = PK(T_u)$. FK (T_i) represents all foreign keys of the table T_i.
- If $F_k \in FK(T_i)$ then Source (F_k) = T_j where T_j is the name of the table in where F_k is the primary key.
- Add Concept(C) – The function adds a concept C to the ontology. It returns a unique id assigned for the concept in the ontology.
- Get Concept Id(C) – Returns the unique id assign to the concept with the name C if C exists, else it returns the value NULL
- Add Property (C, P, D, R) – A function which add a property with the name P to a concept with the name C with the domain D and range R. The function returns a unique id assigned to the property.
- Get Property Id (C, P) – Get the unique id assigned to the property with the name P of the concept with the name C if exists else it returns NULL.
- Link Concepts (C_i, C_j, R) – Links the concept C_i with C_j and assign the name R to the relationship. The constructed relationship is directional from C_i to C_j. The function returns a unique id generated for the relationship.
- Bind Concept (Concept Id, Table Name) – Binds a concept with a table name.
- Bind Relationship (Relationship Id, Relationship Name) – Binds a relationship with a name of a relationship.
- Bind Property (Concept Id, Property Id, Column Name) – Binds a concept property name with a database table column name
- This Bind function records how tables, columns and relationships in the databases are mapped with concept, concept properties and relationships in the ontology. such information are necessary to generate SQL to access data in the DB.
- Add Axiom (A, I) – Adds the axiom A to I where I is either a concept or a relationship.

Rule 1: Construction of concepts in the ontology.

For any $F_j \in FK(T_i) \nRightarrow Fj \subseteq PK(T_i)$ then

c = AddConcept (T_i)
BindConcept (c, T_i)
This rule says that a concept has to be created for every table that does not used to link multiple tables.

Rule 2: Construction of properties of concepts in the ontology.

Let T_i be the name of a concept in the ontology built by applying Rule 1. Then $T_i \in D$.

For each $A_{ij \in Ti}$ if $A_{ij} \notin FK(T_i)$ then

p = Add Property (Ti, Name (A_{ij}), Domain (A_{ij}), Range (A_{ij}))
Bind Property (Get Concept Id (T_i), p, Name (A_{ij}))

Rule 3: Construction of relations between concepts in the ontology.

If ∋ F_i and F_j such that $F_i \in FK(T_i) \wedge F_j \in FK(T_i) \wedge F_i \subseteq pK(T_i) \wedge F_j \subseteq pK(T_i)$ then

r_1 = LinkConcepts(Source(F_i),Source(F_j), Source(F_i).”_”.Source(F_j)) and
r_2= LinkConcepts(Source(F_j),Source(F_i), Source(F_j).”_”.Source(F_i))
BindRelationship(r_1, Source(F_i).”_”.Source(F_j))
BindRelationship(r_2, Source(F_j).”_”.Source(F_i))
where. represents the string concatenation operator.
This implies that connecting tables are used to construct relationships between two tables and the relationships are named by concatenating the names of the related tables.

Rule 4: Construction of inheritances.

If ∋ F_i such that $F_i \in FK(T_i) \wedge F_i = pK(T_i)$ then

LinkConcepts(T_i,Source(F_i), “subClassOf”) and
LinkConcepts(Source(F_i), T_i,“superClassOf”)

4. Implementation and Evaluation

In our prototype system Java programming language and SQL are used to implement the extraction process. The generated ontology is in OWL and the consistency of the generated ontology is checked by using the tool PROTÉGÉ (PROTÉGÉ, 2018). The general purpose top-level ontology WordNet (Fellbaum, 1998; MIller, Beckwith, Fellbaum, GRoss, & Miller, 1990) is used for the ontology enhancement process. The following figure 4.1 shows the meta-model of our ontology. The types of relationships defined between concepts are is_a, attribute and user named relationships. In our ontology attribute of tables are also represented as concepts. The attribute relationship is used to link attributes with their data types and is_a relationship represents sub-class/super-class relationship. Each concept and attribute are assigned with an optional id to point to the relevant synsets in the WordNet database. Currently identifying the relevant concepts from WordNet and assigning their ids with the corresponding concepts in the ontology is done manually.

