IJST EXPLORE International Journal of Science and Technology
CENTRE OF PROFESSIONAL RESEARCH PUBLICATIONS
ISSN 2049-7318

International Journal of Science and Technology Volume 3 No. 6, June, 2014

Towards a Universal Query Language for Resource Description Framework (RDF) and Topic Maps

Fakhre Alam, Nasir Rashid, Muhammad Salam, Muhammad Raees Khan
Department of Computer Science and IT, University of Malakand
K.P.K, Pakistan

ABSTRACT

Resource Description Framework (RDF) and Topic Maps are the two prominent technologies of Semantic Web developed for the purpose to make Web useful for humans and 'understandable' by machines by facilitating knowledge integration and sharing. The basic goal of RDF and Topic Maps is the same, i.e. to successfully turn the vision of semantic web into reality by attaching rich structured semantics to web contents. The problem of interoperability between RDF and Topic Maps emerged due to the difference in their fundamental architectures and their owner organizations which may lead to islands on the future web. The architecture of RDF and Topic Maps is also complex and a suitable query language is the essential pats of it for the precise and quick retrieval of required information from RDF and Topic Maps. Therefore, several types of query languages have been developed for both technologies to retrieve the required information precisely from both RDF and Topic Maps. These query languages have the capability of logic and inferencing and may be used to address the problem of interoperability. The purpose of this paper is to thoroughly investigate all aspects of the available query languages developed for RDF and Topic Maps. In this paper, a comprehensive literature study and comparison of RDF and Topic Maps query languages has been done according to different parameters. It is concluded that a universal query language capable of retrieving and filtering both RDF and Topic Maps data is possible subjected to little more efforts. This paper will not only provide a jump start in the field for new comers but will also help future researchers in selecting one query language over the other for their applications.

Keywords: *Web,Semantic Web, Query Language, Logic, RDF, Topic Maps*

1. INTRODUCTION

The current web also called World Wide Web (WWW) containstens of billions of web pages [1]. These web pages consist of text, images, audio and videos and are interlinked with each other through hypermedia links. This exponential growth of the WWW is due to its simplest design where any person can easily use and add any content to the Web. Despite all this simplicity the human presence is necessary for web contents interpretation and also it comes at the cost of losing rich semantics. In the current web, finding specific information precisely is almost impossible and the problem aggravates more as size of the web increases. Semantic Web is a response to this problem by the original creator of the web which aims to attach metadata to web resources in a format that makes web content machine-process-able. In Semantic Web, machines will integrate information, apply computation on the information, and organize them into a format to be reusable across wide array of domains. Resource Description Framework (RDF) and Topic Maps developed respectively by W3C[1] and ISO[2]are the two prominent technologies developed to fulfil the vision of Semantic Web. These technologies share the same aim of turning the Semantic Web vision into reality by making the Web useful for humans and 'understandable' by machines by facilitating knowledge integration and sharing.

The main purpose behind the development of RDF is to make infrastructure for the Semantic Web and each RDF model is composed of statements where each statement relates two Web resources by using the analogy of subject, predicate, and object. Topic Maps manipulate relevant information on the Web quickly and easily by exchanging and expressing knowledge in a meaningful way [2]. Anything on the Web can be expressed by the Topic Maps in the form of topic, association between topics and the occurrences of topics and associations.

Query Languages are used for asking questions from the databases or information systems and based on these questions answers are provided to the users [3]. When an application is developed and stored in a database storage server, the information can be retrieved by using a query language. Query language is an essential and integral part of a database or information system, as it helps users to manipulate the required information from Database Management System (DBMS) [4].

[1]http://www.w3.org/

[2]http://www.iso.org/iso/home.html

The independence which is necessary between the database and application is also maintained by the query language. Today, databases consist of large volumes of data consisting of text and multimedia contents. To handle such type of system, there is a need of effective query languages to provide easy and user friendly interface not only for experts but also to naïve users.
There are two types of query languages: SQL-based and Logic-based as shown in Figure 1. The Query Languages, based on SQL construct, generally use SELECT statements to manipulate data from the backend. The data is mostly stored in the form of relational databases. The answers which are generated from the execution of these statements are transferred to the users in different forms based on the applications the users use. Many types of open and commercial SQL-based languages are available, and these languages can also transform the generated results to XML and RDF syntax e.g. SQL/XML. Since SQL based databases store data in tabular (relational) form, now it becomes possible due to these advanced features of SQL to create XML and RDF elements from the relational data.Currently, SQL-based languages are widely used on the Web because of their support to distributed databases and the availability of various open source SQL database solutions. The other reason, that SQL based languages became the foundation of business logic, is the support and investment by the top vendors such as Microsoft and Oracle. A number of well established database applications on the Web can work with SQL based query languages, such as MySQL, SQLite[3], Postgre[4] and Firebird[5].

