Automatic Formulation of SQL GROUP BY and HAVING Statements using the Ontology Driven Relational Query Formulation Framework

By

Muhammad Sheraz Anjum

2011 NUST MS PhD-IT-053

Supervisor

Dr. Peter Bloodsworth

Department of Computing

A thesis submitted in partial fulfillment of the requirements for the degree of
Masters of Science in Information Technology (MS IT)
In
School of Electrical Engineering and Computer Science,
National University of Sciences and Technology (NUST),
Islamabad, Pakistan
(July, 2015)

Approval

It is certified that the contents and form of the thesis entitled **“Automatic Formulation of SQL GROUP BY and HAVING Statements using the Ontology Driven Relational Query Formulation Framework”** submitted by **Muhammad sheraz Anjum** have been found satisfactory for the requirement of the degree.

Advisor: **Dr. Peter Bloodsworth**

Signature:____________________________

Date: ___________________________

Committee Member1: **Dr. Kamran Munir**

Signature_____________________________

Date:________________________________

Committee Member2: **Dr. Sharifullah Khan**

Signature _____________________________

Date: ________________________________

Committee Member3: **Dr. Noman Javed**

Signature _____________________________

Date: ________________________________

Dedication

I dedicate this thesis work to my parents who from the start encouraged and supported me. Also to my sisters who are an endless source of motivation.

It is also dedicated to my friends and to my teachers with whom I have an exceptional and admirable relationship.

Certificate of Originality

I hereby declare that this submission is my own work and to the best of my knowledge it contains no materials previously published or written by any other person, nor material which to a substantial extent has been accepted for the award of any degree or diploma at NUST SEECS or at any other educational institute, except where due acknowledgement has been made in the thesis. Any contribution made to the research by others, with whom I have worked at NUST SEECS or elsewhere, is explicitly acknowledged in the thesis.

I also declare that the intellectual content of this thesis is the product of my own work, except for the assistance from others in the project's design and conception or in style, presentation and linguistics which has been acknowledged.

Author Name: **Muhammad Sheraz Anjum**

Signature: ___________________

Acknowledgment

First and foremost praises be to Allah, the Almighty, on whom ultimately we depend for direction and guidance.

I wish to express my deep sense of gratitude to my supervisors **Dr. Peter Bloodsworth** and **Dr. Kamran Munir** for giving me a chance to work with them, in a research environment and polish my skills.

Words are inadequate in offering my thanks to the members of advisory Committee **Dr. Sharifullah Khan** and **Dr. Noman Javed** for their valuable input whenever required. Finally, yet importantly, I would like to express my heartfelt thanks to my colleagues and fellow students who helped me directly or indirectly.

Table of Contents

List of Figures

List of Tables

Abstract

Patterns and relationships within medical data are of great importance for health professionals and clinical researchers. It is a challenging task for clinical researchers to extract interesting patterns from complex medical data, requiring users to develop specific skills in order to create complex database queries. An ontology driven architectural framework, namely OntoQF [1], has recently been proposed to assist users in creating relational database queries. OntoQF works in two phases. In the first phase, OWL ontology is created from relational schema and domain concepts are integrated into it to form domain ontology. In the second phase, domains experts are provided assistance in writing ontology based queries which are translated into equivalent relational queries. The OntoQF was based on OWL1-DL [2] whereas W3C (World Wide Web Consortium) has recently recommended a more expressive and powerful ontology language i.e. OWL2 [3]. OWL2 is an extended version of OWL providing support for more sophisticated ontology constructs and definition of rich data ranges. This work focuses on extending support for some more relational query constructs using selected novel features of OWL2-DL. A set of algorithms is proposed to translate object property cardinality restrictions and data property cardinality restrictions into equivalent relational query restrictions containing GROUP BY and HAVING clauses. Support for novel feature of OWL2 i.e. qualified cardinality restrictions is also provided by proposing an extension to the existing algorithms. In addition, extensions are also proposed to translate quantification restrictions on data properties into equivalent relational query restrictions. Previously, due to the limited expressive power of OWL, support for quantification restrictions was available for object properties only. At the end, all the generated constructs are joined together in a defined manner to create a correct relational query. The generated complex relational queries are validated against a set of test case from medical domain. In conclusion, this work has extended the functionality of OntoQF [1] by providing support for GROUP BY and HAVING constructs using OWL2-DL.

Chapter 1: Introduction

Tremendous amount of data about patient profiles and their medical observations is collected and stored on daily basis. This medical data is heterogeneous and complex in nature. The Patterns and the relationships within this data are of great importance for health professionals and clinical researchers. It's a challenging task to extract these patterns from medical data as specific skills are required for creating complex database queries. Health professionals are usually not well aware of structure of data stored in the databases. This makes it more difficult for the health professionals to write complex queries. The importance of effective and efficient medical knowledge discovery can be clearly noted from the following statement of a well-known expert:

"The effective and efficient application of known worldwide health and medical information and knowledge will have a bigger impact on health and disease than any drug or technology likely to be introduced in the next decade" [4] *(Sir Muir Gray, Director of Oxford University Institute of Health Sciences*).

Information technology has widely played its role to assist clinical researchers in extracting useful information from underlying data. One vital aspect is the provision of assistance in creating database queries to retrieve the needed information. In this regard, Query by example (QBE) and menu driven (MD) techniques are notable [5]. QBE and MD techniques are simple but are not as powerful as structured query languages. It is recently proposed that clinical researchers can be provided assistance in writing structured queries. The expressive power of ontology can be exploited to assist users in query formulation. Domain ontology can store information about structure of data and the domain knowledge at the same time. This capability makes domain ontologies a suitable candidate for the purpose.

An architectural framework, named OntoQF (otology-driven query formulation) [1], is recently proposed for ontology driven relation query formulation. To the best of our knowledge, OntoQF is the only existing technique that exploits assertion and semantic capabilities of description logic based domain ontologies. In OntoQF, domain ontology is used to store the domain knowledge and the information regarding structure of underlying data. This stored information can be used to assist users in formulating valid relational queries. Transactional data is not stored in the ontology which is a major advantage of this technique. The OntoQF was based on OWL1-DL[2] whereas W3C (World Wide Web Consortium) has recently recommended a more expressive and powerful ontology language i.e. OWL2 [3] (details discussed in section 2.5). The expressive power of OWL2 can be exploited to extend the OntoQF for improving its usefulness. The OntoQF supports the formation of relational queries with SELECT, FROM and WHERE clauses. The use of other relational query constructs such as GROUP BY and HAVING remains to be supported in the formulated query.

This thesis focuses on extending OntoQF by adding support for GROUP BY and HAVING constructs using OWL 2-DL. Support for OWL2 can enable definition of more sophisticated semantic restrictions and domain knowledge in the domain ontology that can be used to assist users in the formulation of more complex relational queries.

1.1 Research Aims

Structured queries are a powerful mean to extract required data from underlying databases. SQL (Structured Query Language) [6] is widely used to query over relational databases. Complex relational queries can be created to extract interesting patterns and relationships from the data. But users need to develop specific skills to create sophisticated relational queries. Users also need to have a clear understanding of underlying database schema in order to retrieve relevant and correct information. Complexity of the relational schema may vary depending upon the nature of the application. In case of a complex relational schema, it is more difficult to write correct relational queries to retrieve the required data. The medical data is typically very complex in nature. Clinical researchers often find it difficult to develop a clear understanding of the complex underlying schema. So it’s a very challenging task for the clinical researchers to write valid relational queries to extract interesting patterns [1].

This thesis focuses on extending OntoQF in order to assist the clinical researchers in formulating some further complex database queries using OWL2 to extract further interesting patters e.g. to retrieve the record of all the patients who frequently have high blood pressure or to retrieve the record of all the patients who use antibiotics in excess. The extended OntoQF will maintain the advantage of keeping transactional data in the underlying database.

The test cases to evaluate this extended system are same as used in OntoQF which are selected from a medical domain. The reasons for this selection as given in [1]: (1) medical data is typically very large and complicated in nature and complex relational queries can be of vital importance to extract interesting patterns from such data, (2) clinical researchers are often not well aware of the underlying database structure and hence they require assistance in formulating complex queries. Selecting the same domain for testing the extended version will allow a better comparison with the existing version. On the basis of aforementioned research aims, the research hypothesis and the questions needed to test that hypothesis are presented in the following section.

1.2 Research Hypothesis and Research Questions

The hypothesis of this research work states that:

"*Support of GROUP BY and HAVING relational query constructs and selected new constructs of OWL 2-DL can enhance the usefulness of Ontology Driven Relational Query Formulation Framework (OntoQF)*"

In order to test the above hypothesis, the following original research questions of OntoQF [1] have been extended to test the research hypothesis:

To what extent support of GROUP BY and HAVING *relational query constructs can increase the usefulness of OntoQF? (**Question 1**)*

An extended and revised version of OWL i.e. OWL2 is recently recommended by W3C.
OWL2 is more expressive and powerful ontology language and has many novel features including rich data types and qualified cardinality restrictions. The expressive power of OWL2 allows better modeling of real world entities and hence rich domain ontology can be constructed.

On the basis of aforementioned discussion, the following research question is designed to test the research hypothesis:

*To what extent support of selected new constructs of OWL 2-DL can increase the usefulness of OntoQF? (**Question 2**)*

In OntoQF, database to ontology mappings were used to transform semantic expressions into relational structures. These mappings play a major role in the formulation of executable relational queries. This research focuses on adding support for some of the new features of OWL2, which also need to be mapped to relational structures. To address this requirement, the following research question is updated:

What mappings are required between GROUP BY and HAVING *relational constructs and selected OWL2-DL constructs for query translation? (**Question 3**)*

The above mentioned formulated complex relational queries are to be empirically evaluated in order to check the correctness of the query formulation mechanism. To address this requirement, the following research question is designed:

*How can the correctness of new formulated queries be empirically evaluated? (**Question 4**)*

In conclusion, aim of this research work is to examine that how support for GROUP BY and HAVING relational query constructs using OWL 2-DL can enhance the usefulness of OntoQF. The research methodology adopted to study the research hypothesis with the devised research questions is presented in the following section.

1.3 Research Methodology

This thesis proposes some extensions to OntoQF [1] and the methodology adopted for this research largely remains the same as in OntoQF, which is depicted in Fig 1.1. The research started with the study of existing version of OntoQF. The state of the art versions of technologies, used in OntoQF, were studied to understand the domain. During the initial study, some areas were identified which had potential for further improvement. Based on the findings of initial study, a research hypothesis was defined. An updated set of research questions that were originally proposed in OntoQF was also devised in order to test the hypothesis. The devised research hypothesis and questions led to the next phase of detailed literature review. To answer the research questions, a detailed literature survey was carried out in the next phase. Then on the basis of the acquired domain knowledge, algorithms were proposed to add support of selected OWL2 constructs. In the next phase, the work was extended by proposing algorithms to add support of GROUP BY and HAVING relational query constructs. This paved the way for the next phase in which the mapping algorithm of OntoQF was extended to support new constructs. Then in the evaluation phase, the system was empirically evaluated. Based on the results gathered in the evaluation phase, the strengths and weaknesses of the extended system were identified. Finally, the research was concluded and the future directions were discussed.

1.4 Thesis Organization

After a detailed introduction to the research topic in the current chapter, this section discusses the research objectives of the coming chapters. Thesis organization is also depicted in Fig 1.2. Theoretical background of the research problem and a detailed literature survey are presented in **Chapter 2**. As this research focuses on extending OntoQF for further complex relational query formulation, this chapter mainly reviews the related work on ontology based query formulation techniques. This chapter also reviews the techniques and technologies which are used in the OntoQF and in particular novel features of OWL2-DL are reviewed. The literature survey concludes that the proposed research work is significant and novel.

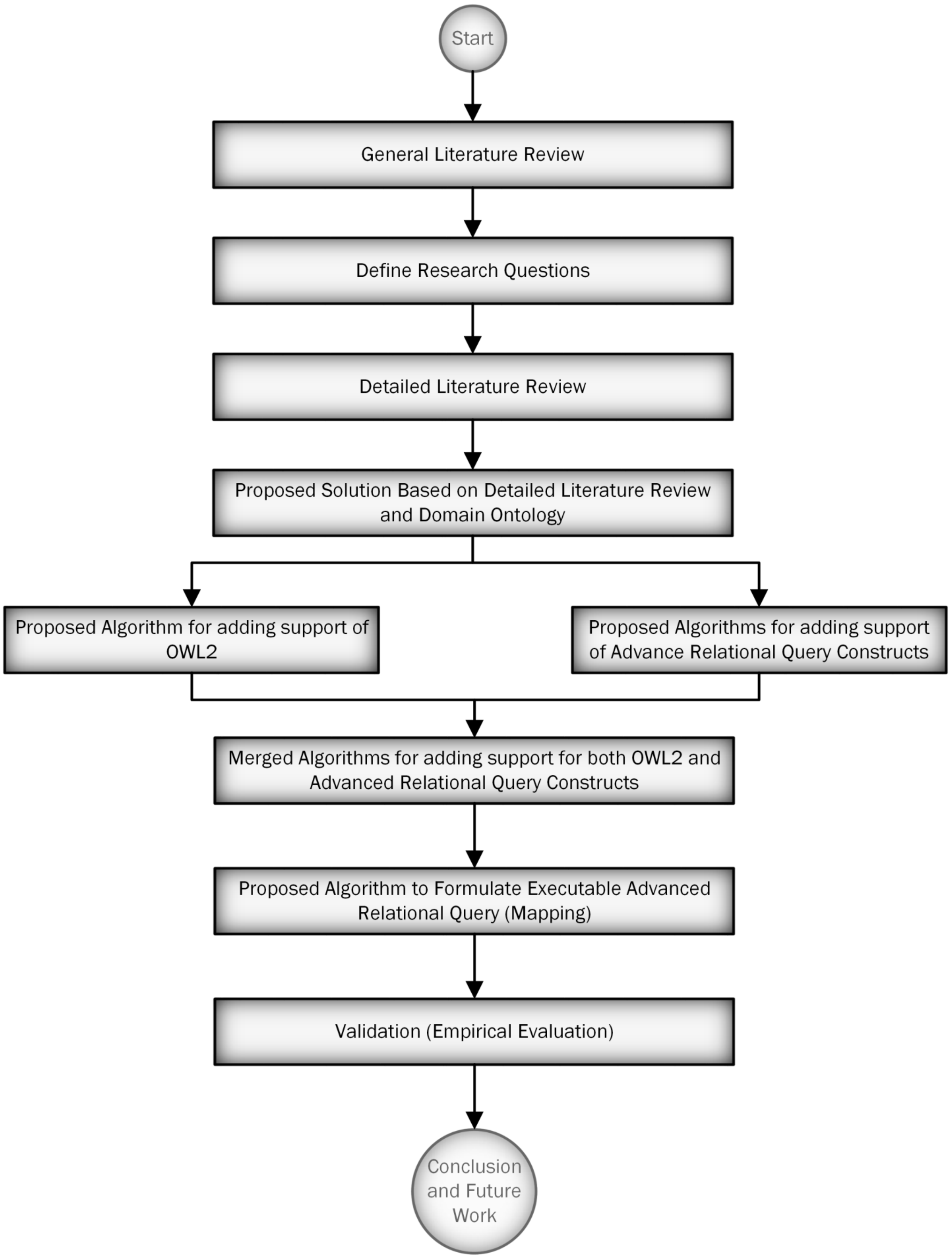

Start
General Literature Review
Define Research Questions
Detailed Literature Review
Proposed Solution Based on Detailed Literature Review and Domain Ontology
Proposed Algorithm for adding support of OWL2
Proposed Algorithms for adding support of Advance Relational Query Constructs
Merged Algorithms for adding support for both OWL2 and Advanced Relational Query Constructs
Proposed Algorithm to Formulate Executable Advanced Relational Query (Mapping)
Validation (Empirical Evaluation)
Conclusion and Future Work

Figure 1.1: Research Methodology (adopted from OntoQF [1])

In the wake of having reviewed the detailed literature, **Chapter 3** discuses that how support of GROUP BY and HAVING query constructs using OWL2-DL can increase the usefulness of OntoQF. This chapter also discusses the major changes that are required in the existing system to achieve the research aims. **Chapter 4** presents the updated algorithms for the formulation of executable relational queries from ontology statements. **Chapter 5** discusses the implementation of prototype that is used for the evaluation of proposed upgrades. This chapter also discusses the results of empirical evaluation. Finally, Chapter 6 concludes the research and a summary of the research outcomes are presented. Furthermore, this chapter suggests the future directions for making further refinements in the OntoQF.