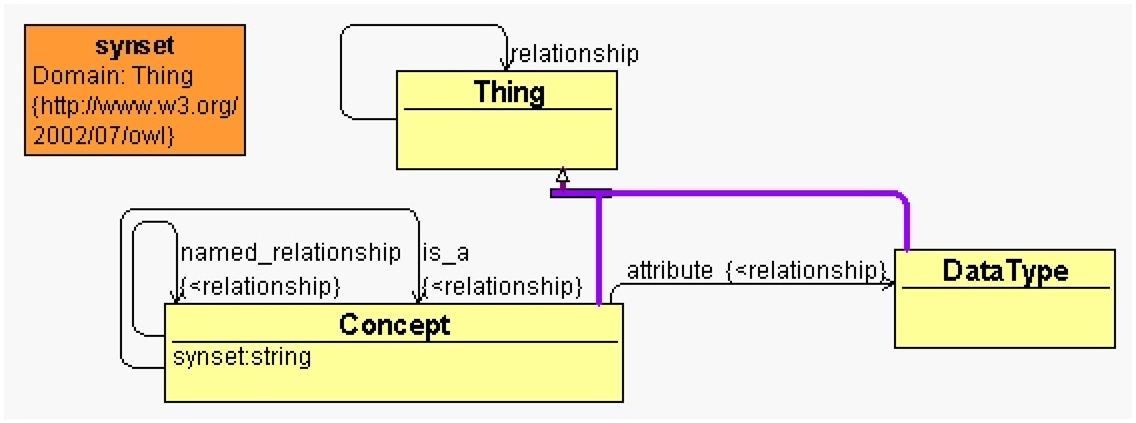

Figure 4.1: Meta model of the ontology

The following figure 4.2 shows the ER diagram of a simple database we have used to test our implemented prototype system and figure 4.3 shows the SQL command used to create the corresponding database in a MySQL database server.