Languages based on logic have a combined features of expressive power of high level logic and the non-navigational feature of relational query languages [5]. Logic based query languages are well suited for the manipulation of information from the huge amount of databases on the Web. These languages not only directly answer the users' keywords like SQL but also predict an answer based on the computation of the consequences of axioms and rules. One can get all the benefits of database languages in logic based languages due to which it is now the necessity of the current web standard. Since Semantic Web metadata are stored in the form of RDF and Topic Maps, logic based languages are suitable to deal with this situation due to their rule based deductive ability and their unification based pattern matching capability.

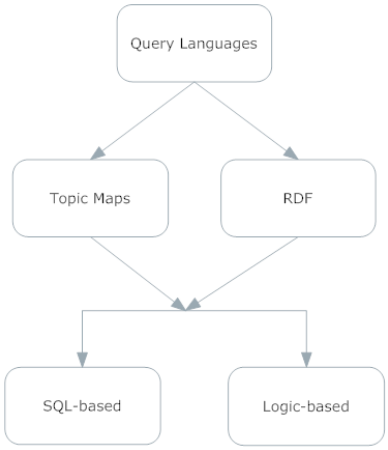

Figure 1. Categorization of RDF and Topic Maps Query Languages

The complexity in RDF and Topic Maps technologies are increasing now-a-days due to the addition of millions of topics and resources with complex associations and wide occurrences. Several types of query languages using logic and inferencing capabilities have been investigated by the research community enabling Semantic Web users to easily retrieve the required information from the underlying complex RDF and Topic Maps architectures.As the manipulation of RDF and Topic Maps information is not a simple job, it is necessary that there should be useful and universal query language to preciselyretrieve RDF and Topic Maps data.

While being overwhelmed with a number of query languages, it is rather cumbersome for researchers and users to select an appropriate query language applicable to both RDF and Topic Maps. This research paper is aimed to provide a comprehensive overview and analysis of the available RDF and Topic Maps query languages, covering all of their possible aspects, and pros and cons. It also attempts to find out how much RDF and Topic Maps query languages can be used together. In this paper, a comprehensive literature study has been done and comparison of RDF and Topic Maps query languages are drawn. Main contributions of the paper are:

- To provides a concise overview of the available query languages for RDF and Topic Maps
- The work is almost unique in its integrity and opens new area of research.
- The classification and comparison of RDF and Topic Maps query languages have never done before in a single document. Therefore, it will

[3]http://www.sqlite.org/

[4]http://www.postgresql.org/

[5]http://www.firebirdsql.org/

provide a compact platform for the users and researchers to grab almost all of the relevant possible information about the topic in a single document.

- To organize and classify the available literature about the topic in an attractive manner to catch and boost interest of the new researchers in the area and take them into new avenues of research.
- To briefly explain and analyze the available query languages which are currently available in various domains and with their success stories and common reasons of failure.

Topic Maps data can be queried with Topic Maps Query Languages, same as that of relational databases are queried with SQL. Topic Maps query languages provide much quicker retrieval of information based on their intelligence and inferencing capability. Based on these inference rules, topics of specific types and their associations in different contexts can be retrieved much easily. Several types of query languages were developed for the effective retrieval and filtering of Topic Maps based information. Among them TMQL, tolog, TMRQL, AsTMA and Toma were more popular. The types of Topic Maps query languages are shown in Figure 2 and the brief introduction of each of these is given in the sub-sections below.