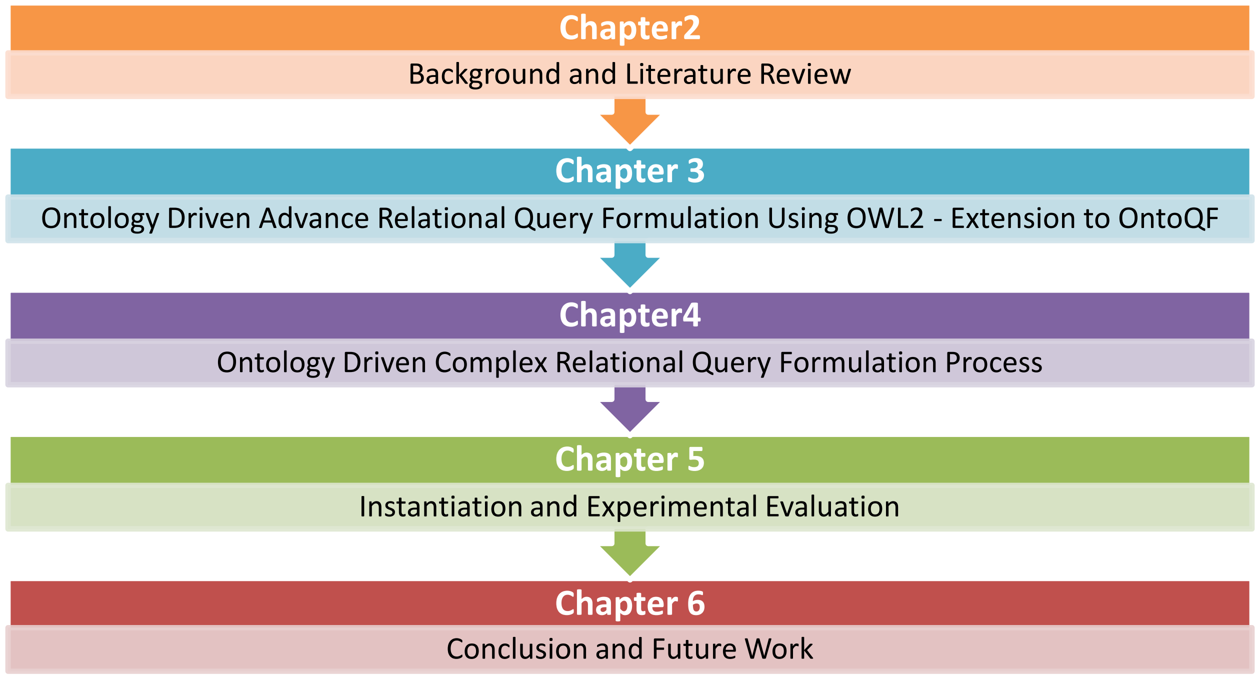

Chapter2
Background and Literature Review
Chapter 3
Ontology Driven Advance Relational Query Formulation Using OWL2 - Extension to OntoQF
Chapter4
Ontology Driven Complex Relational Query Formulation Process
Chapter 5
Instantiation and Experimental Evaluation
Chapter 6
Conclusion and Future Work

Figure 1.2: Thesis Organization

Chapter 2: Background and Literature Review

2.1 Introduction

In the last decade information retrieval and semantic based data modeling have been active research domains. Currently, there are many different types of database systems including relational database systems, multimedia databases systems and semantic based database systems. Data is getting bigger and bigger in size and diverse in nature. Retrieving data effectively from the immense collection of data has become a huge challenge. Many approaches have been proposed to retrieve semantic based information from such gigantic and heterogeneous databases. As a larger part of existing data is being stored in relational format, a key challenge is to convert relational data into semantic data for semantic based information retrieval. For this purpose, many approaches have been proposed for relational data to ontological data transformation. In case of pre existence of both relational database and semantic data definition, relational data to ontological data mapping is required for the transformation. The purpose of mapping may also be to create a wrapper in case of heterogeneous underlying database systems.

As a first step towards, adding meaning to the data, Resource Description Framework (RDF) had been proposed. RDF(S) was an XML based ontology language which enabled users to add meaning along with the structure of data[7]. Later on, a more expressive ontology language OWL (Web Ontology Language) was recommended by W3C in 2004 [2]. In 2009, OWL2 was recommended by w3C which is an even more expressive and powerful ontology language [3]. These ontology languages are compared in detail at the end of this chapter. This chapter also discusses the state of the art ontology based query formulation techniques, database to ontology formulation techniques, and database to ontology mapping techniques.

A roadmap to the literature review is shown in fig 2.1.The literature survey has been categorized into four groups.

a) **Tools and approaches for ontology based query formulation**

The existing ontology based query formulation techniques are discussed and compared in section 2.2. Query formulation is discussed in different scenarios including the cases of heterogeneous data sources and multimedia database.

b) Tools and approaches for database to ontology transformation

State of the art techniques for database to ontology transformation are discussed and compared in section 2.3.

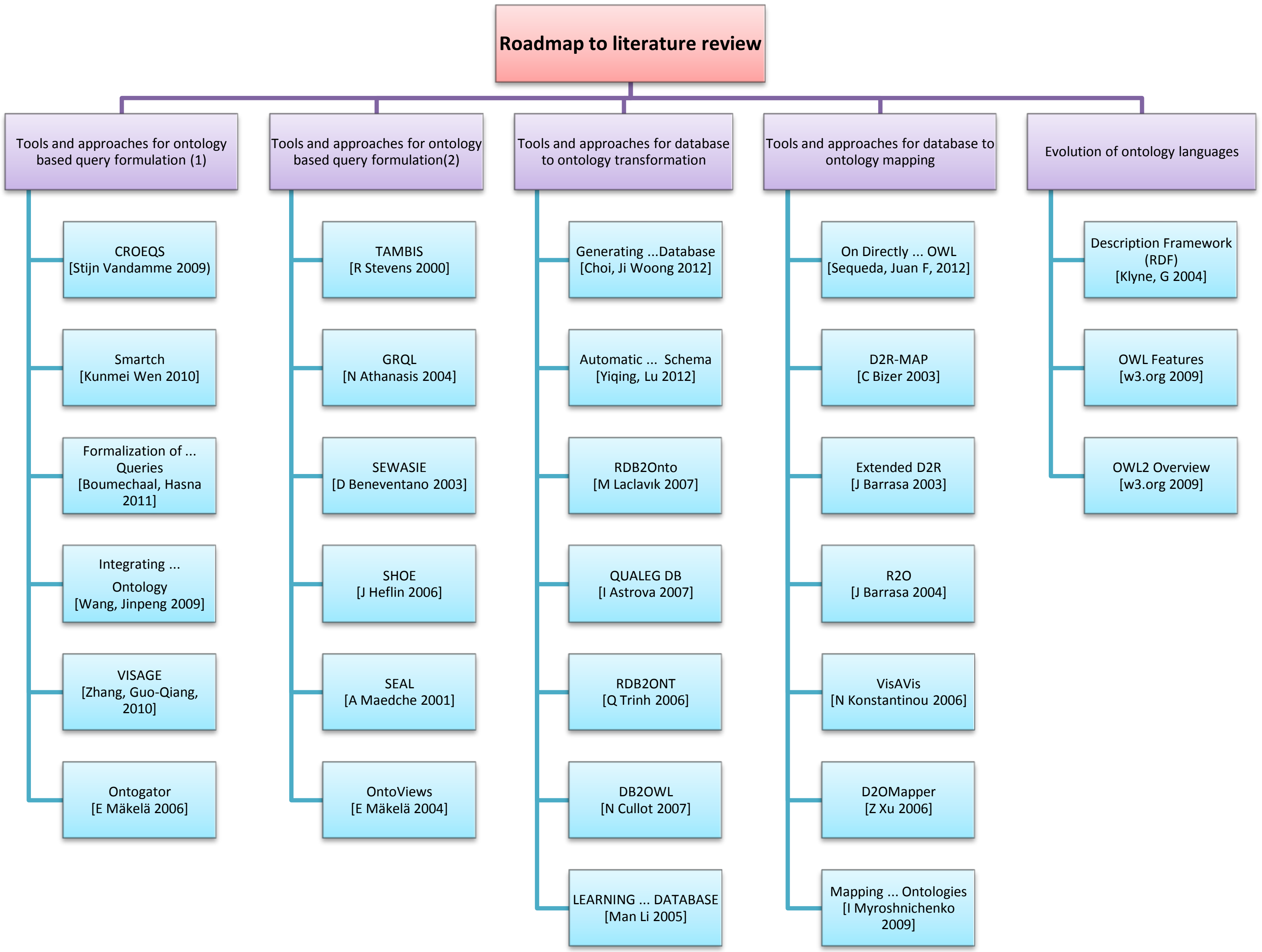
c) Tools and approaches for database to ontology mapping

State of the art techniques for database to ontology mapping are discussed and compared in section 2.4.

d) Evolution of ontology languages

Evolution of ontology languages and their comparison is presented in section 2.5.

In the end, chapter conclusions are discussed in section 2.5.



Roadmap to literature review
Tools and approaches for ontology based query formulation (1)
CROEQS [Stijn Vandamme 2009)
Smartch [Kunmei Wen 2010]
Formalization of ... Queries [Boumechaal, Hasna 2011]
Integrating ... Ontology [Wang, Jinpeng 2009]
VISAGE [Zhang, Guo-Qiang, 2010]
Ontogator [E Mäkelä 2006]
Tools and approaches for ontology based query formulation(2)
TAMBIS [R Stevens 2000]
GRQL [N Athanasis 2004]
SEWASIE [D Beneventano 2003]
SHOE [J Heflin 2006]
SEAL [A Maedche 2001]
OntoViews [E Mäkelä 2004]
Tools and approaches for database to ontology transformation
Generating ...Database [Choi, Ji Woong 2012]
Automatic ... Schema [Yiqing, Lu 2012]
RDB2Onto [M Laclavık 2007]
QUALEG DB [I Astrova 2007]
RDB2ONT [Q Trinh 2006]
DB2OWL [N Cullot 2007]
LEARNING ... DATABASE [Man Li 2005]
Tools and approaches for database to ontology mapping
On Directly ... OWL [Sequeda, Juan F, 2012]
D2R-MAP [C Bizer 2003]
Extended D2R [J Barrasa 2003]
R2O [J Barrasa 2004]
VisAVis [N Konstantinou 2006]
D2OMapper [Z Xu 2006]
Mapping ... Ontologies [I Myroshnichenko 2009]
Evolution of ontology languages
Description Framework (RDF) [Klyne, G 2004]
OWL Features [w3.org 2009]
OWL2 Overview [w3.org 2009]

Fig 2.1: Roadmap to Literature Review

2.2 Tools and Approaches for Ontology Based Query Formulation

Amount of data is increasing day by day and consequently effective searching is getting more and more difficult. Multimedia databases are usually bigger in size than traditional databases due to type of content that they store. In a multimedia database items are annotated with keywords and descriptions. *Stijn Vandamme* designed a semantic search engine named CROEQS [8] for multimedia databases. CROEQS consists of two major modules: (1) ontology based query translator, and (2) text based search engine. Domain ontology is developed for ontology based query translator and temporal information is added to the ontology. An Index on the textual metadata is developed by Text Based search engine.

Using CROEQS, a user can query using semantic clauses, keywords or a combination of both. Unlike pure text based search, the users can effectively apply constraints using semantic clauses too. Stijn Vandamme argues that if the time interval is short for which a temporal statement holds true then the temporal information can play a vital role in confining the result set size. Impact on result set size is analyzed when semantic clause is replaced by an equivalent string as a keyword and also when temporal information is ignored. The results show that size of result set is considerably confined using semantic clauses and also by making use of temporal information. The limitation of CROQS is that it eventually searches on the basis of text and that's why returns some logically inaccurate results.

With the accelerated use of internet, search engines are becoming a part of our daily lives. Consequently, demand for more and more intelligent and accurate search engines is rising. Traditional search engines have several limitations in fulfilling user's needs e.g. Keyword based search engines return many irrelevant results. Further the users may be interested in knowing about some particular concepts, instances and type of relationships between any two instances or concepts. To address these issues *Kunmei Wen* designed and implemented ontology based search engine named Smartch [9]. *Kunmei Wen* proposed a novel method for ranking results while searching concepts, instances and type of relationships between them. Results show that an appropriate ranking method plays a vital role in confining the result set size, particularly in case of association relationship. As due to a larger num of association relationships, there are more chances of overwhelming results in absence of appropriate ranking method.

Using Smartch, the users can query in four different ways: (1) query for a keyword that shows up in a web page, (2) query for all instances that belong to a concept, (3) query for relationship between any two entities, and (4) query for all instances of a user defined concept. Smartch provides a graphical user interface, where the users can select a concept and expand/restrict its properties to define a new concept for the query purpose. It translates user queries in a uniform format using its query processor. Then this query is distributed over two different engines. One part of the query is sent to a traditional search engine. This traditional engine uses an index, made on web pages, to find results. Other part of the query is sent to an inference engine. Results returned by traditional search engine are also forwarded to inference engine. The inference engine then performs reasoning using domain ontology to extract the semantic information. After that, the ranking engine ranks the results returned by the inference engine. Finally, results are filter by a filtering mechanism and returned to the user.

As data on web is increasing exponentially, it is getting more and more difficult to search over the internet. It is essential to have an easy way to search over the web. For this purpose, Boumechaal, Hasna [10] implemented a tool which makes use of ontology to convert user's natural language query into an nRQL query. The resultant query is then sent to the inference engine named RACER, to query the knowledge base. There are two major complications in implementing this system. First, it is difficult for a machine to understand ambiguous natural language queries. Secondly, it is a challenging task to convert the natural language query into a correct and valid nRQL query. A dictionary is used to address the first problem i.e. understanding the ambiguous natural language terms. If a term is not found in ontology, its synonyms are retrieved from the dictionary and searched in ontology. The result of this search is a sequence of entities from the ontology. The returned sequence of entities is represented in the form of triples to address the second problem i.e. conversion of the natural language query into a correct and valid nRQL query. In the end, an algorithm is used to generate nRQL queries on the basis of these triples. The paper claims 85% accuracy while testing this tool on an animal ontology. One limitation of the system is that only one ontology is used in the process of query conversion which is a shortcoming of using this tool.

In modern era, heterogeneity of data sources is a major obstacle in information retrieval. Transparent integration of multiple heterogeneous data sources is a necessity of future. *Jinpeng*

Wang [11] proposed an ontology based technique to integrate heterogeneous data sources. In this approach, an RDF ontology is used as a mediated schema to integrate XML database and relational database. Here, the ontology serves as a shared vocabulary to hold semantics and the data sources work independently. The proposed architecture contains four basic parts: (1) a query processor to parse, translate and rewrite user's SPARQL queries into source specific queries, (2) an RDF ontology as mediated schema, (3) a source description that contains ontology to data source mappings, and (4) a wrapper that provides interface to query the data source. This system translates the SPARQL queries into source specific queries which can be executed by the respective query processors. The model only supports XML and relational database schemas which is a major limitation of this model.

The field of medicine has a great potential to benefit from advancements in computer sciences. Clinical databases contain heterogeneous and complex data which makes it difficult to retrieve information. Clinical researchers are usually not well aware of underlying technologies and hence face problems while querying clinical databases. Guo-Qiang Zhang designed and implemented a query interface, named VISAGE [12], to address this issue. VISAGE assists clinical researchers in data exploration by giving hints about cohort identification and potential hypothesis. The author proposed a novel data access pattern that enables clinical researchers and database managers to work in a collaborative manner. In the proposed pattern, delays are reduced by omitting data requests to database managers, instead clinical researchers directly interact with the database. To make VISAGE a useable tool, a user-centric approach is adopted and end users are engaged at every stage of development. Several types of analysis including need analysis, user analysis, task analysis and functional analysis are carried out to ensure effective and efficient use of VISAGE.

VISAGE has three main components: (1) Query Builder, (2) Query Manager and (3) Query Explorer. Query builder is responsible for visual support and ontology driven vocabulary support. Visual support is provided via components like search bar, radio buttons, check boxes and slide bar. Ontology driven vocabulary support is provided via domain ontology consisting of a list of searchable terms as a starting point. Query manager stores and labels the queries to facilitate sharing and reuse of queries. Query explorer provides a novel feature of on-line data mining. To provide on-line data mining in its true spirit, users are enabled to quickly specify the selection criteria and obtain results from selected data sources. Query Explorer displays results using histograms and pie charts for continuous and categorical data respectively. Clinical researchers can analyze multiple query results to mine knowledge or compare results. On-line data mining is less powerful than off-line approaches but off-line approaches may take months to complete the analysis process. So, VISAGE is not designed to replace other data mining approaches but to enable users to get hints before a complete offline analysis.

The comparisons of ontology based query formulation tools/techniques presented in various research articles and summarized in [13] have been further explored and presented here in table 2.1.

Transactional data transformation required

A few tools/techniques oblige transactional data to be transformed in the ontology in order to support the query formulation while few others need just schema information to be transformed in the ontology.

Support for multimedia database

Multimedia databases are specially designed to offer support for data in multimedia format. A few tools/techniques support the query formulation for the multimedia media databases.

Support for different levels of granularity

A few tools/techniques support the formation of generalized and specialized queries so that most suitable level can be chosen.

Support for multiple data sources of heterogeneous nature

A few tools/techniques support the formation of query over multiple data sources of heterogeneous nature. In such tools/techniques schema information of all the sources is stored in ontology that acts as a wrapper.

Support for queries in natural language

A few tools/techniques support natural language queries.

	Transactional data transformation required	Support for multimedia database	Support for different levels of granularity	Support for multiple data sources of heterogeneous nature	Support for queries in natural language	Support for queries using semantic clauses	Support for text based queries	Query by Combination of both (text and semantic clause)
TAMBIS [14]	Yes	No	Yes	No	No	Yes	Yes	Yes
GRQL [15]	Yes	No	No	No	No	Yes	Yes	Yes
SEWASIE [16]	No	No	No	Yes	No	Yes	Yes	Yes
SHOE [17]	Yes	No	No	No	No	Yes	Yes	Yes
Ontogator [18]	Yes	Yes	Yes	No	No	Yes	Yes	Yes
OntoViews [19]	Yes	Yes	Yes	No	No	Yes	Yes	Yes
CROEQS [8]	No	Yes	No	No	No	Yes	Yes	Yes
Smartch [9]	Yes	No	No	No	No	Yes	Yes	No
Boumechaal, Hasna 2011 [10]	Yes	No	No	No	Yes	No	Yes	No
Wang, Jinpeng 2009 [11]	No	No	No	Yes	No	Yes	No	No

Table 2.1: Comparison table of ontology based query formulation tools/techniques

<u>Support for queries using semantic clauses</u>

A few tools/techniques support semantic queries using classes and properties e.g query formulated by applying a constraint on values of a property.

<u>Support for text based queries</u>

A few tools/techniques offer support for keyword based queries.

2.3 Tools and Approaches for Database to Ontology Transformation

Semantic web is considered the future of current web structure so the existing web must be enriched with ontologies to build semantic web. Web content is currently based on multiple heterogeneous data sources. Three-fourths of these data sources are based on relational databases. So, relational databases are the most critical target to be transformed into corresponding ontologies. Researchers have proposed different mapping rules to extract ontologies from relational databases. A major hurdle in implementation of many transformation techniques is the requirement of highly normalized databases. Another hurdle is the inefficiency of extraction process as extraordinary number of join operations is performed to extract complete ontology.