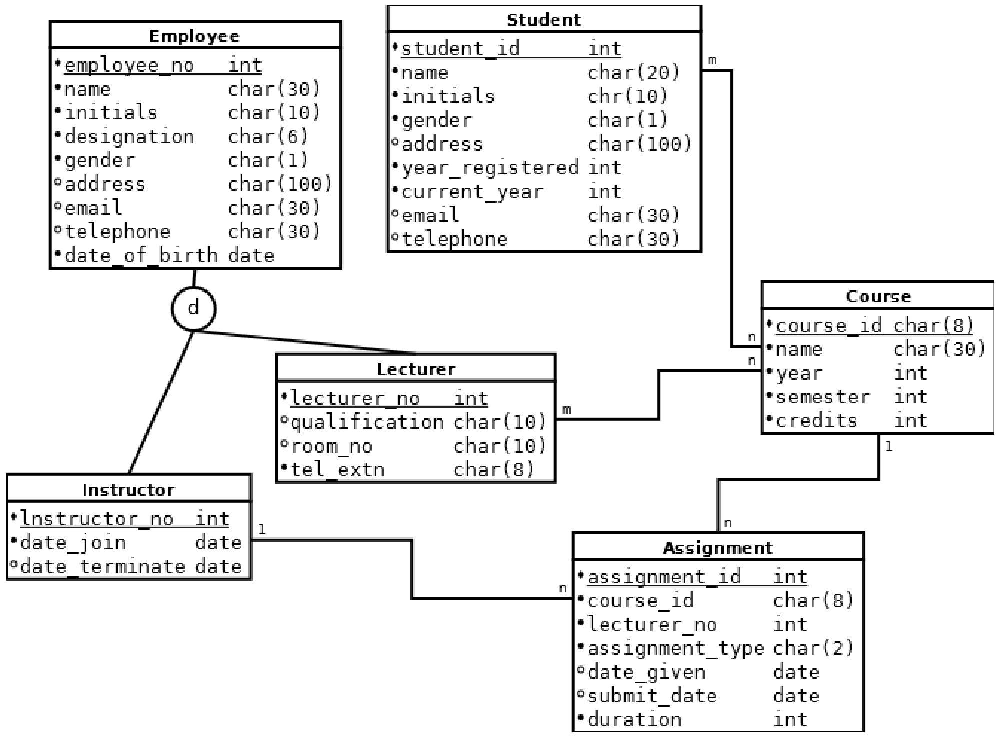

Figure 4.2: ER diagram of the system used for the prototype implementation

```
create table student(
student_id int NOT NULL
AUTO_INCREMENT,
name varchar(20),
initials varchar(10),
gender varchar(1),
address varchar(100),
year_registered date,
current_year int,
email varchar(30),
telephone varchar(20),
primary key(student_id)
);
```

```
create table employee(
employee_no int NOT NULL
AUTO_INCREMENT,
name varchar(30),
initials varchar(10),
designation varchar(6),
gender varchar(1),
address varchar(100),
email varchar(30),
telephone varchar(30),
date_of_birth date,
primary key(employee_no)
);
```

```
create table course(
course_id int NOT NULL
AUTO_INCREMENT,
name varchar(30),
year int,
semester int,
credits int(100),
primary key(course_id)
);

create table lecturer(
```

```
create table assignment(
assignment_id int NOT NULL
AUTO_INCREMENT,
course_id char(8) REFERENCES
course(course_id),
lecturer_no int REFERENCES
lecturer_no(lecturer_no),
assignment_type char(2),
date_given date,
submit_date date,
duration int,
lecturer_no int REFERENCES
employee(employee_no),
qualification varchar(10),
room_no varchar(10),
tele_ext varchar(8),
primary key(lecturer_no)
);

create table instructor(
instructor_no int REFERENCES
employee(employee_no),
date_join date,
date_terminate date,
primary key(instructor_no)
);
```

```
primary key(assignment_id)
);

create table instructor_assignment(
instructor int REFERENCES
instructor(instructor_no),
assignment int REFERENCES
assignment(assignment_id),
date_assigned date,
primary key(instructor,assignment)
);

create table lecturer_course(
lecturer int REFERENCES
lecturer(lecturer_no),
course int REFERENCES course(course_id),
date_assigned date,
primary key(lecturer,course)
);
```

Figure 4.3: SQL commands used to create the database in a MYSQL server

The concept hierarchy of the OWL ontology generated by our prototype system for the database is given in Figure 4.4.

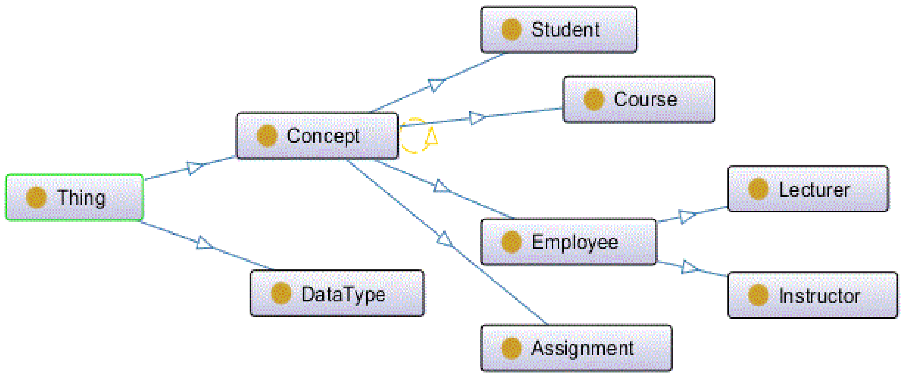

Figure 4.4: Concept hierarchy of the ontology generated by the prototype

5. Discussion and Future Work

The proposed solution was evaluated by building a prototype and applying it on a set of sample database schemas. The results of the evaluation show that the proposed system is technically feasible and has the potential to build semantically rich ontology on top of any relational DB. The ontologies built by the proposed approach can subsequently be used to support text-based queries. One of the main limitations of the proposed approach is the limitations in the enhancement process. Currently, only the synonyms of the WordNet are used to enhance the ontologies. We are exploring how hypernyms and hyponyms defined in WordNet could be integrated into the ontologies to further enhance the semantics of the generated ontologies. Also, during our research we found that the terms to represent the intended meanings of some the items (table names/attribute names) in the database could not be found in the WordNet.