2. TOPIC MAPS QUERY LANGUAGES

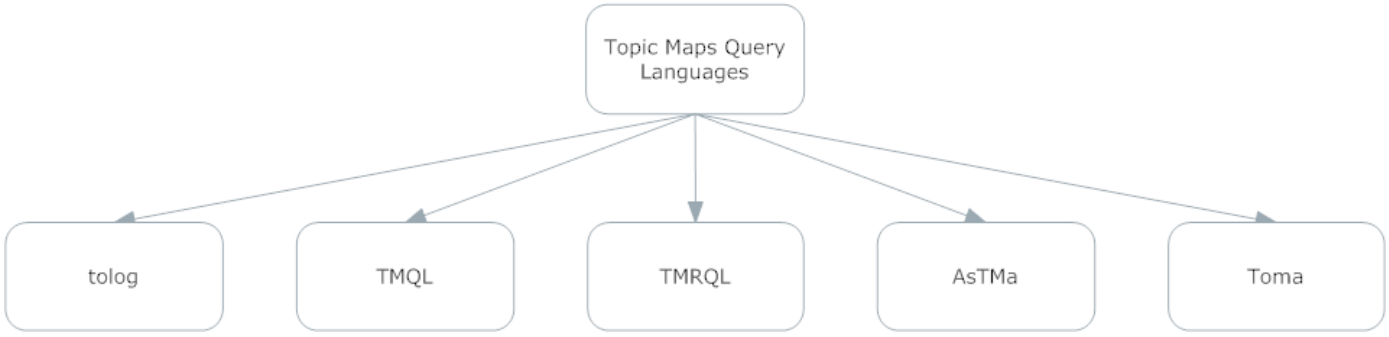

Figure 2 Types of Topic Maps Query

2.1. Topic Map Query Language (TMQL)

Topic Map Query Language (TMQL) is standardized by the ISO no 18048[6]. It has same as that of SQL and XMLconstructs which are simple and familiar to most of the developers [6]. However, SQL was developed for well-defined structure in RDBMS, while TMQL is applicable to any kind of situation, such as huge quantity, continuously varying information and for semi-structured environment. Using TMQL, information from Topic Maps can be retrieved easily and quickly because it provides the facility for the users to simplify the application development up to a great extent.

TMQL gives many facilities such as XPath which detects and then compares certain patterns in the data, navigates between them to compute and return the final results to the application.

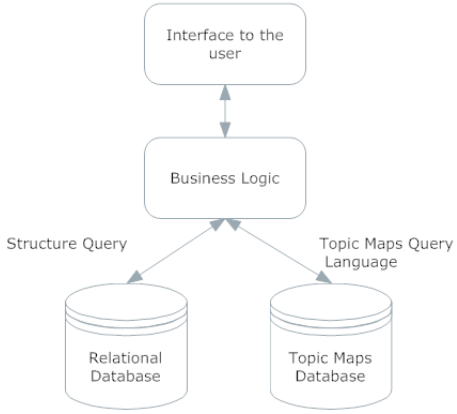

Figure 3. TMQL in business logic [6]

In Figure 3, the user will connect to the business logic through web browser, which may be a company servelet. This servelet will be further connected to the RDBMS and Topic Maps repositories through SQL and TMQL respectively. Through SQL, well defined structured data such as company addresses and information stored in RDBMS will be accessed which is semi-structured; and metadata such as company worker's skill

[6]http://www.isotopicmaps.org/tmql/

information stored in Topic Maps repository will be retrieved through Topic Maps Query Languages (TMQL).
Figure 4 shows the architecture of TMQL system context. In this architecture, the TMQL interface provides a bridge between applications and Topic Maps databases.

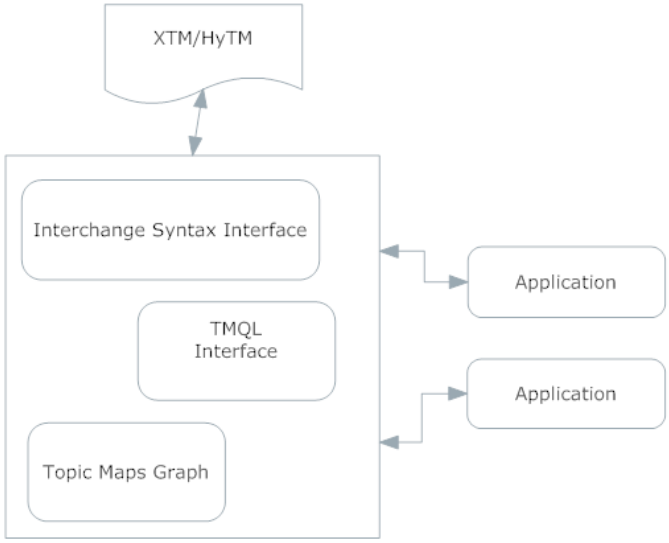

Figure 4.Architecture and TMQL System Context [6].