Choi, Ji Woong [20] proposed a novel set of transformation rules to extract ontologies from the relational databases. In the proposed approach, database is required in first normal form only with primary key in each relation. Furthermore, unlike many of the previously proposed techniques, the extracted ontologies can be extended by reasoning. Ontology is extracted from relational database in three phases. In the first phase, TBox is written using relational schema. A unique feature of this phase is that the foreign key is not simply used as a pointer to the referenced table but classes are generated from both referenced and referencing columns. The class related to the referencing column is defined as the subclass of the one related to the referenced column. In the second phase, ABox is written using database instances without performing extraordinary number of join operations. The frequency of join operations is reduced because of distinctive handling of foreign keys. In the third and last phase, the ontology is extended by inferring associations.

The use of ontologies in different domains is continuously increasing. Though ontologies can be constructed manually but it is a very exhaustive task. As a step towards automated ontology construction, *Lu Yiqin* [21] proposed a set of mapping rules to formulate an ontology from a relational schema. In the transformation process, the paper claims to consider the relational schema and also the concept module between relational schema and real world. Proposed mapping rules are the key in conversion of database schema into ontology. The major rules include: (1) form a class in ontology for equivalent table in relational schema, (2) form single class for multiple tables describing the same instance, (3) map foreign key column to Object property, (4) map non-foreign key column to Datatype property, (4) map unique constraint to InverseFunctionalProperty, (5) map NULL constraint to the minCardinality constraint with value of 1 (6) primary key constraint is mapped to both InverseFunctionalProperty setting minCardinalty as 1. The author explains these mapping rules with the help of use cases. A comparison of proposed technique to some existing techniques is also presented.

The comparisons of database to ontology transformation tools/techniques presented in various research articles and summarized in [13] have been further explored and presented here in table 2.2.

Level of automation in transformation

It is the level of human intervention needed in relational schema to ontology transformation process.

Ontology extendable by reasoning

A few tools/techniques support extension of ontology by applying reasoning on the transformed ontology.

Support for dynamic web

A few tools/techniques offer support for dynamic web by providing support for relational databases as most of the web content is in relational format.

Preservation of structural constraints

A few tools/techniques preserve the structural constraints in the transformation process e.g. referential constraints.

Multiple levels of granularity in the generated ontology

A few tools/techniques express data at different levels of granularity for effective search and integration.

Output ontology format

Different ontology transformation tools/techniques construct ontologies in different formats.

Selective conversion

A subset of database can be transformed into ontology.

	Type	Level of automation in transformation	Ontology extendable by reasoning	Support for dynamic web	Preservation of structural constraints	Multiple levels of granularity in the generated ontology	Output ontology format	Selective conversion
RDB2Onto [22]	Tool	Both (Fully & Semi automated)	No	Yes	No	No	RDF/OWL	Yes
RDB2ONT [23]	Tool	Automated	No	Yes	Yes	Yes	OWL	No
QUALEGDB [24]	Tool	Automated	No	Yes	Yes	No	OWL	No
Yiqing, Lu 2012 [21]	Algorithm	Automated	No	Yes	Yes	No	OWL	No
Choi, Ji Woong 2012 [20]	Algorithm	Automated	Yes	Yes	Yes	No	OWL	No
DB2OWL [25]	Tool	Automated	No	Yes	Yes	No	OWL	No
Man Li 2005 [26]	Algorithm	Both (Fully & Semi automated)	Yes	Yes	Yes	No	OWL	No

Table 2.2: Comparison table of database to ontology transformation tools/techniques

2.4 Tools and Approaches for Database to Ontology Mapping

Ontology can be created from relational database using a database to ontology transformation technique. But if both relational database and ontology are pre-existent then database to ontology mapping could be needed. Two major motivations for database to ontology mapping are: (1) transformation of database instances into ontology instances, and (2) creation of an intermediate schema, to serve as wrapper, while integrating multiple heterogeneous data sources. Manual database to ontology mapping is possible, but is an exhaustive and error prone task. In the last decade, many automatic mapping techniques are proposed by researchers. Sequeda, Juan F [27] recently proposed an automatic database to ontology mapping technique. The author claims to assure two basic desired properties of semantic mapping; (1) information preservation, and (2) query preservation. Information preservation refers to the ability of reconstructing the original database using the database to ontology mappings. And the terms query preservation refers to the ability of translating every relational query into corresponding semantic query.

Two other desirable properties of semantic mapping are monotonicity and semantic preservation. A semantic mapping is monotonic if entire mapping needs not to be recomputed after some updates in the database. Semantic preservation refers to the preservation of integrity constraints in semantic mapping. The author argues that a monotone mapping can't be semantic preserving at the same time. In the proposed approach, mapping is generated using relational schema and sets of primary and foreign keys. Mapping is defined as an OWL ontology which can be used to extract RDF instances from relational database. In mapping ontology, object properties and datatype properties are used to model foreign keys and relational attributes respectively. Three types of triples are generated while translating the relational instances into RDF instances; (1) table triples are created for all the tuples in the database to refer the tables they belong, (2) reference triples are created to hold the information about all the referential relationships, and (3) Literal triples are created for all the attribute of every tuple to hold their literal values.

The comparisons of database to ontology mapping tools/techniques presented in various research articles and summarized in [13] have been further explored and presented here in table 2.3.

<u>Fully declarative</u>

This term refers to the expressive power of a mapping language.

<u>Level of automation in mapping</u>

It refers to the level of human intervention needed in mapping process.

<u>Support for dynamic web</u>

Relational databases are used on the back end of dynamic web. So support for dynamic web can be provided by handling relational databases.

<u>Format of defined mappings</u>

Mapping tools and techniques provide mapping definitions in different formats.

<u>Goal is to convert DB instances into ontology instances</u>

This term states that either transformation of relational data instances into ontological data instances is the final goal or not.

<u>Use of data values in relations</u>

This term states that either data values in database relations are used in mapping process or not.

	Type	Fully declarative	Level of automation in mapping	Support for dynamic web	Format of defined mappings	Goal is to convert DB instances into ontology instances	Use of data values in relations
D2R-MAP [28]	Language	No	N/A	Yes	XML	Yes	Yes
Extended D2R [29]	Language	No	N/A	Yes	XML	Yes	Yes
R_2O [30]	Language	Yes	N/A	Yes	XML	Yes	Yes
VisAVis [31]	Tool	N/A	Semi automated	Yes	Ontology	No	Yes
D2OMapper [32]	Tool	N/A	Both (Fully & Semi automated)	Yes	XML	Yes	No
SFSU ER Design tools [33]	Tool	N/A	Automated	Yes	Ontology	No	No
Sequeda, Juan F, 2012 [27]	Algorithm	N/A	Automated	Yes	Ontology	Yes	No

Table 2.3: Comparison table of database to ontology mapping tools/techniques

2.5 Evolution of Ontology Languages

Many ontology languages were developed in the recent past. Most of the developed languages were XML based e.g. XOL[34]. XML allows users to add meaningful tags with the data elements but these tags are not meaningful for the machine. XML was extended to develop Resource Description Framework (RDF) for semantic web. The RDF schema (RDFS) elements enable users to add semantics in an ontology using subjects, predicates and objects[7]. RDFS is easy to use but it has limited expressive power to define real world concepts. A more expressive ontology language, Web Ontology Language (OWL), was recommended by W3C in 2004 [2]. OWL has 3 sublanguages; OWL-Full, OWL-DL, and OWL-Lite. Each of these sublanguages contains a subset of OWL constructs. OWL-Full contains all of OWL constructs and it is not fully decidable. Whereas OWL-DL contains largest decidable subset of OWL and OWL-Lite is even more restrictive in terms of allowed constructs. Selection between OWL-full and OWL-DL depends upon the required meta-modeling facilities and selection between OWL-DL and OWL-Lite depends upon the required expressive power.

OWL 2 was recommended by W3C in 2009 [3] which is an improved and extended version of OWL 1. OWL 2 provides new and more expressive constructs including qualified cardinality restrictions, disjoint properties and property chains. OWL2 has 3 new profiles; OWL 2-EL, OWL 2-QL, and OWL 2-RL. Each of these contains a subset of OWL-2 constructs. Selection among these profiles depends upon the required ontology structure and nature of the reasoning task. OWL 2-EL is appropriate for the scenarios where plenty of classes and properties are to be modeled. OWL 2-QL is appropriate for the scenarios where heaps of individuals are to be added. OWL 2-RL is appropriate for the scenarios where scalable reasoning is needed without losing much expressive power. All these new profiles of OWL 2 are more restrictive as compared to OWL1-DL. Some restrictions of OWL 1-DL are relaxed in OWL 2-DL making it most expressive yet decidable subset of OWL2 constructs. A comparison between RDF and OWL 1 was presented in [13] which is extended for OWL2 and presented in Table 2.4.

Concepts	RDF(s)	OWL 1	OWL 2
Data Types	✓	✓	✓
Extensions	✓	✓	✓
Bounded Lists	✓	✓	✓
Formal Semantics	✓	✓	✓
Reification	✓	✓	✓
Inheritance	✓	✓	✓
Equivalence	×	✓	✓
Class Definitions	×	✓	✓
Constraints	×	✓	✓
Enumerations	×	✓	✓
Quantification Restrictions	×	✓	✓
Propositional Connectives	×	✓	✓
Literal Value Restrictions	×	✓	✓
Cardinality Constraints	×	✓	✓
Inference	×	✓	✓
Self Restriction	×	×	✓
Disjoint Properties	×	×	✓
Property Chains	×	×	✓
Keys	×	×	✓
Qualified Cardinality Restrictions	×	×	✓
Rich Data Types	×	×	✓

Table 2.4: A comparison between RDF(s), OWL 1 and OWL 2 [13]

This research work primarily aims to exploit OWL 2-DL's expressive power to generate complex relational queries. OWL 2 allows definition of sophisticated restrictions which can be translated into relational query restrictions. Key novel features of OWL 2 (as presented in table 2.4) include; self restriction, disjoint properties, property chains, keys, qualified cardinality restrictions and rich data types. Self restriction allows defining a class of individuals which are linked to themselves by a given object property. This is somewhat similar to the self join scenarios in relational domain but is not exactly the same. In case of self join, one row of a table is joined with another row of the same table but not with itself. On the other hand, in case of self restriction, every individual is to be connected with itself. This is how self restriction is not a suitable candidate to be translated into equivalent relational query restriction.

OWL 2 allows to state that the given object properties are disjoint. One individual cannot be connected to another individual by 2 disjoint properties simultaneously. This feature can be exploited to automatically detect illogical queries, which is a natural future work of this research. OWL 2 also allows defining object properties as compound of other object properties. This feature can be exploited to define properties more efficiently in the domain ontology. There was no mechanism available in OWL 1 to state that the individuals of a given class are uniquely identified by values of a given property. OWL 2 provides this feature by allowing definition of a set of (object or datatype) properties as key properties. This feature is of great utility for the representation of transactional data in the ontology. In OntoQF [1], transactional data is not translated in the ontology so this feature cannot add much value to this research work.

Cardinality restrictions allow users to define restriction on the cardinality of a given property. OWL 1 allows defining class of individuals who have at least n connections (with other individuals or literals) by a given property. OWL 2 allows defining more complex cardinality restrictions by also applying limit on the range of the property. These sophisticated restrictions can be exploited to formulate relational queries with advanced constructs of GROUP BY and HAVING. In this thesis, some extensions are proposed to translate cardinality restrictions and qualified cardinality restrictions into equivalent relational query restrictions. Furthermore, only allowed data types in OWL 1 are integers and strings. Users are not allowed to define their own data ranges by applying restrictions on basic data types. On the other hand, OWL 2 provides support for rich data types (decimal, float, double, etc.). Users can also define their own data ranges by applying sophisticated restrictions on basic data types. Meaningful quantification restrictions on datatype properties can be defined in the presence of these rich data ranges. Moreover, meaningful qualified cardinality restrictions can also be defined using data type properties. These types of restrictions can be of vital importance while querying on relational databases. So, some extensions are also proposed to extend the translation support for these restrictions.

2.6 Chapter Summary and Conclusion

It has been concluded in this chapter that some mechanism should be provided to assist users in query formulation, especially to those who don't have any knowledge regarding underlying information structure. Ontologies are suitable for providing assistance in query formulation. Ontologies can play a vital role where users need to query on heterogeneous data sources or need to apply semantic query, without converting all the data in ontology format. Further it has also been observed that relational databases can be transformed into or mapped onto an ontological database. Finally, it has been concluded that no existing system provides assistance in complex query formulation including GROUP BY and HAVING clauses, without requiring the user to have knowledge of underlying information structure.

It has also been observed that OWL DL is the most expressive but decidable profile of OWL. OWL2 DL is even more expressive as OWL2 is an extended version of OWL. Sophisticated logical constraints defined using constructs of OWL2, can be mapped onto relational query constructs. So, expressive power of OWL2 DL should be exploited to assist users in generating complex relational queries. To achieve this goal, some extensions to the existing architectural framework, named OntoQF, are proposed in the next chapter (chapter 3).

Chapter 3: Ontology Driven SQL GROUP BY and HAVING Statements Formulation using OWL2 - Extension to OntoQF [1]

3.1 Introduction

After a detailed literature review in the domain of ontology based query formulation, it has been concluded that none of the existing approaches, to the best of our knowledge, support relational query formulation with ROUP BY and HAVING clauses. OntoQF [1] is an ontology base query formulation approach which requires only the metadata to be transformed in the domain ontology. Moreover, OntoQF assists query formulation without requiring the understanding of the underlying information structure. OntoQF is based on OWL1 and provides support for SELECT, FROM and WHERE clauses of the relational query. Hence, in this research, an extension to OntoQF is proposed by adding support for GROUP BY and HAVING relational constructs. To achieve this objective, there is a need to identify the extent to which these relational constructs can enhance the usefulness of OntoQF. This requirement will lead us to answer the research question number 1 i.e.

"To what extent support of GROUP BY and HAVING relational query constructs can increase the usefulness of OntoQF?"

Furthermore, it has been discussed in the literature review that a more expressive ontology language OWL2 has been recommended by W3C [3] while OntoQF [1] is based on OWL1. Sophisticated concept restrictions of OWL2-DL can be translated into equivalent sophisticated relational queries. However, it may not be mandatory or even doable to translate all OWL2-DL concept restrictions into relational queries. Rich data types in OWL2-DL may also play a vital role in formulating useful queries. Hence, there is a need to identify OWL2-DL constructs that can be translated into equivalent relational query constructs in the query formulation process. This requirement will lead us to answer the research question number 2 i.e.

"To what extent support of selected new constructs of OWL 2-DL can increase the usefulness of OntoQF?"

It has also been concluded in the literature review that OWL2-DL concept restrictions can be of different complexity so the query formulation is dependent upon: (1) structure of underlying relational database schema, and (2) combinations and complexity of OWL2-DL concept restrictions. OntoQF [1] translation algorithms need to be extended to translate the OWL2-DL constructs into equivalent relational query constructs. Furthermore, relational database to ontology mapping is also needed to be extended to transform generated query into executable relational query with GROUP BY and HAVING constructs. This requirement will lead us to answer the research question number 3 i.e.

"What mappings are required between GROUP BY and HAVING relational constructs and selected OWL2-DL constructs for query translation?"

On the basis of above discussion, some extensions to the query formulation algorithms of OntoQF [1] are proposed in order to provide support for GROUP BY and HAVING clauses using OWL2-DL. This extension requires modeling of domain ontology using OWL2-DL. Domain experts can define domain knowledge using sophisticated ontology statements which can be translated into equivalent sophisticated relational queries.

In this chapter, section 3.2 briefly introduces the ontology driven extended relational query formulation approach. Details regarding extended knowledge representation in ontology using sophisticated constructs of OWL2-DL are presented in section 3.3. Section 3.4 discusses the extended translation algorithms to translate sophisticated semantic clauses into corresponding relational clauses including GROUP by and HAVING. Section 3.5 discusses the extended mapping process which is required to build the executable query from the results of translation process. In the end, chapter summary and conclusions are presented in section 3.6.

3.2 Ontology Driven SQL GROUP BY and HAVING Statements Formulation Approach

OntoQF [1] consists of two phases; (1) pre-processing, and (2) translation (as shown in Figure 3.1). Both of these phases are extended to provide support for GROUP BY and HAVING clauses using OWL2-DL. The pre-processing phase further consists of two sub-phases; (1) database to ontology transformation, and (2) database to ontology mapping. Database to ontology transformation uses schema information to formulate ontology and then domain concepts are added to it, to make it domain ontology. Database to ontology transformation only translates domain metadata (part of data that contains semantic information of a domain) and information regarding relationships into ontology. On the other hand, transaction tables (tables created to represent many to many relationship between the entities) and general schema restrictions (e.g. unique, not null, value constraints etc.) are not translated into ontology. This is because; OntoQF only deals with SELECT clause of relational query and any information to maintain database consistency during the operations of insert and update is not required. In this research, formation of domain ontology is extended by adding support for OWL2-DL constructs which allows the domain experts to define more sophisticated concept restrictions. Database to ontology mapping is also extended in order to provide support for GROUP BY and HAVING constructs and aggregate functions.

3.3 Extended OntoQF Domain Ontology Formulation

A generic ontology stores real world concepts and needs to be extended before it can be used in particular domains. Domain ontology is the ontology which is constructed for a specific domain e.g. lungs cancer, nanophysics etc. In most of the cases, construction of domain ontology is not the ultimate goal but domain ontology is built to be used for particular purposes. In this research work, domain ontology is extended to assist users in formulation of relational queries with advanced constructs of GROUP BY and HAVING. There is no generic globally accepted way of constructing domain ontology. Requirements for query formulation are kept in mind while defining mechanism for construction of domain ontology [1]. OWL2-DL is selected for this research work as it is the largest decidable subset of OWL2. A DL knowledge base (KB) [35] has two parts: (1) ABox is the assertion part which describes the domain structure just like schema in a database, and (2) TBox is the terminology part which describes concrete situations. Domain ontology formulation process is depicted in figure 3.2.