References

[1] Alonso-Calvo, R., Maojo, V., Billhardt, H., Martin-Sanchez, F., Garcia-Remesal, M., & Pe'rez-Rey, D. (2007). An Agent- and Ontology-based System for Integrating Gene, Protein, and Disease Databases. Journal of Biomedical Informatics, 40(1), 17-29.
[2] An, J., & Young, B. (2018). Methodology for Automatic Ontology Generation Using Database Schema Information. Mobile Information Systems, 2018.
[3] Astrova, I. (2005). Towards the Semantic Web - An Approach to Reverse Engineering of Relational Databases to Ontologies. ADBIS research communications (152).
[4] Astrova, I., Korda, N., & Kalja, A. (2007). Rule-Based Transformation of SQL Relational Databases to OWL. Proceedings of the 2nd International Conference on Metadata & Sematic Research. Corfu Island,Greece.
[5] Banu, A., Fatima, S. S., & Khan, K. U. (2011). Semantic-Based Querying Using Ontology in Relational Database of Library Management System. International Journal of Web & Semantic Technology (IJWesT), 2(4).
[6] Codd, E. F. (1982). "Relational Databases: A Practical Foundation for Productivity". Communication of the ACM, 25(2).
[7] Corazzon, R. (2017). Notes on the History of Ontology, Descriptive and Formal Ontology: A Resource Guide to Contemporary Research. Retrieved 2018, from www.formalontology.it
[8] Cullot, N., Ghawi, R., & Yetongnon, K. (2007). DB2OWL: A Tool for Automatic Databse-to-Ontology Mapping. 15th Italian Symposium on Advanced Databse System, (pp. 491-494).
[9] Fellbaum, C. (Ed.). (1998). WordNet: An Electronic Lexical Database. Cambridge, USA: MIT Press.
[10] Genesereth, M. R., & Nilsson, N. (1987). Logical Foundations of Artificial Intelligence. CA, USA: Morgan Kaufmann.
[11] Gómez-Pérez, A., Fernández-López, M., & Corcho, O. (2004). Theoretical Foundations of Ontologies" in Ontological Engineering. Springer.
[12] Gruber, T. R. (1993). "A Translation Approach to Portable Ontology Specification". Knowledge Acquisition, 5(2), 199-220.
[13] Guarino, N. (1995). Formal Ontology Conceptual Analysis and Knowledge Representation. International Journal of Human-Computer Studies, 43(5), 625-640.
[14] Guarino, N. (1998). Formal Ontology and Information Systems. FOIS, (pp. 3-15). Trento, Italy.
[15] Guizzardi, G. (2005). Ontological Foundation for Structural Conceptual Models. Enschede, Netherlands: PhD Thesis, Center for Telematics and Information Technology.
[16] Karunaratna, D. D., Gray, W. A., & Fiddian, N. (1998a). Establishing a Knowledge Base to Assist Integration of Heterogeneous Databases. 16th British National Confetence on Databases. Cardiff, U.K.
[17] Karunaratna, D., Gray, W., & Fiddian, N. (1998b). A Knowledge Based Approach for Database Integration. 9th International Conference on Management of Data. Hyderabad, India.
[18] Martinez-Cruz, C., Blanco, I., & Vila, M. (2012). Ontologies Versus Relational Databases: Are they so different? Artificial Intelligence Review, 38(4), 271-290.
[19] MIller, G., Beckwith, R., Fellbaum, C., GRoss, D., & Miller, K. (1990). Introduction to WordNet: an on-line lexical database. International Journal of Lexicography, 3(4), 235-244.
[20] PROTÉGÉ. (2018). A free, open-source ontology editor and framework for building intelligent systems. Retrieved 09 2018, from https://protege.stanford.edu/

[21] Sheth, A., & Larson A, J. (1990). Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases. ACM Computing Surveys, 22(3), 183-236.
[22] Sir, M., Bradac, Z., & Petr, F. (2015). Ontology versus Databases. ScienceDirect, 48(4), 220-225.
[23] Studer, R., Benjamins, R., & Fensel, D. (1998). Knowledge Engineering: Principles and Methods. Data & Knowledge Engineering, 25(1), 161-197.
[24] Yu, L. (2007). Introduction to the Semantic Web and Semantic Web Services. Chapman \& Hall/CRC.
[25] Zhou, S., Ling, H., Han, M., & Zhang, H. (2010). Ontology Generator from Relational Database Based on Jena. Computer and Information Science, 3(2).

*Corresponding author.
E-mail address: ddk@ ucsc.cmb.ac.lk