2.2. tolog

tolog is a Topic Maps query language developed by Lars Marius Garshol, working at Ontopia. This query language is similar to SQL but also with logic and inference capability [7]. Each type of information related to Topic Maps such as topic, topic with multiple names and scope, subject, associations, scope and role in different types of situations can be queried using tolog. Since tolog is based on logic, there will be more responsibility for user to ask question and then tolog will use all its claim and general statements to deduce a valid statement. The true and accurate statement can be deduced by tolog with the help of predicate. This predicate establishes a relationship between sets of values which are further stored in the form of a table. When the query runs against database, all the sets of values return that are similar to the query from the data table.

Using opera web browser, one can run several types of queries. For example if a user wants to retrieve all possible topics and each topic type, he/she can get this by running the query: *'instance-of($TOPIC, $TYPE)?*. Executing this query will result hundreds of lines result, a sample of which is shown in Table 1. In the table some topics such as Khyber appears twice because it is the instances of two types.

Table 1. All Topics and types

TOPICS	TYPES
UOP	University
UOM	University
Khyber	TV Company
Khyber	Place
NTS	Organization
Pakistan	Country

Similarly, if we run the query: written-by($ Ghani, Book)?. All the topics and associations related to Ghani (author) and the book will be retrieved from Topic Maps. If we want to retrieve only university type, the query will be: *instance-of($TOPIC, University)?*.This will give the results shown in Table 2.

Table2. All Universities

TOPIC
University of Peshawar
University of Malakand
University of Swat

In the same way, by replacing the topic with a particular reference type and putting type in the second variable, we can find all types of which a topic is an instance, '*instance-of(Khyber, $TYPE)?'*. The result generated by tolog, by running this query, is shown in Table 3.

Table 3. All types of which a topic is an instance

TYPE
TV Company
Place
Country

In Table 3, the word "Country" has come along with TV Company and Place because Country is defined as the super type of Khyber in Topic Maps. This shows that any instance of place is also an instance of country. By making use of information, tolog gets the result that Khyber is the instance of three types: TV Company, Place and Country.

Despite its logic and inference capabilities, tolog has some shortcomings due to which it cannot manipulate each resource of Topic Maps [7]. The limitations of tolog are its implementation of query only on associations and as a query result, it only displays topics. This query language greatly depends on unique IDs for Topics and these IDs may not be unique in case when Topic Maps are retrieved from large amount of XML documents. In tolog, a rule file is necessary to write before querying a Topic Maps and also there is no mechanism in tolog to modify Topic Maps.

2.3. Topic Maps Relational Query Language (TMRQL)

The relational model, which was already developed, provides the base for Topic Maps Query Languages. Based on relational concepts, a new query language called TMRQL was developed by Kal Ahmed and Graham Moore in Networked Planet[7]. TMRQL provides easy accessibility and usefulness for the users to query Topic Maps. A set of relational views can be described by the TMRQL and because of these relational views, an abstract relational model for the Topic Maps data model can be created [8]. This will be the point of attraction to work in this type of query language for the developer as compared to the language based on totally new concepts.

Advantages of TMRQL include, the familiarity of the developer with an already developed language can result sound theoretical base, output flexibility, avoid duplicate previous error and enhance adoption and use. On the other hand, TMRQL is difficult to be implemented due to its complex nature and inconsistencies in SQL support [8]. TMDM is generally implemented on a system of relational database technology, therefore, it will be difficult to implement TMRQL on such systems.