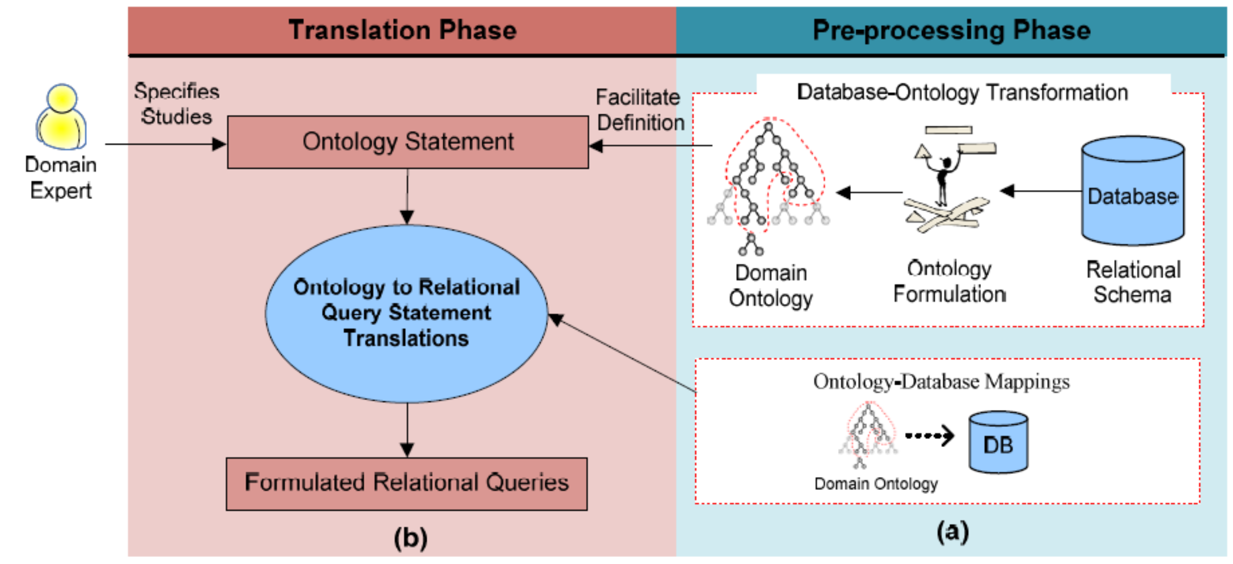


Figure 3.1: The Pre-processing and Translation Phases of Ontology-Driven Query Formulation Approach [13]

Relational database consists of tables (relations), attributes and relationships between the tables. Attributes are used to store data and references (foreign keys). On the other hand, domain ontology consists of classes, datatype/object properties and assertions. Datatype properties are used for linking class individuals to literal values and object properties are used for linking class individuals to other class individuals. In order to formulate domain ontology, which can be used in the process of query formulation, relational schema information needs to be transformed into ontology. Tables are required to be in third normal form (3NF) in order to be transformed into ontology. Tables that contain semantics or domain metadata are translated into ontology classes. Columns that store literal values and do not contain any metadata, are transformed into datatype properties. And columns that store metadata i.e. reference information, are transformed into object properties. Tables that are created to represent a many to many relationship are called transaction/bridge tables. Transaction tables are not transformed while building domain ontology as OntoQF does not use transactional data in the process of query formulation. Moreover, general restrictions defined in relational schema (e.g. unique, not null and value restrictions) are also not transformed into domain ontology. These general restrictions are applied to restrict data entry (insert or update) and OntoQF deals with SQL SELECT queries only [1].

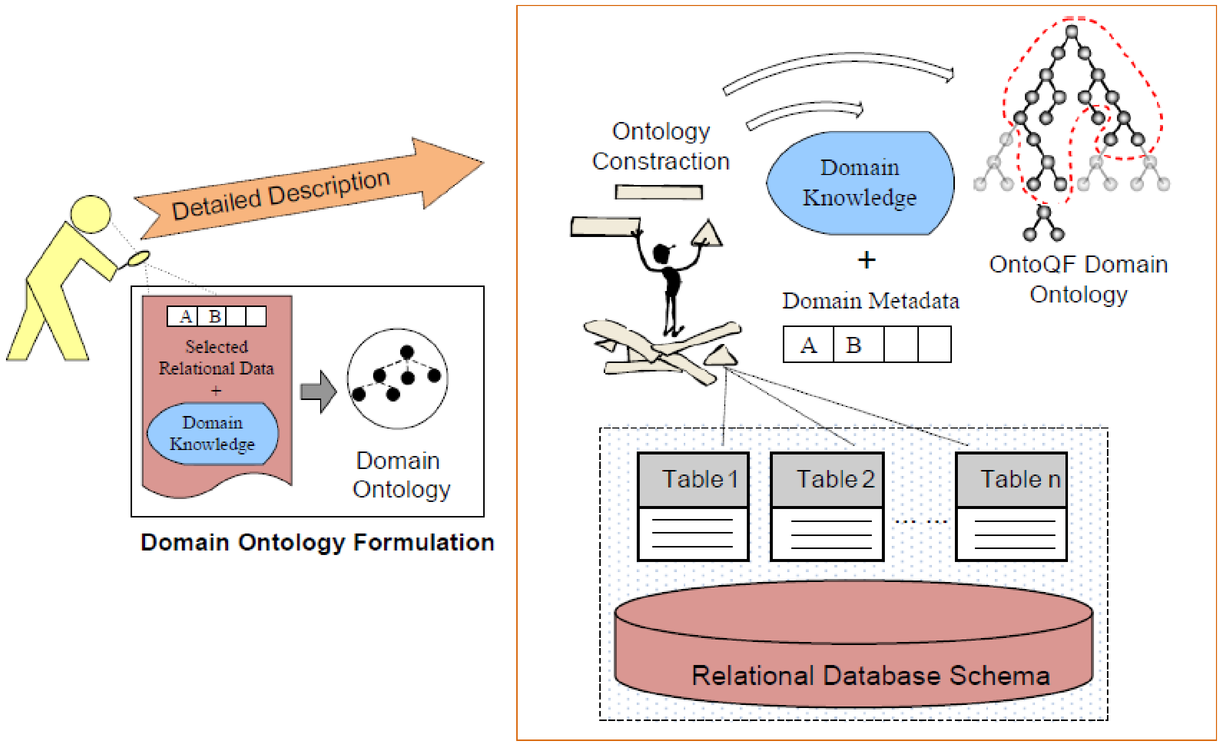


Figure 3.2: Domain Ontology Formulation for Query Formulation [13]

3.3.1 Domain Knowledge Representation in the Extended OntoQF Ontology

Once the structural part of the ontology has been completed, domain knowledge can be added in it to make it domain ontology. In OWL2-DL [3], the domain knowledge can be expressed in terms of property assertions and concept restrictions for individuals and classes (concepts) respectively. Individuals are named objects from the domain of interest and classes are sets of individuals that satisfy the conditions specified in the class expression. Classes are the simplest examples of class expressions. Complex class expressions are constructed using classes and property expressions. Property expressions relate individuals with other individuals or relate individuals with literal values. There are two types of properties in OWL2-DL which can be used to construct property expressions; (1) Object Properties, and (2) Datatype properties. Object properties are binary relationships between pairs of individuals, such as father-of, works-at etc. And datatype properties link individuals to literals, Such as has-age, has-weight etc.

In OWL2-DL, the concept restrictions, used to express domain knowledge can be of arbitrary complexity ranging from simple to complex concept restrictions. There are four major types of constructs (as shown in figure 3.3) that can be used to formulate a class expression; (1) Propositional Connectives, (2) Cardinality Restrictions, (3) Qualified Cardinality Restrictions, and (4) Quantification Restrictions. All these constructs support object properties as well as datatype properties to formulate class expressions. Propositional Connectives can be further categorized into 3 types; (1) Intersection of Class Expressions, (2) Union of Class Expressions, and (3) Complement of Class Expressions. Intersection (∩) of class expressions is used to define the class of all individuals that are instances of all the specified classes. Union (∪) of class expressions is used to define the class of all individuals that are instances of at least one among the specified classes. And complement (¬) of class expressions is used to define the class of all individuals that are not instances of the specified class.

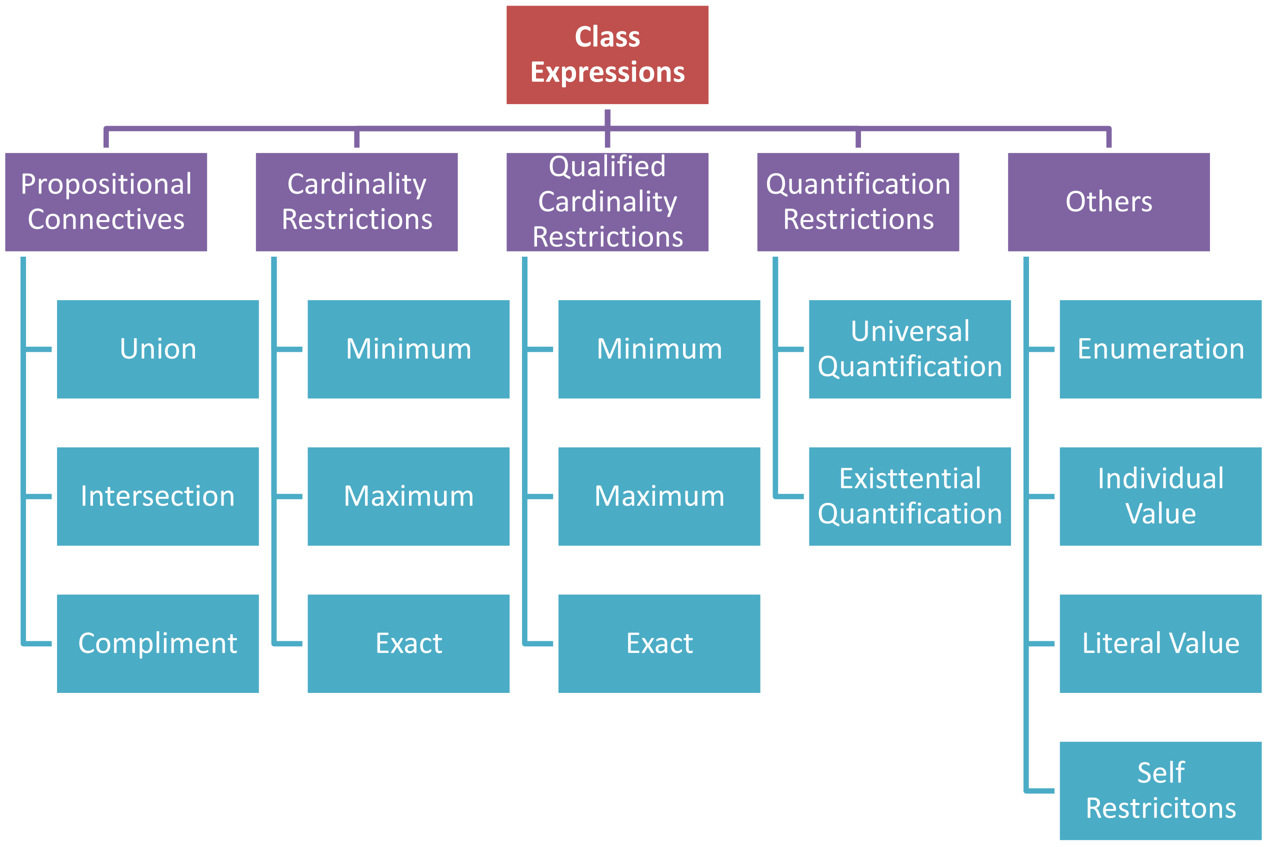


Figure 3.3: Class Expressions in OWL2-DL

Any individual may have zero or many values for a particular object property/datatype property. Cardinality restrictions are used to apply restriction on number of values that any individual may have for a particular property. Cardinality restrictions can be further classified into three categories; (1) Minimum Cardinality, (2) Maximum Cardinality, and (3) Exact Cardinality. Minimum cardinality restrictions are used to define the class of all individuals that have at-least given number of values for a particular object property/datatype property. For example, class of all the individuals that have at least one phone number or class of all the individuals that have at least one child. Similarly, maximum or exact cardinality restrictions are used to define the class of all individuals that have at-max or exactly given number of values for a particular property respectively. Cardinality restrictions can also be qualified. In the case of qualified cardinality restriction, the range of the property is also restricted. Qualified cardinality restrictions are also used to apply restriction on the number of values that any individual may have for a particular property. In addition, the values should be in qualifying data range or should be instances of qualifying class for datatype properties or object properties respectively. For example, class of all the individuals that have at least one phone number of Warid Telecom or class of all the individuals that have exactly one son. Qualified cardinality restrictions can also be further categorized into three different categories; (1) Minimum Qualified Cardinality, (2) Maximum Qualified Cardinality, and (3) Exact Qualified Cardinality.

In OWL2-DL, class expressions can also be formed using quantification restrictions. There are two main types of quantification restrictions; (1) Existential Quantification, and (2) Universal Quantification. Universal Quantification ($\forall$) is used to define the class of all individuals that are only connected to individuals of a specific class or literals of a specific data range by a given object property or datatype property respectively. Existential Quantification ($\exists$) is used to define the class of all individuals that are connected to individuals of a specific class or literals of a specific data range by a given object property or datatype property respectively. In addition to these constructs, literal value restrictions can also be applied for datatype properties in order to formulate class expressions. For example, class of all the individuals that have 70 kg weight.

In original OntoQF [1], SQL query formulation support is available for; (1) Propositional Connectives, (2) Quantification restrictions for object properties, and (3) Literal value restrictions for datatype properties. This research work extends the support for; (1) Quantification restrictions for datatype properties, and (2) Cardinality restrictions for both datatype properties and object properties, and (3) Qualified cardinality restrictions for both datatype properties and object properties (A list of class expressions supported in original OntoQF and extended OntoQF is presented in table 3.1). These OWL2-DL restrictions can be translated into equivalent relational queries to retrieve the data from relational database that satisfies the given restriction. In addition to rich class expressions, another key novelty of OWL2-DL is the capability of expressing rich data ranges. In OWL1, only supported data ranges were strings and integers. OWL1 does not allow defining new data ranges by applying restriction on existing data ranges. For example, using OWL1, it is not possible to define a data range containing integer values greater or equal to hundred. In OWL2, rich data ranges can be defined by applying restrictions on existing data types [36]. In the proposed extension to OntoQF, this capability is exploited to provide support for; (1) Quantification restrictions for datatype properties, (2) Cardinality and qualified cardinality restrictions for datatype properties.

Class Expressions	Supported in Original OntoQF	Supported in Extended OntoQF
Intersection Of Class Expressions	✓	-
Union Of Class Expressions	✓	-
Complement Of Class Expressions	✓	-
Existential Quantification With Object Properties	✓	-
Universal Quantification With Object Properties	✓	-
Literal Value/Comparative Restrictions With Datatype Properties	✓	-
Cardinality Restrictions With Object Properties	×	✓
Qualified Cardinality Restrictions With Object Properties	×	✓
Cardinality Restrictions With Datatype Properties	×	✓
Qualified Cardinality Restrictions With Datatype Properties	×	✓
Existential Quantification With Datatype Properties	×	✓
Universal Quantification With Datatype Properties	×	✓

Table 3.1: Class Expressions supported in Original OntoQF and Extended OntoQF

3.4 Translation of Sophisticated Ontology Statements into Equivalent Relational Query Expressions Including GROUP BY and HAVING Clauses

Translation of ontological constructs into equivalent relational query constructs is a major part of OntoQF [1]. This research is concerned with providing translation support for more expressive and sophisticated OWL2-DL constructs. Translation support for qualified cardinality restrictions and rich data ranges is provided in order to enable formation of sophisticated relational queries. Furthermore, this research work is concerned with providing support for GROUP BY and HAVING relational query constructs in the query formulation process. Extended translation algorithms are proposed which result in the formation of WHERE and HAVING clauses of relational query. The join conditions, SELECT, FROM and GROUP BY clauses are formulated by a separate extended mapping algorithm. All the semantic constructs are individually translated into equivalent relational query constructs and their results are combined to formulate the final query. Correctness of one relational query clause can be affected by content of other relational query clauses. Therefore, a decent mapping algorithm is required to formulate correct executable relational queries.

3.5 Extended Ontology to Database Mapping

Once the semantic restrictions are individually translated into equivalent relational query restrictions, further processing is required to formulate executable relational queries. This required processing includes; (1) mapping of the semantic properties onto the corresponding relational database structure, (2) integration of all the results produced by translation algorithms in order to correctly formulate SELECT, FROM and GROUP BY clauses in addition to the join conditions. To fulfill these requirements, ontology vocabulary needs to be mapped onto relational database vocabulary and mapping results needs to be stored for later use [1]. The mapping process requires in depth domain knowledge, hence mappings are identified manually by domain experts. These mappings include details about datatype properties, object properties, aggregate functions, column names, table names, primary keys, foreign keys and database name. Ontology to database mapping is stored in the database table named mappings as shown in table 3.2. The mapping process is not complicated for the queries that access one or more attributes from a single table. However, mapping process can be quite complicated in cases; (1) query involves different restrictions on multiple columns, (2) query involves multiple tables and hence multiple join operations are required, and (3) query involves aggregate functions. In this regard, an extended database to ontology mapping algorithm is proposed and implemented in extended OntoQF (Discussed in Chapter 4).

Table Name: mappings

Column Name	Constraint
Property_ID (PK)	Unique/ Not Null
PropertyName	Unique/ Not Null
PropertyType	Not Null
PropertyTable	Not Null
PropertyColumn	Not Null
OtherTables	Not Null
AggregateFunction	Not Null
JoinConditions	Not Null

Table 3.2: Table to store mappings in Extended OntoQF

3.6 Chapter Summary and Conclusion

In an attempt to answer the research questions, it has been discussed and concluded that the support for GROUP BY and HAVING clauses using OWL2-DL can increase the usefulness of OntoQF. This research proposes major changes in three key areas; (1) Domain knowledge representation in OntoQF domain ontology, (2) Translation of sophisticated ontology statements into equivalent relational query expressions including GROUP BY and HAVING, and (3) Ontology to database mappings to build executable relational queries. Sophisticated OWL2-DL class expressions are used to store domain knowledge in extended OntoQF domain ontology. Domain ontology stores information about relationships among the tables and domain metadata from the relational database. This information is used in the process of ontology driven relational query formulation. The translation of sophisticated ontology clauses into corresponding relational query clauses enables to apply sophisticated restrictions on the records retrieved from the relational database. In the end, property restrictions are mapped to relational database structures including GROUP BY and HAVING in order to produce executable relational queries. In chapter 4, we present details of the extended translation algorithms and the extended mapping algorithm required for the ontology based relational query formulation.

Chapter 4: Ontology Driven SQL GROUP BY and HAVING Statements Formulation Process – Extension to OntoQF

4.1 Introduction

In the wake of having introduction to the proposed extension to OntoQF in chapter 3, this chapter concentrates on the details of the proposed extension for ontology driven complex relational query formulation. The translation of semantic clauses into equivalent relational query clauses was among the key features of original OntoQF. The key research contribution of this thesis is the proposed extension for the translation of sophisticated semantic clauses of OWL2-DL into corresponding complex relational query clauses. To achieve this extended translation functionality, an extension to the query generation algorithms is proposed.

The number and type of clauses in a relational query depends upon the nature of data that the user wants to retrieve. It can be very simple and may look like Q1: 'SELECT column2 FROM relation1'. It may look a bit more complex like Q2: 'SELECT relation1.column2 FROM relation1, relation2 WHERE relation1.column1= relation2.column1 and relation2.column2 = xyz'. It can also contain some aggregate functions and some complex conditions defined on aggregate functions like Q3: 'SELECT count (relation1.column2), relation2.column2 FROM relation1, relation2 WHERE relation1.column1= relation2.column1 and relation2.column3 = xyz GROUP BY relation2.column2 HAVING count (relation1.column2) > 123'. Here, the query Q1 contains only SELECT and FROM clauses. SQL SELECT clause is used to project the part of the relation that user wants to retrieve and FROM clause is used to list the tables involved. On the other hand, the query Q2 also contains WHERE clause which is used to mention join conditions (in case multiple tables involved) and to apply restrictions on data that user wants to retrieve. In the end, the query Q3 contains an aggregate function and GROUP BY and HAVING relational query clauses. The aggregate functions are applied on columns in the SELECT clause and restrictions on values returned by these functions are applied in HAVING clause. Aggregate functions are applied on group of values, formulated on basis of attributes mentioned in the GROUP BY clause.

In our proposed extension, WHERE and HAVING clause restrictions are generated by extended translation algorithms which individually translate semantic restrictions into complex relational query restrictions. Details of these translation algorithms are presented in detail in section 4.2. The SELECT, FROM and GROUP BY clauses in addition to the join conditions are generated by a separate mapping algorithm, presented in section 4.3. In the end, chapter summary and conclusions are presented in section 4.4.

4.2 Translating OWL2-DL Statements to Relational Query Expressions

The OntoQF domain ontology contains domain metadata and domain knowledge stored as sophisticated concept restrictions. These concept restrictions are flexible in terms of number of OWL2-DL constructs included [1]. In contrast, the SQL query may consist of SELECT, FROM, WHERE, GROUP BY and HAVING clauses which enable users to describe the data they want to retrieve. In this regard, OWL2-DL constructs are individually translated into equivalent relational query constructs in order to translate the sophisticated concept restrictions into complex relational queries. The syntax rules for OWL2-DL sophisticated constructs followed in this research are listed in table 4.1. In this research, translation algorithms are proposed for each; (1) Cardinality restrictions with object properties (P), (2) Qualified cardinality restrictions with object properties (P), (3) Cardinality restrictions with datatype properties (P_D), (4) Qualified cardinality restrictions with datatype properties (P_D), (5) Existential quantification restriction i.e. someValuesFrom (∃) with datatype properties (P_D), and (6) Universal quantification restriction i.e. allValuesFrom (∀) with datatype properties (P_D). The conventions adopted in above mentioned OWL2-DL to relational query translation algorithms are shown in table 4.2.

Construct syntax	Description
$C,D \longrightarrow A$	Atomic concept (class)
¬ C	Negation
C ∩ D	Intersection
C ∪ D	Union
□P . C	Universal quantification restriction with object property
∃P . C	Existential quantification restriction with object property
□P_D . DR	Universal quantification restriction with datatype property
∃P_D . DR	Existential quantification restriction with datatype property
P_D θ [value], where θ in	Literal value restriction with datatype property

{≤ , ≥ , =}	
θ [value] P, where θ in {≤ , ≥ , =}	Cardinality restrictions with object properties
θ [value] P . C, where θ in {≤ , ≥ , =}	Qualified cardinality restrictions with object properties
θ [value] P_D, where θ in {≤ , ≥ , =}	Cardinality restrictions with datatype properties
θ [value] P_D . DR, where θ in {≤ , ≥ , =}	Qualified cardinality restrictions with datatype properties

Table 4.1: Syntax rules for OWL2-DL constructs

Convention	Description
Q_F	It is a formulated SQL query for the current OWL2-DL construct
Q_f	It is a formulated WHERE clause condition for the current OWL2-DL construct
H	It is a formulated HAVING clause condition for the current OWL2-DL construct
C	It is an ontology concept (class)
C_1, C_2, … C_n	These are multiple ontology concepts (classes)
DR	It is an ontology data range
P	It is an object property which links class individuals with other class individuals
P_D	It is a datatype property which links class individuals with literals

Table 4.2: Conventions adopted in extended OntoQF

4.2.1 Translation of Cardinality Restrictions with Object Properties (P)

OWL2 allows the formation of class expression by applying restriction on the cardinality of an object property expression e.g. class of persons who have at most 2 children. ***ObjectMinCardinality***, ***ObjectMaxCardinality*** and ***ObjectExactCardinality*** are the class expressions that can be used to express the class of individuals which have at least, at most and exactly a specified number of connections respectively, to other individuals, by an object property expression [3]. The following class expression contains all the individuals that are connected by the object property, i.e. PrescribedDrugs, to at least 3 different individuals.

ObjectMinCardinality(3 a:PrescribedDrugs)

For cardinality restriction with object property, the corresponding relational query is required to retrieve values from relational database that satisfies "θ [value] P" condition, where θ in {≤ , ≥ , =}. The proposed mechanism to translate cardinality restriction with object property into SQL HAVING clause restriction is presented below as Algorithm 1.

Algorithm 1. Translation of Cardinality Restrictions with Object Properties (P)

```
h         ←   is formulated SQL HAVING clause condition for P
P         ←   in "θ [value] P", P is an object property, where θ in {≤ , ≥ , =}
[value]   ←   is a quantity value that is to be satisfied
if P then
    if P MinCardinality then
        h   ← P >= [value]
    else if  P MaxCardinality then
        h   ← P <= [value]
    else if P ExactCardinality then
        h   ← P = [value]
    end if
    return h
end if
```

4.2.2 Translation of Qualified Cardinality Restrictions with Object Properties (P)

OWL2 allows the formation of class expression by applying qualified restriction on the cardinality of an object property expression e.g. class of persons who have at least 2 children who are boys. ***ObjectMinCardinality***, ***ObjectMaxCardinality*** and ***ObjectExactCardinality*** are the class expressions that can be used to express the class of individuals which have at least, at most and exactly a specified number of connections respectively, to the instances of the qualifying class expression, by an object property expression [3]. The following class expression contains all the individuals that are connected by the object property, i.e. PrescribedDrugs, to at most 3 different individuals of type antibiotic.

ObjectMaxCardinality(3 a:PrescribedDrugs a:Antibiotic)

For qualified cardinality restriction with object property, the corresponding relational query is required to retrieve values from relational database that satisfies "θ [value] P. C" condition, where θ in {≤ , ≥ , =}. The proposed mechanism to translate qualified cardinality restriction with object property into SQL WHERE and HAVING clause restrictions is presented below as Algorithm 2.

Algorithm 2. Translation of Qualified Cardinality Restrictions with Object Properties (P)

```
$Q_f$      ← is a formulated SQL WHERE clause condition for qualified cardinality restriction
h          ← is formulated SQL HAVING clause condition for P
P          ← in "θ [value] P. C", P is an object property, where θ in {≤ , ≥ , =}
[value]    ← is a quantity value that is to be satisfied
if P then
    if P MinCardinality then
        h  ← P >= [value]
    else if  P MaxCardinality then
        h  ← P <= [value]
    else if P ExactCardinality then
        h  ← P = [value]
    end if
// START of the section taken from OntoQF [1]
        if ∃ C then
                {Check for one or more subclasses of class C}
                if ∃ c: p(c), where p is a subclass determination function that is p (c) is True
                iff c ∈ ($c_1$, $c_2$, . . . , $c_n$) then
                        if only one subclass of C is found then
                                $Q_f$ ← P ≡ c
```

```
                    else
                            {if more than one subclasses of C are found}
                            $Q_f \longleftarrow$ P IN ($c_1 \cup c_2 \cup . . . \cup c_n$)
                    end if
            else
                    {if class C does not have any subclasses}
                    $Q_f \longleftarrow$ P ≡ C
            end if
    end if
// END of the section taken from OntoQF
   return $Q_f$, h
end if
```

4.2.3 Translation of Cardinality Restrictions with Datatype Properties (P_D)

OWL2 allows the formation of class expression by applying restriction on the cardinality of a datatype property expression e.g. class of persons who have at least 2 mobile numbers. ***DataMinCardinality***, ***DataMaxCardinality*** and ***DataExactCardinality*** are the class expressions that can be used to express the class of individuals with at least, at most and exactly a specified number of connections respectively, to different literals, by a data property expression [3]. The following class expression contains all the individuals that are connected by the data property, i.e. hasHeartRate, to at least 4 different literals.

DataMinCardinality(4 a:hasHeartRate)

For cardinality restriction with datatype property, the corresponding relational query is required to retrieve values from relational database that satisfies "θ [value] P_D" condition, where θ in {≤ , ≥ , =}. The proposed mechanism to translate cardinality restriction with datatype property into SQL HAVING clause restriction is presented below as Algorithm 3.

Algorithm 3. Translation of Cardinality Restrictions with Datatype Properties (P_D)

```
h        ← is formulated SQL HAVING clause condition for $P_D$
$P_D$    ← in "θ [value] $P_D$", $P_D$ is a datatype property, where θ in {≤ , ≥ , =}
[value]  ← is a quantity value that is to be satisfied
if $P_D$ then
    if $P_D$ MinCardinality then
        h ← $P_D$ >= [value]
    else if $P_D$ MaxCardinality then
        h ← $P_D$ <= [value]
    else if $P_D$ ExactCardinality then
        h ← $P_D$ = [value]
    end if
    return h
end if
```

4.2.4 Translation of Qualified Cardinality Restrictions with Datatype Properties (P_D)

OWL2 allows the formation of class expression by applying qualified restriction on the cardinality of a datatype property expression e.g. class of batsmen who have scored at least 10 centuries in international cricket. ***DataMinCardinality***, ***DataMaxCardinality*** and ***DataExactCardinality*** are the class expressions that can be used to express the class of individuals which have at least, at most and exactly a specified number of connections respectively, to different literals that are in the qualifying data range, by a data property expression [3]. The following class expression contains all the individuals that are connected by the data property, i.e. hasBloodPressure, to at most 4 different literals of type highBloodPresure.

DataMaxCardinality(4 a:hasBloodPressure a:highBloodPresure)

For qualified cardinality restriction with datatype property, the corresponding relational query is required to retrieve values from relational database that satisfies "θ [value] P_D.DR" condition, where θ in {≤ , ≥ , =}. The proposed mechanism to translate qualified cardinality restriction with datatype property into SQL WHERE and HAVING clause restrictions is presented below as Algorithm 4.

Algorithm 4. Translation of Qualified Cardinality Restrictions with Datatype Properties (P_D)

```
Qf       ←   is a formulated SQL WHERE clause condition for qualified cardinality restriction
h        ←   is formulated SQL HAVING clause condition for $P_D$
$P_D$    ←   in "θ [value] $P_D$. DR", $P_D$ is a datatype property, where θ in {≤ , ≥ , =}
[value]  ←   is a quantity value that is to be satisfied
if $P_D$ then
  if $P_D$ MinCardinality then
    h  ←  $P_D$ >= [value]
  else if $P_D$ MaxCardinality then
    h  ←  $P_D$ <= [value]
  else if $P_D$ ExactCardinality then
    h  ←  $P_D$ = [value]
  end if
      if ∃ DR then
          if enumeration restriction is found then
              {enumerationValues ϵ ($v_1$, $v_2$, . . . , $v_n$) }
              $Q_f$ ← $P_D$ IN ($v_1$ ∪ $v_2$ ∪ . . . ∪ $v_n$)
          else if only minInclusive restriction is found then
              {restrictionValue V}
              $Q_f$ ← $P_D$ >= V
          else if only maxInclusive restriction is found then
              {restrictionValue V}
              $Q_f$ ← $P_D$ <= V
          else if both maxInclusive and minInclusive restrictions are found then
              {restrictionValues $V_1$, $V_2$}
              $Q_f$ ← $P_D$ BETWEEN $V_1$ AND $V_2$
```

```
                end if
        end if
    return $Q_f$, h
end if
```

4.2.5 Translation of Existential Quantification someValuesFrom (∃) with Datatype properties (P_D)

OWL2 allows the formation of class expression by applying existential quantification restriction over a datatype property expression e.g. class of batsmen who have scored triple centuries in international cricket. ***DataSomeValuesFrom*** is the class expression that can be used to express the class of individuals which are connected to at least one literal that is in the specified data range, by a data property expression [3]. The following class expression contains all the individuals that are connected by the data property, i.e. hasHeartRate, to at least one literal of type lowHeartRate.

DataSomeValuesFrom(a:hasHeartRate a:lowHeartRate)

For existential quantification restriction with datatype property, the corresponding relational query is required to retrieve values from relational database that satisfies "$\exists P_D$.DR" condition. The proposed mechanism to translate existential quantification restriction with datatype property into SQL WHERE clause restriction is presented below as Algorithm 5.

Algorithm 5. Translation of Existential Quantification someValuesFrom (∃) with Datatype Properties (P_D)

```
$Q_f$     $\leftarrow$    is a formulated SQL WHERE clause condition for $\exists P_D$.DR
$P_D$     $\leftarrow$    in "$\exists P_D$.DR", $P_D$ is a data type property
if ∃ DR then
        if enumeration restriction is found then
                {enumerationValues $\in$ ($v_1$, $v_2$, . . . , $v_n$) }
                $Q_f \leftarrow P_D$ IN ($v_1 \cup v_2 \cup . . . \cup v_n$)
        else if only minInclusive restriction is found then
```

```
                {restrictionValue V}
                $Q_f \longleftarrow P_D >= V$
        else if only maxInclusive restriction is found then
                {restrictionValue V}
                $Q_f \longleftarrow P_D <= V$
        else if both maxInclusive and minInclusive restrictions are found then
                {restrictionValues $V_1$, $V_2$}
                $Q_f \longleftarrow P_D$ BETWEEN $V_1$ AND $V_2$
        end if
        return $Q_f$
end if
```

4.2.6 Translation of Universal Quantification allValuesFrom (∀) with Datatype Properties (P_D)

OWL2 allows the formation of class expression by applying universal quantification restriction over a datatype property expression e.g. class of students who have secured 3+ GPA in every semester. ***DataAllValuesFrom*** is the class expression that can be used to express the class of individuals which are only connected to literals that are in the specified data range, by a data property expression [3]. The following class expression contains all the individuals that are connected by the data property, i.e. highBloodPresure, only to literals of type lowHeartRate.

DataAllValuesFrom(a:hasBloodPressure a:highBloodPresure)

For universal quantification restriction with datatype property, the corresponding relational query is required to retrieve values from relational database that satisfies "$\forall P_D.DR$" condition.

The DeMorgan's Law [37] state that: $\neg \forall x.\, p(x) = \exists x.\, \neg p(x)$ (a)

In our case, (a) could be written as: $\neg\forall P_D.\, DR = \exists P_D.\, \neg DR$ (b)

Applying negation on both sides of (b) results in: $\neg (\neg\forall P_D.\, DR) = \neg (\exists P_D.\, \neg DR)$

It can be further simplified as: $\forall P_D. DR = \neg (\exists P_D. \neg DR)$

Consequently, to translate universal quantification restriction into relational query, we first need to translate $\exists P_D.\neg DR$ into corresponding relational query expression. Then in order to deal with the outer negation, compliment operation is applied to the result set. The proposed mechanism to translate universal quantification restriction with datatype property into SQL WHERE clause restriction is presented below as Algorithm 6.

Algorithm 6. Translation of Universal Quantification allValuesFrom (∀) with Datatype Properties (P_D)

```
$Q_f$ $\leftarrow$ is a formulated SQL WHERE clause condition for $\forall P_D.DR$
$P_D$ $\leftarrow$ in "$\forall P_D. DR$", $P_D$ is a data type property
**if** $\forall$ DR **then**
    **if** enumeration restriction is found **then**
        {enumerationValues $\in$ ($v_1, v_2, \ldots, v_n$) }
        $Q_f \longleftarrow P_D$ NOT IN ($v_1 \cup v_2 \cup \ldots \cup v_n$)
    **else if** only minInclusive restriction is found **then**
        {restrictionValue V}
        $Q_f \longleftarrow P_D < V$
    **else if** onlymaxInclusive restriction is found **then**
        {restrictionValue V}
        $Q_f \longleftarrow P_D > V$
    **else if** both maxInclusive and minInclusive restrictions are found **then**
        {restrictionValues $V_1, V_2$}
        $Q_f \longleftarrow P_D$ NOT BETWEEN $V_1$ AND $V_2$
    **end if**
// START of the section taken from OntoQF [1]
    Generate SQL-where-clause-condition
    **begin** Generate $\forall$ Sub-query
        $I \longleftarrow$ SQL-where-clause-condition
```

```
                O $\longleftarrow$ a relational database query string
                O $\longleftarrow$ call OntoQF-Mapping (I) {see in section 4.3}
        end Generate $\forall$ Sub-query
        {now in order to handle outer NOT in the 'NOT (x.E NOT IN p (x))' expression}
        $Q_f \longleftarrow$ [domain-column of class C] NOT IN [O]
// END of the section taken from OntoQF
        return $Q_f$
end if
```

4.3 OntoQF Mapping Algorithm [1]-Extended

The OWL2-DL statements are individually translated into equivalent relational query constructs including GROUP BY and HAVING clauses. These constructs need further processing to formulate executable relational queries which includes; (1) mapping of semantic properties to corresponding relational database structure, (2) Integrations of results returned by translation algorithms to formulate SELECT, FROM and GROUP BY clauses in addition to join conditions. Ontology vocabulary is manually mapped to relational database vocabulary to fulfill the above mentioned requirements. The results of these mappings are stored for later use (as discussed in section 3.5). The mappings includes information about; datatype properties, object properties, aggregate functions, column names, table names, primary keys, foreign keys and database name. This mapping information is stored in a database table. The OntoQF extended mapping algorithm takes the previously formulated WHERE clause restrictions (as discussed in sections 4.2.2, 4.2.4, 4.2.5, 4.2.6) and HAVING clause restrictions (as discussed in sections 4.2.1, 4.2.2, 4.2.3, 4.2.4) as input and formulates the executable relational query. The algorithm to formulate complex relational queries using the mapping information is presented below with the proposed extensions as Algorithm 7.

Algorithm 7. OntoQF Extended Mapping Algorithm [1]

```
$T_1, T_2, T_3$, S, F, W, H, G $\longleftarrow$ Null { where $T_1, T_2, T_3$, S, F, H, G stores DB table & column names,
table joining conditions, DB table & aggregate functions, SQL SELECT, FROM, WHERE, HAVING
and GROUP BY clauses, respectively}.
```

```
W ⟵ $Q_f$ {is a translated object P and data type $P_D$ property restrictions generated by
Algorithms 2, 4, 5, 6}
H ⟵ h {is a translated object P and data type $P_D$ cardinality restrictions generated by
Algorithms 1, 2, 3, 4}.
$Q_F$ ⟵ {Null is a formulated SQL query}
// START of the section taken from OntoQF [1]
for each translated ontology property restriction do
    $T_1$ ⟵ stores database table and column information using ontology to database mappings
        for $P/P_D$ property restrictions
    W ⟵ replace $P/P_D$ in W with $T_1$
end for
// END of the section taken from OntoQF
for each translated ontology cardinality restriction do
    $T_3$ ⟵ stores database table and aggregate function information using ontology to
        database mappings for $P/P_D$ cardinality restriction
    H ⟵ replace $P/P_D$ in H with $T_3$
end for
// START of the section taken from OntoQF [1]
$T_1$ ⟵ remove duplicate table names from $T_1$
for each participating database table in $T_1$ do
    F ⟵ append table names to F separating by comma ( , )
    $T_2$ ⟵ save related join conditions
end for
F ⟵ generate FROM clause with F
$T_2$ ⟵ remove duplicate join conditions from $T_2$
for every participating join condition in $T_2$ do
    W ⟵ append join conditions to W with AND operator
end for
S ⟵ retrieve user selected data columns
```

```
// END of the section taken from OntoQF
 $T_3 \longleftarrow$ remove duplicate aggregate functions from $T_3$
for every participating aggregate function in $T_3$ do
    S $\longleftarrow$ append aggregate functions to S separating by comma ( , )
end for
        /* In case at least one aggregate function in present in select clause, all the selections
        other than aggregate functions must me listed in group by clause to formulate a correct
        relational query */
if any aggregate function(s) in S then
    for every participating non-aggregate selection in S do
        G $\longleftarrow$ append non-aggregate selected columns to G separating by comma ( , )
    end for
end If
$Q_F \longleftarrow$ append S, F, W, G and H
Return $Q_F$
```

4.4 Chapter Summary and Conclusion

The extended ontology driven relational query formulation process has been discussed in this chapter. It has been discussed and concluded that the proposed extension to OntoQF can enable the translation of some more semantic statements into executable complex relational queries. Extensions to the translation algorithms are proposed for the independent translation of OWL2-DL constructs into corresponding relational query constructs including the GROUP BY and HAVING clauses. Extended translation algorithms translate individual OWL2-DL constructs into WHERE and HAVING clause restrictions. The ontology vocabulary is then to be replaced with the database vocabulary to make it compatible with the underlying schema. For this purpose, extension to the mapping algorithm is proposed to: (1) integrate the restrictions generated by independent translation algorithms, and (2) to map the ontology vocabulary to the underlying relational database vocabulary. The result of the mapping algorithm is the complex relational query that can be executed on the underlying relational database. The extended OntoQF can be used with any relational database schema due to the independent translation of

OWL2-DL constructs into corresponding relational construct. In this research, the accuracy of generated queries is investigated by applying extended OntoQF in a practical domain. A decent part of integrated Health-e-Child patients’ database schema [38] is chosen for this purpose which is the same part that was used to test the OntoQF [1].

Chapter 5: Experimental Evaluation of the Proposed Extensions to OntoQF

5.1 Introduction

This chapter discusses the details of the evaluation phase carried out to validate the extended query formulation process. The Health-e-Child (HeC) [38] database of patients and their medical data is used for evaluation which is the same as was used to validate OntoQF. The reason for this selection, as given in [1], is that the medical data is typically very large and complicated in nature and writing relational queries for such data is a challenging task. Section 5.2 discusses the implementation of prototype and the steps followed for the translation of semantic statements into relational query expressions are discussed in Section 5.3. Section 5.4 presents the application of the extended query formulation process with the help of case studies. Section 5.5 brings into light the evaluation strategies adopted to validate this research work and details of the adopted approach are presented in Section 5.6. Results obtained from the evaluation process are explained in Section 5.7. In the end, this research work is concluded by answering the research hypothesis in the light of research questions.

5.2 The Implementation of Extended OntoQF Prototype System

The extended OntoQF architecture is depicted here from prototype's implementation perspective. The aim of the prototype implementation is to formulate complex relational queries using the domain knowledge. The Graphical User Interface (GUI) enables the users to search and select the domain knowledge concepts. The domain knowledge concepts are used to generate corresponding relational queries which are executed on underlying relational database and results are viewed by the users as the output. The extended OntoQF prototype implements the major extensions proposed in the relational query formulation process. The functionality is implemented by addressing both; (1) Translation of sophisticated ontology statements into equivalent relational query expressions including GROUP BY and HAVING (as discussed in section 3.4), and (2) Ontology to database mappings (as discussed in section 3.5).

The rules for; the transformation of domain specific knowledge into domain ontology and expression of domain knowledge in domain ontology are proposed by K. Munir [1]. The OntoQF is initiated, as per proposed rules, using a decent part of integrated Health-e-Child patients' database schema [38]. The same part of integrated Health-e-Child patients' database schema is used for the evaluation of the extensions proposed in the thesis. In this research, support for OWL-2 is proposed in order to exploit sophisticated OWL2 constructs for defining domain knowledge than may help in generation of sophisticated relational queries.

The listed environment, tools and utilities have been used for the extended OntoQF prototype development:

- Microsoft Windows 7 [39]: This operating system is used for the implementation and testing of the extended OntoQF prototype.
- NetBeans IDE 8.0.2 [40]: It is used to implement the major extensions in relational query formulation process in Java.
- MySQL version 5.6.12 [41]: It is used as the backend relational database. It is also used to store mapping information and domain knowledge for the extended OntoQF prototype.

The prototype is implemented as a proof of concept for the proposed extensions to the query formulation process. Relational database (MySQL) is used at the backend to store the domain knowledge and mappings required for the executable relational query formulation. The database table implemented to store extended database to ontology mappings is shown in table 5.1. The mappings are extended by adding details of aggregate functions which enable OntoQF to provide support for (qualified) cardinality restrictions. The implementation of a fully functional system with the extended functionality is left as future work.

Column Name	Constraint
Property_ID (PK)	Unique/ Not Null
PropertyName	Unique/ Not Null
PropertyType	Not Null
PropertyTable	Not Null
PropertyColumn	Not Null

OtherTables	Not Null
AggregateFunction	Not Null
JoinConditions	Not Null

Table 5.1: Table to store mappings in Extended OntoQF

5.3 Translation of Semantic Statements into Relational Query Expressions

The query formulation module of OntoQF [1] translates the semantic statement into executable relational query expression according to the underlying database schema. This thesis extends the query formulation module by proposing extensions to both translation algorithms and mapping algorithm as discussed in Sections 4.2 and 4.3. The extended translation is carried out by following similar steps as that of original OntoQF [1] which are presented below:

Step 1: OWL2-DL statement is written by domain experts, using domain knowledge expressed in ontology, in accordance to the eligibility criteria.

Step 2: Semantic statements are translated to WHERE and HAVING clause restrictions by the extended algorithms for semantic statements to relational query translation

Step 3: Extended ontology to database mapping algorithm maps the generated restrictions to the underlying database schema.

Step 4: Extended ontology to database mapping algorithm generate the SELECT, FROM and GROUP BY clauses in addition to the join conditions.

Step5: In the end, the extended mapping algorithm join together the generated SELECT, FROM, WHERE, GROUP BY and HAVING clauses and append the generated join conditions with the WHERE clause in order to formulate the executable relation query corresponding to the underlying database schema.

5.4 The Application of the Extended Query Formulation Process

The OntoQF [1] was instantiated using HeC case studies [38] from medical domain. The Extended system is also instantiated using case studies from medical domain. The aim of using medical data is to mine interesting patterns from different clinical parameters in patients' history.

The result of the case study is the set of patients' data that matches the restrictions mentioned in the search criteria. In this research, we are only concerned with the valid relational queries generation so details of medical background and clinical objectives are not discussed. The template used to describe the clinical case studies is depicted in Table 5.2. Three case studies are presented here along with their clinical objectives and resultant formulated queries.

Sr. No.	Element	Description
1	Study Number	Number to uniquely identify a case study
2	Title of Study	Brief description of the case study
3	Eligibility Criteria	Criteria to be satisfied by a patient to be a part of the result set
4	Extended OntoQF OWL2-DL Expression	Representation of eligibility criteria using OWL2-DL expressions formulated by the domain expert
5	Extended OntoQF Formulated Relational Query	A valid relational query formulated by the extended OntoQF according to the underlying database schema

Table 5.2: Template for Extend OntoQF Case Studies Pattern

5.4.1 Clinical Case Study-1

This case study has the clinical objective of mining the relationship between any type of heart surgery and excessive use of medicine in case patient is over-weight. The computational purpose of the study is to retrieve the record of patients who match the eligibility criteria for this study.

Title of Study:

"Patients who had any type of heart surgery and have excessive medicine prescription history and have history of being over-weight"

Eligibility Criteria:

Patients must meet the below mentioned criteria in order to be included in the result set:

- Have a history of any type of heart surgery
- Have a history of excessive medicine prescription
- Have a history of being over-weight

Extended OntoQF OWL2-DL Expression:

“Patients AND

Ǝ hasClinicalObservation HeartSurgery AND

PrescribedDrugs minCardinality 3 AND

Ǝ hasWeight overWeight"

Extended OntoQF Formulated Relational Query:

“**SELECT** patient.firstname, clinicalobservation.observationName,
COUNT(drug_therapy.Drug_id), clinicalexamination.Weight
FROM clinicalobservation, Patient, clinicalexamination, clinicaltestobservation, drug, drug_therapy
WHERE Patient.Patient_ID = clinicalexamination.Patient_ID AND
clinicalExamination.Examination_ID = clinicaltestobservation.Examination_ID AND
Clinicaltestobservation.ClinicalObservation_ID = clinicalobservation.ClinicalObservation_ID
AND ***clinicalobservation.observationName IN ('TotalHeartSurgery' , 'PartialHeartSurgery' , 'NearTotalHeartSurgery' , 'HeartSurgeryBiopsy')*** AND clinicalExamination.Examination_ID = drug_therapy.Examination_ID AND drug_therapy.Drug_id = drug.Drug_id AND
clinicalexamination.Weight > 80
GROUP BY patient.firstname, clinicalobservation.observationName,
clinicalexamination.Weight
HAVING ***COUNT(drug_therapy.Drug_id)>=3***”

5.4.2 Clinical Case Study-2

This case study has the clinical objective of mining the relationship between use of anti depressions along-with history of high blood pressure and has clinical observation of many abnormalities. The computational purpose of the study is to retrieve the record of patients who match the eligibility criteria for this study.

Title of Study:

“Patients who have a history of frequent high blood pressure along-with usage of antidepressants and have clinical observation of many abnormalities”

Eligibility Criteria:

Patients must meet the below mentioned criteria in order to be included in the result set:

- Have a history of frequent high blood pressure
- Have used antidepressants
- Have clinical observation of many abnormalities

Extended OntoQF OWL2-DL Expression:

“Patients AND

hasBloodPressure minCardinality 2 highBloodPresure AND

Ǝ PrescribedDrugs Antidepression AND

hasClinicalObservation minCardinality 2 Abnormalities”

Extended OntoQF Formulated Relational Query:

“**SELECT** patient.firstname, COUNT(clinicalexamination.BloodPressure), drug.Drug_name, COUNT(clinicaltestobservation.ClinicalObservation_ID)
FROM clinicalexamination, Patient, drug, drug_therapy, clinicalobservation, clinicaltestobservation
WHERE Patient.Patient_ID = clinicalexamination.Patient_ID AND ***clinicalexamination.BloodPressure >= 129*** AND clinicalExamination.Examination_ID = drug_therapy.Examination_ID AND drug_therapy.Drug_id = drug.Drug_id AND ***drug.Drug_name IN ('Ativan_lorazepam' , 'Celexa_citalopram' , 'Prozac_fluoxetine' , 'Tenormin_atenolol' , 'Xanax_alprazolam')*** AND clinicalExamination.Examination_ID =

clinicaltestobservation.Examination_ID AND Clinicaltestobservation.ClinicalObservation_ID = clinicalobservation.ClinicalObservation_ID AND ***clinicalobservation.observationName IN ('EatingAbilityWithDifficulty' , 'GettingUpAbilityDifficulty' , 'VascularAbnormalities' , 'WalkingDifficulty')***
GROUP BY patient.firstname, drug.Drug_name
HAVING ***COUNT(clinicalexamination.BloodPressure)>=2*** AND
COUNT(clinicaltestobservation.ClinicalObservation_ID)>=2"

5.4.3 Clinical Case Study-3

This case study has the clinical objective of mining the relationship between excessive use of 'Amoxil_amoxicillin' along-with clinical observation of many infections and the history of regular high blood pressure. The computational purpose of the study is to retrieve the record of patients who match the eligibility criteria for this study.

Title of Study:

"Patients who have an excessive prescription history of 'Amoxil_amoxicillin' along-with clinical observation of many infections and have history of regular high blood pressure"

Eligibility Criteria:

Patients must meet the below mentioned criteria in order to be included in the result set:

- Have clinical observation of many infections
- Have a history of excessive 'Amoxil_amoxicillin' prescription
- Have a history of regular high blood pressure

Extended OntoQF OWL2-DL Expression:

"Patients AND

hasClinicalObservation minCardinality 3 Infections AND

PrescribedDrugs minCardinality 2 Amoxil_amoxicillin AND

∀ hasBloodPressure highBloodPresure"

Extended OntoQF Formulated Relational Query:

"**SELECT** patient.firstname, COUNT(clinicaltestobservation.ClinicalObservation_ID), COUNT(drug_therapy.Drug_id), clinicalexamination.BloodPressure
FROM clinicalobservation, Patient, clinicalexamination, clinicaltestobservation, drug, drug_therapy
WHERE Patient.Patient_ID = clinicalexamination.Patient_ID AND clinicalExamination.Examination_ID = clinicaltestobservation.Examination_ID AND Clinicaltestobservation.ClinicalObservation_ID = clinicalobservation.ClinicalObservation_ID AND ***clinicalobservation.observationName IN ('Bullous_Impetigo' , 'Gonorrhoeae' , 'H_Influenzae' , 'Rat_Bite_Fever' , 'Scaldedskin_Syndrome' , 'Gonorrhoeae' , 'Streptococcus' , 'Enteroviruses' , 'Herpes_Simplex_Virus' , 'Varicella_Virus')*** AND clinicalExamination.Examination_ID = drug_therapy.Examination_ID AND drug_therapy.Drug_id = drug.Drug_id AND ***drug.Drug_name ='Amoxil_amoxicillin'*** AND ***patient.Patient_ID NOT IN*** (**SELECT** patient.Patient_ID **FROM** Patient, clinicalexamination **WHERE** Patient.Patient_ID = clinicalexamination.Patient_ID and ***NOT***(***clinicalexamination.BloodPressure >= 129***))
GROUP BY patient.firstname, clinicalexamination.BloodPressure
HAVING ***COUNT(clinicaltestobservation.ClinicalObservation_ID)>=3*** AND ***COUNT(drug_therapy.Drug_id)>=2***"

5.5 The Evaluation Process for the Proposed Extensions to OntoQF

The basic purpose of evaluating the proposed extensions to OntoQF is to assess the extent to which the proposed support of GROUP BY and HAVING clauses using OWL2-DL can enhance the usefulness of the OntoQF [1]. The evaluation process does not aim to assess other computational attributes such as security, query optimization etc. In this research, we are only concerned with the valid relational queries generation which is evaluated by checking the correctness of the queries generated by the extended system. In Section 4.2, it was discussed that the extended system proposes support for six OWL 2-DL constructs that can be used in any order to define semantic statements containing domain knowledge. A '*Matrix of Validation Scenarios*' has been designed and presented in Table 5.3 in order to cover all the possible combinations of

OWL 2-DL constructs. The primary aims of implementing the '*Matrix of Validation Scenarios*' as given in [1] are; clearly show which OWL 2-DL to relational query translations are covered in the testing process, check the accuracy of the generated queries, indentify more situations that result into formulation of unsatisfiable relational queries and to make sure that all the possible combinations of OWL2- DL constructs are considered in the evaluation process. There are total thirty six unique combinations among of total sixty four combinations as depicted in Table 5.3.

5.6 Empirical Evaluation of the Proposed Extension using the Matrix of Validation Scenarios

A comprehensive evaluation is carried out in order to check the correctness of the proposed extension. In addition to the individual OWL2-DL construct translations, all the possible combinations of constructs are executed with the assistance of the Matrix of Validation Scenarios presented in the previous section. In the current section, only those scenarios are presented which have failed the tests and have resulted into unsatisfiable relational queries whereas ***results of all the executions are fully reported in Appendix A***. The template used to describe the detailed outcomes of the test executions is depicted in Table 5.4.

	∃ with O.P	∀ with O.P	∃ with D.P	∀ with D.P	Car with O.P	QCR with O.P	Cardinality with D.P	QCR with D.P
∃ with O.P	Y	N	N	N	N	Y	N	N
∀ with O.P	N*	N	N	N	N	N	N	N
∃ with D.P	N*	N*	Y	N	N	N	N	Y
∀ with D.P	N*	N*	N*	N	N	N	N	N
Car with O.P	N*	N*	N*	N*	N	N	N	N
QCR with O.P	Y*	N*	N*	N*	N*	Y	N	N
Cardinality with D.P	N*	N*	N*	N*	N*	N*	N	N
QCR with D.P	N*	N*	Y*	N*	N*	N*	N*	Y

Table 5.3: Extended OntoQF experimental Evaluation Coverage Matrix

N: Correct query formulation

Y: May lead to the generation of unsatisfiable queries

*** Repeating cases**

Sr. No.	Element	Description
1	Study Number	Number to uniquely identify a test execution
2	Use of OWL2-DL Constructs	The OWL2-DL constructs as per the Matrix of Validation Scenarios
3	OWL2-DL Statements	Representation of eligibility criteria using OWL2-DL expressions formulated by the domain expert
4	Extended OntoQF Formulated Relational Query	The relational query formulated by the extended OntoQF according to the underlying database schema
5	Validation Results	The evaluation results as Pass or Fail. In case of failure, a brief description of the reason is also described

Table 5.4: Template for Detailed Outcomes of the Test Executions

5.7 Assessment of Results Obtained from Evaluation Process

It has been observed in the Evaluation phase that the proposed extension is capable of translating all the selected OWL2-DL constructs, in different combinations and orders, into corresponding relational database constructs. The evaluation process also exposed some more scenarios that result into unsatifiable translations. In case of unsatisfiable translations, the generated queries may not be able to produce correct results according to the ontological descriptions. K. Munir proposed the use of non-materialized database views to handle the unsatisfiable translations [1]. In the research the identified unsatisfiable semantic expression types are: (1) $\exists$ P.C $\cap$ $\exists$ P.C, (2) $\exists$ P.C $\cap$ θ [value] P. C, where θ in $\{\leq, \geq, =\}$, (3) $\exists$ P_D.DR $\cap$ $\exists$ P_D.DR, (4) $\exists$ P_D.DR $\cap$ θ [value] P_D.DR, where θ in $\{\leq, \geq, =\}$, (5) θ [value] P. C $\cap$ θ [value] P. C, where θ in $\{\leq, \geq, =\}$ and (6) θ [value] P_D.DR $\cap$ θ [value] P_D.DR, where θ in $\{\leq, \geq, =\}$. All these unsatisable translation scenarios can be described by the following rule:

"*For a given Ontological statement, the OWL2-DL to relational query translation is not satisfiable if the OWL2-DL constructs include a conjunction operation between any combination of existential quantification restriction and qualified cardinality restriction on distinct ontology classes using the similar Object Property/Datatype Property*"

Details of all the exposed unsatisfiable scenarios are discussed in the following subsections.

5.7.1 ∃ P.C ∩ ∃ P.C

The translation of an OWL2-DL statement that includes an intersection operation between existential quantification restrictions on distinct ontology classes using a similar object property results into the formulation of unsatisfiable relational query. Such formulated relational query may not be able to retrieve the records from relational database according to the ontology description. Evaluation test results for a similar clinical study are presented in Table 5.5 where the following semantic statement was translated into a relational query:

Patients AND ∃ PrescribedDrugs Antidepression AND ∃ PrescribedDrugs Panadol

This above mentioned semantic statement refers to the patients who have been prescribed both anti-depression and Panadol in their recorded medical history. The translation of this semantic statement resulted into the following relational query expression:

SELECT <Skipped>
FROM drug, Patient, clinicalexamination, drug_therapy
WHERE <Join conditions> AND
drug.Drug_name IN ('Ativan_lorazepam' , 'Celexa_citalopram' , 'Prozac_fluoxetine' , 'Tenormin_atenolol' , 'Xanax_alprazolam') AND **drug.Drug_name** ='Panadol'

In this case, the translation of the OWL2-DL statement results into generation of a SQL query that applies two different WHERE clause conditions on the same column of a relational database schema, which may retrieve no results. For example in the case study under consideration, the WHERE clause conditions **drug.Drug_name** IN ('Ativan_lorazepam' , 'Celexa_citalopram' , 'Prozac_fluoxetine' , 'Tenormin_atenolol' , 'Xanax_alprazolam') AND **drug.Drug_name** ='Panadol' can never be true at the same time.

Study Number	1
OWL2-DL Constructs Used	∃ P.C ∩ ∃ P.C
OWL2-DL Statements	Patients AND ∃ PrescribedDrugs Antidepression AND ∃

	PrescribedDrugs Panadol
Extended OntoQF Formulated Relational Query	SELECT patient.firstname, drug.Drug_name FROM drug, Patient, clinicalexamination, drug_therapy WHERE Patient.Patient_ID = clinicalexamination.Patient_ID AND clinicalExamination.Examination_ID = drug_therapy.Examination_ID AND drug_therapy.Drug_id = drug.Drug_id AND drug.Drug_name IN ('Ativan_lorazepam' , 'Celexa_citalopram' , 'Prozac_fluoxetine' , 'Tenormin_atenolol' , 'Xanax_alprazolam') AND drug.Drug_name ='Panadol'
Validation Results	Failed (Reason: OWL2-DL constructs include a conjunction operation between existential quantification restrictions using the similar Object Property that results into an unsatisfiable relational query)

Table 5.5: Evaluation Results for Semantic Expression Ǝ P.C ∩ Ǝ P.C

5.7.2 Ǝ P.C ∩ θ [value] P. C, where θ in {≤ , ≥ , =}

The translation of an OWL2-DL statement that includes an intersection operation between an existential quantification restriction and a qualified cardinality restriction on distinct ontology classes using a similar object property results into the formulation of unsatisfiable relational query. Such formulated relational query may not be able to retrieve the records from relational database according to the ontology description. Evaluation test results for a similar clinical study are presented in Table 5.6 where the following semantic statement was translated into a relational query:

Patients AND Ǝ PrescribedDrugs Antidepression AND PrescribedDrugs minCardinality 2 Antibiotic

This above mentioned semantic statement refers to the patients who have been prescribed Antibiotic at least twice and also have been prescribed anti-depression in their recorded medical history. The translation of this semantic statement resulted into the following relational query expression:

SELECT <Skipped>, COUNT(drug_therapy.Drug_id)

FROM drug, Patient, clinicalexamination, drug_therapy

WHERE <Join conditions> AND

drug.Drug_name IN ('Ativan_lorazepam' , 'Celexa_citalopram' , 'Prozac_fluoxetine' , 'Tenormin_atenolol' , 'Xanax_alprazolam') AND **drug.Drug_name** IN ('Amoxil_amoxicillin' , 'Cipro_ciprofloxacin' , 'Polymox_amoxicillin' , 'Trimox_amoxicillin' , 'Vibramycin_doxycycline')

GROUP BY <Skipped>

HAVING COUNT(drug_therapy.Drug_id)>=2

In this case, the translation of the OWL2-DL statement results into generation of a SQL query that applies two different WHERE clause conditions on the same column of a relational database schema, which may retrieve no results. For example in the case study under consideration, the WHERE clause conditions **drug.Drug_name** IN ('Ativan_lorazepam' , 'Celexa_citalopram' , 'Prozac_fluoxetine' , 'Tenormin_atenolol' , 'Xanax_alprazolam') AND **drug.Drug_name** IN ('Amoxil_amoxicillin' , 'Cipro_ciprofloxacin' , 'Polymox_amoxicillin' , 'Trimox_amoxicillin' , 'Vibramycin_doxycycline') can never be true at the same time.

Study Number	2
OWL2-DL Constructs Used	Ǝ P.C ∩ θ [value] P. C, where θ in {≤ , ≥ , =}
OWL2-DL Statements	Patients AND Ǝ PrescribedDrugs Antidepression AND PrescribedDrugs minCardinality 2 Antibiotic
Extended OntoQF Formulated Relational Query	SELECT patient.firstname, drug.Drug_name, COUNT(drug_therapy.Drug_id) FROM drug, Patient, clinicalexamination, drug_therapy WHERE Patient.Patient_ID = clinicalexamination.Patient_ID AND clinicalExamination.Examination_ID = drug_therapy.Examination_ID AND drug_therapy.Drug_id = drug.Drug_id AND drug.Drug_name IN ('Ativan_lorazepam' , 'Celexa_citalopram' , 'Prozac_fluoxetine' , 'Tenormin_atenolol' , 'Xanax_alprazolam') AND drug.Drug_name IN

	('Amoxil_amoxicillin' , 'Cipro_ciprofloxacin' , 'Polymox_amoxicillin' , 'Trimox_amoxicillin' , 'Vibramycin_doxycycline') GROUP BY patient.firstname, drug.Drug_name HAVING COUNT(drug_therapy.Drug_id)>=2
Validation Results	Failed (Reason: OWL2-DL constructs include a conjunction operation between an existential quantification restriction and a qualified cardinality restriction using the similar Object Property that results into an unsatisfiable relational query)

Table 5.6: Evaluation Results for Semantic Expression ∃ P.C ∩ θ [value] P. C, where θ in {≤ , ≥ , =}

5.7.3 ∃ P_D.DR ∩ ∃ P_D.DR

The translation of an OWL2-DL statement that includes an intersection operation between existential quantification restrictions on distinct data ranges using a similar datatype property results into the formulation of unsatisfiable relational query. Such formulated relational query may not be able to retrieve the records from relational database according to the ontology description. Evaluation test results for a similar clinical study are presented in Table 5.7 where the following semantic statement was translated into a relational query:

Patients AND ∃ hasBloodPressure highBloodPresure AND ∃ hasBloodPressure normalBloodPresure

This above mentioned semantic statement refers to the patients who have both high blood pressure and low blood pressure measurements in their recorded medical history. The translation of this semantic statement resulted into the following relational query expression:

SELECT <Skipped>
FROM clinicalexamination, Patient
WHERE <Join conditions> AND
clinicalexamination.BloodPressure >= 130 AND **clinicalexamination.BloodPressure** BETWEEN 110 AND 129

In this case, the translation of the OWL2-DL statement results into generation of a SQL query that applies two different WHERE clause conditions on the same column of a relational database schema, which may retrieve no results. For example in the case study under consideration, the WHERE clause conditions **clinicalexamination.BloodPressure** >= 130 AND **clinicalexamination.BloodPressure** BETWEEN 110 AND 129 can never be true at the same time.

Study Number	3
OWL2-DL Constructs Used	$\exists P_D.DR \cap \exists P_D.DR$
OWL2-DL Statements	Patients AND ∃ hasBloodPressure highBloodPresure AND ∃ hasBloodPressure normalBloodPresure
Extended OntoQF Formulated Relational Query	SELECT patient.firstname, clinicalexamination.BloodPressure FROM clinicalexamination, Patient WHERE Patient.Patient_ID = clinicalexamination.Patient_ID AND clinicalexamination.BloodPressure >= 130 AND clinicalexamination.BloodPressure BETWEEN 110 AND 129
Validation Results	Failed (Reason: OWL2-DL constructs include a conjunction operation between existential quantification restrictions using the similar Datatype Property that results into an unsatisfiable relational query)

Table 5.7: Evaluation Results for Semantic Expression $\exists P_D.DR \cap \exists P_D.DR$

5.7.4 $\exists P_D.DR \cap \theta$ [value] $P_D.DR$, where θ in $\{\leq, \geq, =\}$

The translation of an OWL2-DL statement that includes an intersection operation between an existential quantification restriction and a qualified cardinality restriction on distinct data ranges using a similar datatype property results into the formulation of unsatisfiable relational query. Such formulated relational query may not be able to retrieve the records from relational database according to the ontology description. Evaluation test results for a similar clinical study are presented in Table 5.8 where the following semantic statement was translated into a relational query:

Patients AND ∃ hasBloodPressure normalBloodPresure AND hasBloodPressure minCardinality 2 highBloodPresure

This above mentioned semantic statement refers to the patients who have high blood pressure measurements at least twice and also have low blood pressure measurements in their recorded medical history. The translation of this semantic statement resulted into the following relational query expression:

SELECT <Skipped>, COUNT(clinicalexamination.BloodPressure)
FROM clinicalexamination, Patient
WHERE <Join conditions> AND
clinicalexamination.BloodPressure BETWEEN 110 AND 129 AND
clinicalexamination.BloodPressure >= 130
GROUP BY <Skipped>
HAVING COUNT(clinicalexamination.BloodPressure)>=2

In this case, the translation of the OWL2-DL statement results into generation of a SQL query that applies two different WHERE clause conditions on the same column of a relational database schema, which may retrieve no results. For example in the case study under consideration, the WHERE clause conditions **clinicalexamination.BloodPressure** BETWEEN 110 AND 129 AND **clinicalexamination.BloodPressure** >= 130 can never be true at the same time.

Study Number	4
OWL2-DL Constructs Used	∃ P_D.DR ∩ θ [value] P_D.DR, where θ in {≤ , ≥ , =}
OWL2-DL Statements	Patients AND ∃ hasBloodPressure normalBloodPresure AND hasBloodPressure minCardinality 2 highBloodPresure
Extended OntoQF Formulated Relational Query	SELECT patient.firstname, clinicalexamination.BloodPressure, COUNT(clinicalexamination.BloodPressure) FROM clinicalexamination, Patient WHERE Patient.Patient_ID = clinicalexamination.Patient_ID AND clinicalexamination.BloodPressure BETWEEN 110 AND 129 AND

	clinicalexamination.BloodPressure >= 130 GROUP BY patient.firstname, clinicalexamination.BloodPressure HAVING COUNT(clinicalexamination.BloodPressure)>=2
Validation Results	Failed (Reason: OWL2-DL constructs include a conjunction operation between an existential quantification restriction and a qualified cardinality restriction using the similar Datatype Property that results into an unsatisfiable relational query)

Table 5.8: Evaluation Results for Semantic Expression $\exists P_D.DR \cap \theta$ [value] $P_D.DR$, where θ in $\{\leq, \geq, =\}$

5.7.5 θ [value] P. C $\cap$ θ [value] P. C, where θ in $\{\leq, \geq, =\}$

The translation of an OWL2-DL statement that includes an intersection operation between qualified cardinality restrictions on distinct ontology classes using a similar object property results into the formulation of unsatisfiable relational query. Such formulated relational query may not be able to retrieve the records from relational database according to the ontology description. Evaluation test results for a similar clinical study are presented in Table 5.9 where the following semantic statement was translated into a relational query:

Patients AND PrescribedDrugs minCardinality 2 Amoxil_amoxicillin AND PrescribedDrugs minCardinality 3 Antibiotic

This above mentioned semantic statement refers to the patients who have been prescribed Antibiotic at least thrice and also have been prescribed Amoxil_amoxicillin at least twice in their recorded medical history. The translation of this semantic statement resulted into the following relational query expression:

SELECT <Skipped>, COUNT(drug_therapy.Drug_id)
FROM drug, Patient, clinicalexamination, drug_therapy
WHERE <Join conditions> AND
drug.Drug_name ='Amoxil_amoxicillin' AND **drug.Drug_name** IN ('Amoxil_amoxicillin' , 'Cipro_ciprofloxacin' , 'Polymox_amoxicillin' , 'Trimox_amoxicillin' , 'Vibramycin_doxycycline')
GROUP BY <Skipped>

HAVING COUNT(drug_therapy.Drug_id)>=2 AND COUNT(drug_therapy.Drug_id)>=3

In this case, the translation of the OWL2-DL statement results into generation of a SQL query that applies two different WHERE clause conditions on the same column of a relational database schema, which may retrieve no results. For example in the case study under consideration, the WHERE clause conditions **drug.Drug_name** ='Amoxil_amoxicillin' AND **drug.Drug_name** IN ('Amoxil_amoxicillin' , 'Cipro_ciprofloxacin' , 'Polymox_amoxicillin' , 'Trimox_amoxicillin' , 'Vibramycin_doxycycline') can never be true at the same time.

Study Number	5
OWL2-DL Constructs Used	θ [value] P. C ∩ θ [value] P. C, where θ in {≤ , ≥ , =}
OWL2-DL Statements	Patients AND PrescribedDrugs minCardinality 2 Amoxil_amoxicillin AND PrescribedDrugs minCardinality 3 Antibiotic
Extended OntoQF Formulated Relational Query	SELECT patient.firstname, COUNT(drug_therapy.Drug_id) FROM drug, Patient, clinicalexamination, drug_therapy WHERE Patient.Patient_ID = clinicalexamination.Patient_ID AND clinicalExamination.Examination_ID = drug_therapy.Examination_ID AND drug_therapy.Drug_id = drug.Drug_id AND drug.Drug_name ='Amoxil_amoxicillin' AND drug.Drug_name IN ('Amoxil_amoxicillin' , 'Cipro_ciprofloxacin' , 'Polymox_amoxicillin' , 'Trimox_amoxicillin' , 'Vibramycin_doxycycline') GROUP BY patient.firstname HAVING COUNT(drug_therapy.Drug_id)>=2 AND COUNT(drug_therapy.Drug_id)>=3
Validation Results	Failed (Reason: OWL2-DL constructs include a conjunction operation between qualified cardinality restrictions using the similar Object Property that results into an unsatisfiable relational query)

Table 5.9: Evaluation Results for Semantic Expression θ [value] P. C ∩ θ [value] P. C, where θ in {≤ , ≥ , =}

5.7.6 θ [value] P_D.DR ∩ θ [value] P_D.DR, where θ in {≤ , ≥ , =}

The translation of an OWL2-DL statement that includes an intersection operation between qualified cardinality restrictions on distinct data ranges using a similar datatype property results into the formulation of unsatisfiable relational query. Such formulated relational query may not be able to retrieve the records from relational database according to the ontology description. Evaluation test results for a similar clinical study are presented in Table 5.10 where the following semantic statement was translated into a relational query:

Patients AND hasBloodPressure minCardinality 3 highBloodPresure AND hasBloodPressure minCardinality 2 lowBloodPresure

This above mentioned semantic statement refers to the patients who have high blood pressure measurements at least thrice and also have low blood pressure measurements at least twice in their recorded medical history. The translation of this semantic statement resulted into the following relational query expression:

SELECT <Skipped>, COUNT(clinicalexamination.BloodPressure)
FROM clinicalexamination, Patient
WHERE <Join conditions> AND
<u>**clinicalexamination.BloodPressure** >= 130 AND **clinicalexamination.BloodPressure** <= 110</u>
GROUP BY <Skipped>
HAVING COUNT(clinicalexamination.BloodPressure)>=3 AND
COUNT(clinicalexamination.BloodPressure)>=2

In this case, the translation of the OWL2-DL statement results into generation of a SQL query that applies two different WHERE clause conditions on the same column of a relational database schema, which may retrieve no results. For example in the case study under consideration, the WHERE clause conditions <u>**clinicalexamination.BloodPressure** >= 130 AND **clinicalexamination.BloodPressure** <= 110</u> can never be true at the same time.

Study Number	6
OWL2-DL Constructs Used	θ [value] P_D.DR ∩ θ [value] P_D.DR, where θ in {≤ , ≥ , =}
OWL2-DL Statements	Patients AND hasBloodPressure minCardinality 3 highBloodPresure AND hasBloodPressure minCardinality 2 lowBloodPresure
Extended OntoQF Formulated Relational Query	SELECT patient.firstname, COUNT(clinicalexamination.BloodPressure) FROM clinicalexamination, Patient WHERE Patient.Patient_ID = clinicalexamination.Patient_ID AND clinicalexamination.BloodPressure >= 130 AND clinicalexamination.BloodPressure <= 110 GROUP BY patient.firstname HAVING COUNT(clinicalexamination.BloodPressure)>=3 AND COUNT(clinicalexamination.BloodPressure)>=2
Validation Results	Failed (Reason: OWL2-DL constructs include a conjunction operation between qualified cardinality restrictions using the similar Datatype Property that results into an unsatisfiable relational query)

Table 5.10: Evaluation Results for Semantic Expression θ [value] P_D.DR ∩ θ [value] P_D.DR, where θ in {≤ , ≥ , =}

5.8 Concluding this Research and Answering the Research Hypothesis

In order to answer the Research Hypothesis and associated research questions present in Chapter 1; the proposed extensions to the algorithms (Chapter 4) and instantiation of the prototype (Chapter 5) have been empirically evaluated using the adopted evaluation process (Chapter 5). Research questions that were formulated to prove/disprove the hypothesis are answered in this section. The research hypothesis states that:

"*Support of GROUP BY and HAVING relational* query *constructs and selected new constructs of OWL 2-DL can enhance the usefulness of Ontology Driven Relational Query Formulation Framework (OntoQF)*"

The first designed research question states: "*To what extent support of GROUP BY and HAVING relational query constructs can increase the usefulness of OntoQF?" (***Question 1)** This research work has exposed that the support for GROUP BY and HAVING clauses can remarkably increase the usefulness of OntoQF [1] by allowing the translation of cardinality restrictions into equivalent relational query restrictions. This support allows the users to mine some more interesting patterns e.g. to retrieve the records of those patients who have been prescribed medicines in excess in their recorded medical history. To make it possible, an extension to the translation algorithms is proposed in Chapter 4.

The second research question designed to answer the research hypothesis states: *"To what extent support of selected new constructs of OWL 2-DL can increase the usefulness of OntoQF?" (**Question 2**)* This research work has exposed that the support for selected OWL2-DL can also remarkably increase the usefulness of OntoQF by further strengthening the translation engine of OntoQF [1]. It has been noted in the literature survey (Chapter 2) that OWL2 is an extended and more expressive version of OWL. Support for OWL2 can add support for the translation of qualified cardinality restrictions on both object properties and data type properties. Support for qualified cardinality restrictions with data types are provided by exploiting the rich data types supported in OWL2. Furthermore, the rich data types of OWL2 paved the way to provide support for existential quantification restrictions and universal quantification restrictions with the datatype properties which were only allowed with the object properties in OWL1. To make it possible some extensions were proposed in Chapter4 which allows users to mine some more interesting patterns e.g. to retrieve the records of those patients who have been prescribed Antibiotics (particular type of medicine) in excess in their recorded medical history or to retrieve the records of those patients who have always normal blood pressure measurements in their recorded medical history.

The third designed research question states: "*What mappings are required between GROUP BY and HAVING relational constructs and selected OWL2-DL constructs for query translation?*" (**Question 3**) It has been observed that the ontological concept restrictions can be of varying level of complexity and they may involve multiple conditions. So the valid query formulation is not only dependent on the corresponding database schema but also on correctly translating different combinations of semantic constructs and then merging the translation results into a single query. After the proposed extension in the translation algorithms, the mapping algorithm of OntoQF [1] is extended in order to accommodate selected constructs of OWL2 and SQL GROUP BY and HAVING clauses.

The forth research question designed to answer the research hypothesis states: "*How can the correctness of new formulated queries be empirically evaluated?*" (**Question 4**) The discussions based on research questions 1-3 suggest that the Support of GROUP BY and HAVING relational query constructs and selected new constructs of OWL 2-DL can enhance the usefulness of Ontology Driven Relational Query Formulation Framework (OntoQF). However, to further test the usefulness of the proposed extension and correctness of the generated queries, a comprehensive evaluation was required. In this regard, a matrix of validation scenarios was designed to evaluate the system without missing any combination of OWL2-DL selected constructs. Some more semantic constraints were detected that result in the formation of unsatisfiable relational queries. All these unsatisable translation scenarios were described by the following rule: "*For a given Ontological statement, the OWL2-DL to relational query translation is not satisfiable if the OWL2-DL constructs include a conjunction operation between any combination of existential quantification restriction and qualified cardinality restriction on distinct ontology classes using the similar Object Property/Datatype Property*". K. Munir proposed the use of non-materialized database views to handle the unsatisfiable translations [1].

The above discussion concluded that the outcomes of the designed research questions proved that **"*Support of GROUP BY and HAVING relational query constructs and selected new constructs of OWL 2-DL can enhance the usefulness of Ontology Driven Relational Query Formulation Framework (OntoQF)*"**, with some limitations as reviewed above. The research contributions and evaluation results discussed in this section have paved the way for future work directions that are discussed in Chapter 6.

Chapter 6: Conclusion and Future Directions

The focus of this research work has been on investigating the extent to which support of group by and having relational query constructs and selected new constructs of owl 2-dl can enhance the usefulness of OntoQF. This focus has paved the way to proposing extensions to the translation algorithms and mapping algorithms of OntoQF. These proposed extensions enable users to mine more interesting patterns from the underlying relational database. Section 6.1 presents the major contributions of this research work. Section 6.2 concludes this research work by suggesting future directions.

6.1 Summary of Thesis Contributions

The major contribution of this thesis is to provide support of SQL GROUP BY and HAVING clauses and OWL 2-DL in the process of ontology driven relational query formulation. It has been concluded that, to the best of our knowledge, OntoQF [1] is the only system yet proposed that uses the assertion capabilities of OWL-DL ontologies in order to provide assistance in relational query formulation. It has also been concluded in this thesis that no existing system provides assistance in query formulation including GROUP BY and HAVING clauses, without requiring the user to have knowledge of underlying information structure. GROUP BY and HAVING clauses can play a useful role in enabling users to mine interesting patterns from the underlying relational database as they can be used to apply restrictions on the cardinality of formulated groups. The support for GROUP BY and HAVING has been provided in the thesis by proposing extensions to the ontology to relational query translation algorithms.

It has also been concluded in this thesis that OWL 2-DL is the most expressive yet decidable profile of OWL. OWL 2 has many novel features and constructs among which few are exploited to enhance the usefulness of OntoQF. Rich data types of OWL are particularly exploited in order to enable users mine more interesting patterns from the underlying relational database. This has been achieved by proposing extensions to the ontology to relational query translation algorithms. However, some unsatisfiable ontology definitions were identified in the process of comprehensive evaluation. A mechanism to handle these unsatisfiable query translations is already proposed by K. Munir [1]. Furthermore, extension to the mapping algorithm was also proposed in order to accommodate; selected constructs of OWL 2-DL and SQP GROUP BY and HAVIG clauses. In the end, it has been proved in this thesis that **"*Support of GROUP BY and HAVING relational query constructs and selected new constructs of OWL 2-DL can enhance the usefulness of Ontology Driven Relational Query Formulation Framework (OntoQF)*"**, with some limitations as reviewed above.

6.2 Future Directions

The research contributions and evaluation results have paved the way for future work directions. One of the major contributions of this research is to provide support for SQL GROUP BY and HAVING clauses. The cardinality restrictions are translated into a HAVING clause restriction using an aggregate function (SUM). Support for other aggregate functions may be added in order to enable users some more interesting patterns. Furthermore, support for other type of joins including; non-equi joins and outer-joins may also be provided. Further research is required to investigate how this support can be provided. Another major contribution of this research is to provide support for selected constructs of OWL2. Further research may be done in order to investigate how the support for some other OWL2 construct can be provided. Finally, in the evaluation process it was noted that some ontology restrictions result into formulation of unsatisfiable relational queries. Domain experts need to be automatically warned if they specify some unsatisfiable ontology restriction.

References

[1] K. Munir, M. Odeh, and R. McClatchey, “Ontology-driven relational query formulation using the semantic and assertional capabilities of OWL-DL,” *Knowledge-Based Syst.*, vol. 35, pp. 144–159, 2012.

[2] “OWL Web Ontology Language Overview.” [Online]. Available: http://www.w3.org/TR/owl-features/. [Accessed: 15-Jun-2015].

[3] “OWL 2 Web Ontology Language Document Overview (Second Edition).” [Online]. Available: http://www.w3.org/TR/owl2-overview/. [Accessed: 15-Jun-2015].

[4] “Medical Knowledge Discovery Research.” [Online]. Available: http://medicalknowledge.igs.umaryland.edu/Informations/MedicalKnowledgeDiscoveryResearchProjectHomePage.html. [Accessed: 15-Jun-2015].

[5] M. M. Zloof, “Query-by-example: the invocation and definition of tables and forms,” in *Proceedings of the 1st International Conference on Very Large Data Bases*, 1975, pp. 1–24.

[6] M. Y. M. Yen and R. W. Scamell, “Human factors experimental comparison of SQL and QBE,” *IEEE Trans. Softw. Eng.*, vol. 19, no. 4, pp. 390–409, 1993.

[7] G. Klyne and J. J. Carroll, “Resource description framework (RDF): Concepts and abstract syntax,” 2006.

[8] S. Vandamme, J. Deleu, T. Wauters, B. Vermeulen, and F. De Turck, “CROEQS: Contemporaneous role ontology-based expanded query search-Analysis of the result set size,” in *Image Analysis for Multimedia Interactive Services, 2009. WIAMIS’09. 10th Workshop on*, 2009, pp. 169–172.

[9] K. Wen, R. Li, and B. Li, “Searching concepts and association relationships based on domain ontology,” *Proc. - 9th Int. Conf. Grid Cloud Comput. GCC 2010*, pp. 432–437, 2010.

[10] H. Boumechaal and Z. Boufaida, “Formalization of natural language queries,” *INISTA 2011 - 2011 Int. Symp. Innov. Intell. Syst. Appl.*, pp. 495–499, 2011.

[11] J. Wang, J. Lu, Y. Zhang, Z. Miao, and B. Zhou, “Integrating heterogeneous data source using ontology,” *J. Softw.*, vol. 4, no. 8, pp. 843–850, 2009.

[12] G.-Q. Zhang, T. Siegler, P. Saxman, N. Sandberg, R. Mueller, N. Johnson, D. Hunscher, and S. Arabandi, “VISAGE: A Query Interface for Clinical Research.,” *AMIA Summits Transl. Sci. Proc.*, vol. 2010, no. 3, pp. 76–80, 2010.

[13] K. Munir, M. Odeh, and R. McClatchey, “Ontology-driven relational query formulation using the semantic and assertion capabilities of OWL-DL domain ontologies,” *Data Knowl. Eng.*, 2010.

[14] N. W. Paton, R. Stevens, P. Baker, C. a. Goble, S. Bechhofer, and a. Brass, “Query processing in the TAMBIS bioinformatics source integration system,” *Proceedings. Elev. Int. Conf. Sci. Stat. Database Manag.*, 1999.

[15] N. Athanasis, V. Christophides, and D. Kotzinos, “Generating On the Fly Queries for the Semantic Web : The ICS-FORTH Graphical RQL Interface (GRQL) 1,” *Iswc*, pp. 486–501, 2004.

[16] G. Gardarin, H. Kou, K. Zeitouni, X. Meng, and H. Wang, “SEWISE: An Ontology-based Web Information Search Engine.,” in *NLDB*, 2003, vol. 2003, pp. 106–119.

[17] J. Heflin and J. Hendler, “Searching the Web with SHOE,” *Artif. Intell. Web Search*, pp. 35–40, 2000.

[18] E. Mäkelä, E. Hyvönen, and S. Saarela, “Ontogator - A Semantic View-Based Search Engine Service for Web Applications,” *Semant. Web - ISWC 2006, 5th Int. Semant. Web Conf. ISWC 2006, Athens, GA, USA, Novemb. 5-9, 2006, Proc.*, vol. 4273, pp. 847–860, 2006.

[19] E. Mäkelä, E. Hyvönen, and S. Saarela, “Ontoviews–a tool for creating semantic web portals,” *Semant. Web–ISWC*, pp. 797–811, 2004.

[20] J. W. Choi and M. H. Kim, “Generating OWL ontology from relational database,” *Proc. - 2012 3rd FTRA Int. Conf. Mobile, Ubiquitous, Intell. Comput. Music 2012*, pp. 53–59, 2012.

[21] L. Yiqing, L. Lu, and L. Chen, “Automatic Learning Ontology from Relational Schema,” *IEEE Symp. Robot. Appl.*, pp. 592–595, 2012.

[22] M. Šeleng, M. Laclavík, Z. Balogh, and L. Hluchý, “RDB2Onto: Approach for creating semantic metadata from relational database data,” *Proc. 9th Int. Conf Informatics*, pp. 113–166, 2007.

[23] Q. Trinh, K. Barker, and R. Alhajj, “RDB2ONT: A tool for generating OWL ontologies from relational database systems,” *Proc. Adv. Int. Conf. Telecommun. Int. Conf. Internet Web Appl. Serv. AICT/ICIW’06*, vol. 2006, p. 170, 2006.

[24] I. Astrova, N. Korda, and A. Kalja, “Rule-Based Transformation of SQL Relational Databases to OWL Ontologies,” *Transformation*, pp. 1–16, 2007.

[25] R. Ghawi and N. Cullot, “Database-to-ontology mapping generation for semantic interoperability,” *VDBL’07 Conf. VLDB …*, p. 8, 2007.

[26] M. L. M. Li, X.-Y. D. X.-Y. Du, and S. W. S. Wang, “Learning ontology from relational database,” *2005 Int. Conf. Mach. Learn. Cybern.*, vol. 6, no. August, pp. 18–21, 2005.

[27] J. F. Sequeda, M. Arenas, and D. P. Miranker, “On directly mapping relational databases to RDF and OWL,” *Proc. 21st Int. Conf. World Wide Web - WWW ’12*, p. 649, 2012.

[28] C. Bizer, “D2R MAP - A Database to RDF Mapping Language,” *Business*, pp. 2–3, 2003.

[29] J. Barrasa, O. Corcho, and A. Gómez-Pérez, “Fund Finder : A case study of database-to-ontology mapping Case study,” *Semant. Integr. Work. 2nd Int. Semant. Web Conf. ISWC2003*, p. 6, 2003.

[30] J. Barrasa, Ó. Corcho, and A. Gómez-pérez, “R 2 O , an Extensible and Semantically Based Database- to-ontology Mapping Language,” *Proc. 2nd Work. Semant. Web Databases*, no. August, pp. 1069–1070, 2004.

[31] N. Konstantinou, D. E. Spanos, M. Chalas, E. Solidakis, and N. Mitrou, “VisAVis: An approach to an intermediate layer between ontologies and relational database contents,” *CEUR Workshop Proc.*, vol. 239, pp. 1050–1061, 2006.

[32] Z. Xu, S. Zhang, and I. Dong, “Mapping between relational database schema and OWL ontology for deep annotation,” *Proc. - 2006 IEEE/WIC/ACM Int. Conf. Web Intell. (WI 2006 Main Conf. Proceedings), WI’06*, pp. 548–552, 2007.

[33] I. Myroshnichenko and M. C. Murphy, “Mapping ER schemas to OWL ontologies,” *ICSC 2009 - 2009 IEEE Int. Conf. Semant. Comput.*, pp. 324–329, 2009.

[34] P. D. Karp, V. K. Chaudhri, and J. Thomere, “XOL: An XML-based ontology exchange language.” July, 1999.

[35] T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, and F. Yergeau, “Extensible markup language (XML),” *World Wide Web Consort. Recomm. REC-xml-19980210. http//www. w3. org/TR/1998/REC-xml-19980210*, vol. 16, 1998.

[36] “OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax (Second Edition).” [Online]. Available: http://www.w3.org/TR/owl2-syntax/. [Accessed: 15-Jun-2015].

[37] S. Lipschutz, “Schaum’s outline of theory and problems of set theory and related topics,” 1964.

[38] A. Branson, T. Hauer, R. McClatchey, D. Rogulin, and J. Shamdasani, “A data model for integrating heterogeneous medical data in the Health-e-Child Project,” *Stud. Health Technol. Inform.*, vol. 138, p. 13, 2008.

[39] “Microsoft Windows, 2009. Microsoft Windows7.” [Online]. Available: http://onthehub.com/. [Accessed: 17-Jun-2015].

[40] “NetBeans, 2009. NetBeans IDE 6.5.1.” [Online]. Available: https://netbeans.org/downloads/. [Accessed: 17-Jun-2015].

[41] “MySQL, 2007. Part of WAMP server.” [Online]. Available: http://www.wampserver.com/. [Accessed: 17-Jun-2015].