2.4. Asymptotic Topic Map (AsTMa)

AsTMa[9] is a family of languages used to implement constraints and query Topic Maps. AsTMa query language was developed by Dr. Robert Barta[8]at Bond University. This query language is similar to the style of XQuery with functional capabilities and has recommendation for several types of path languages. In AsTMa, topic and associations can be matched by an open and close rule construct using different types of operators and then both are combined to create a tree structure which can be re-structured easily.
Although AsTMa query language is rather complex but its basic mode of operation is easy and simple [9]. AsTMa query statements work according to the choice of user much like that of WHERE statement in SQL which selects the desired rows. AsTMa is more flexible because it can categorize the users' interest and return the precise answers to them. Although AsTMa is one of the fully implemented query language but the lazy evaluation[9] which is one of the necessary components in a programming language is not introduced yet. In this query language, function objects are not passed around and there is no description for incomplete evaluated functions.

2.5. Toma

Toma query language was designed for Topic Maps after tolog and AsTMA. The syntax of Toma is similar to SQL but provides more powerful syntax features like path expression of Toma can easily query data and ontology at the same time [10]. Topic, associations and occurrences can be accessed, modified, and converted more easily with the help of Toma. The statements such as SELECT, INSERT, UPDATE and DELET used by SQL can also be used by Toma along with MERGE and EXPORT statements. Toma can also implement constraints on Topic Maps and manipulate its data, therefore it is also called Topic Maps Constraints Language (TMCL), and Topic Maps Manipulation Language (TMML).

2.6. Tabulated Representation and Analysis of Topic Maps Query Languages

Table 4 shows the analysis of Topic Maps Query Languages based on the discussions in sections 2.1, 2.2, 2.3, 2.4 and 2.5 above. In Table 4, Topic Maps query languages are analyzed based on the parameters: organization/ projects under which each query language developed, its implementation, results and accuracy of each language, how much scalable each query language is and the scalability/extensibility of each query language. It is obvious from the table that almost every type of query constructs is similar to SQL along with logic and inference capabilities.

3. RDF Query Languages

RDF data is stored in the form of a graph; consisting of RDF triple set which contains URIs and blank nodes. RDF data is organized in a semi-structured form and a complex association is established between triple. To retrieve this data effectively, RDF query languages were developed to access and manipulate data stored in RDF model and apply operations on it [11]. RDF query languages have the capability to know about RDF triple relationship, to deal and interact with semi-structured data and the ability to operate on the data built on cross-language and cross-platform. When there is a huge amount of data and high operation overhead, RDF query languages are useful in such a situation to access data in shorter period of time and accuracy. These languages also have some more features such as aggregate functions for extracting statistics, comparing different values and their data types, closure property for extracting values outside its scope, generalized path expressions for debugging and semantic capabilities.

[7]http://www.networkedplanet.com/

[8]http://www.topicmapslab.de/people/Robert_Barta

[9]http://en.wikipedia.org/wiki/Lazy_evaluation

Table 4. Analysis of Topic Maps Query Languages

	Query Language	Organization/ Project	Implementation	Query result accuracy	Scalability/ Extension	Remarks
Topic Maps Query Languages	**TMQL**	ISO/IEC JTC1 SC34 WG3	Implemented	High	Yes	• Applicable to huge quantity, continuously varying information and for semi structure environment • Retrieves TM information easily and quickly • Can detect and then compare certain patterns in the data • Too hard to understand and implement
	Tolog	Ontopia	Implemented	Medium	Yes	• Logic and inference capability • Can deduce a valid statement • Cannot manipulate every resource of Topic Maps
	TMRQL	Networked Planet	Not	High	---	• Provides easy accessibility • Relational views • Familiarity for developer • Output flexibility • Avoids repeating previous error • Difficult to implement due to its complex nature and inconsistencies in SQL support
	AsTMa	Topic Maps Lab	Implemented	Low	Yes	• Easy and simple to operate • Works according to the choice of user • Flexible • Lazy evaluation is not define yet
	Toma	Ontopia (Space Applications)	Not	High	Yes	• Provides more powerful syntax features • Can also implement constraints and manipulate TM • Uses SELECT, INSERT, UPDATE and DELET • Currently, Toma queries can only be run using a command-line client

Several languages have been developed for the effective retrieval and filtering of RDF metadata information. These languages are further categorized into two approaches: the one in which RDF data can be viewed as relational or XML database, is called SQL/XQL[10], and the other technique which uses knowledge representation and reasoning techniques to view metadata on the Web is called knowledge based [12]. SPRAQL, RDQL, RQL, SeRQL, and XsRQLare the most popular query languages for RDF model as shown in Figure 5. The brief introduction of each of these is given in the sub-sections below:

[10]http://openacs.org/xowiki/SQL_-_XQL

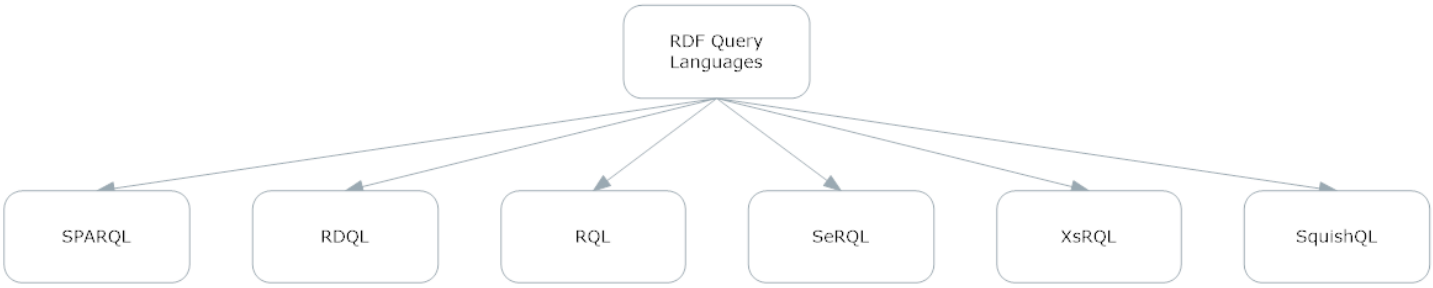

Figure 5 Types of RDF Query Languages

3.1. SPARQL

SPARQL is a query language standardized by W3C, used to retrieve data from RDF graph stored in triple format [13]. The root of this query language leads to SquishQL and RDQL (RDF query languages). SQL is the main motivating force behind it. The data stored in triple format is based on certain graph patterns. SPARQL query is run against these graph patterns and based on matching triple patterns with query; the result is displayed to the users. Information are retrieved in URIs, blank nodes, plain and typed literals form by SPARQL. One of the reason of its wide spread use is due to its ability to transfer SPARQL query to other query format such as RDBMS query language e.g. SQL or XML query language e.g. XQuery.
Using HTTP protocol, the queries of SPARQL are sent from the client to the service called end point [14]. The interface between client and end point is based on system friendly protocol, which creates difficulty for the users while interpreting it. User asks queries and answer of these queries return in a meaningful way to the user. An interface is required in SPARQL.

Most of the commands used in SPARQL are similar to the SQL fundamental commands such as SELECT, WHERE, FROM and ORDER BY[15]. These basic commands are used to find the selected data and then return it, find matches in RDF graph patterns and arrange information according to the specific field. SPARQL query starts from prefix keyword and each variable is followed by the symbol '?'. Although SPARQL query language has several features which make it one of the most widely used RDF query language, but there are some limitations which should be properly addressed [16]. If there is a statement in RDF triple containing negative statement, then it is difficult for SPARQL to handle it. SPARQL has also no support for important property of matching basic graph patterns called path expression. In other words, path expression is a variable length path and assigning properties to predicate in RDF triple.

SPARQL syntax and queries are based on Turtle syntax, and with the help of these queries users can clearly and explicitly interact with RDF graph through SPARQL. Some of the basic queries of SPARQL are described below:

- If we have an RDF database named 'uop.rdf' and the user wants to find URL of a web page created by a person named Ali, then it would be queried with SPARQL as follows:

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?url
FROM     <uop.rdf>
WHERE
     {
        ?web-creator  foaf:name  "Ali".
        ?web-creator  foaf:website  ?url .
     }
```

The result of the above query is shown in Table 5.

Table 5. To find URL of a web page created by a person named Ali from uop.rdf database

url
<http//:www.szic-upesh.edu.pk>

As SPARQL is based on turtle syntax, therefore, at the beginning we write PREFIX for the FOAF name spaces. Each variable is preceded by '?' or '$' symbols, the SELECT statement will return a variable named url. The name of the RDF graph in the third line is optional in the FROM statement and it shows the name of the file stored locally. The graph pattern in the form of triple is included in the WHERE block, represented in turtle syntax.

- To get name of the book's author along with books name from RDF books database the query will be:

```
PREFIXfoaf:
<http://xmlns.com/foaf/0.1/>
SELECT ?author   ?book
```

```
WHERE
  {
   ?x foaf:auther  ?author.
    ?x  foaf:book   ?book
   }
```

By running the above query against RDF graph patterns, the result in Table 6 will be displayed:

Figure 6. Name of the book's author along with books name from RDF books database

Author	Book
"Ghani Khan"	"The Pathan"
"The Land of War Elephants"	"Mathew Wilson"

- If SPARQL query is to retrieve that book which has a price less than Rs.20 along with title, then we add the FILTER keyword which apply the value constraints, as shown below:

```
PREFIX dc:
<http://purl.org/dc/elements/1.1/>
       PREFIX           ns:          <
http://www.apnaorg.com /ns#>
SELECT  ?title  ?price
      WHERE { ?x  ns:price ?price.
FILTER ?price < 20.
       ?x dc:title ?title . }
```

The result in the Table 7 will be generated by the above query

Table 7. Book which has a price less than Rs.20 along with title

Title	Price
Land of War Elephants	15

- To retrieve records from RDF databases, according to ascending order, the following query will be executed:

```
PREFIX dc:
<http://purl.org/dc/elements/1.1/>
          PREFIX ns: <
http://www.apnaorg.com /ns#>
SELECT  ?title  ?price ?pub-date
      WHERE { ?x  ns:price ?price.
               ?x dc:title ?title .
?x ns:pub-date ?pub-date
}
ORDER BY ASC[?pub-date]
```

The above query will return all information including books titles, prices and publication years by ascending it according to the publication date as shown in Table 8.

Table 8. Records from RDF databases in ascending order according to publication date

Title	Price	Pub-date
The Pathan	20	1985
Land of War Elephants	15	1990

3.2. RDF Data Query Language (RDQL)

RDQL is a query language for RDF data model developed by Hewlett Packard, and standardized by W3C in early 2004 [17]. This query language is supported by several RDF systems, mostly implemented in Jena framework. The basic approach of RDQL was taken from an earlier language called SQUISH. The language was data-oriented and declarative, in which only the information was retrieved and stored in the RDF database. The inferencing mechanism in RDQL is very less as compared to other RDF query languages, but it can resolve complex queries very easily and concisely. Its basic statements and keywords are mostly similar to SQL such as SELECT and FROM, and RDF data is viewed by this query language as triple format. RDQL is designed for only limited functionality and it is very simple to use. Some of the shortcomings of RDQL include; its performance issues related to its storage engine, restriction of OR operation which is used between RDF triple and the huge amount of data retrieved from data storage due to its matching of every triple with the first one [18].

3.3. Relational Query Language (RQL)

RQL is another declarative query language for accessing both RDF data and schema information [19]. This query language was developed by Greg Karvounarakis working inICS-FORTH[11] research project. To implement this query language efficiently and get the desired results quickly, the RDF data in the form of triple should be mapped and then it will be converted into relational databases. This type of mapping is good in situations where data is only needed to be stored in simple relational form from specific RDF triple format. In situation when the user data consists of complex vocabularies, then it can be transformed into the RDF/OWL to be efficiently

[11]http://www.ics.forth.gr/

accessed by the RQL. This query language hasbecome one of the most powerful tool as compared to other languages e.g. SPARQL, due to its unique features of generalized path expression and support for both data and schema information. RQL can also support XML schema data types, arithmetic operations, aggregate functions, namespaces and recursive traversal of class and property hierarchy [20].

3.4. Sesame RDF Query Language (SeRQL)

SeRQL is RDF query language, derived from already existing query languages of RDF such as SPARQL, RDQL and RQL [21]. The general syntax of this query language is the same as RQL, but unlike that of RQL, its formal interpretation is based directly on the RDF model theory due to which this query language can easily parse compared to RQL. SeRQL is implemented on Sesame system and has many features such as string matching with case sensitivity, boolean constraints and performed operations on set theories, nested queries, default name spaces, blank node identification and support of generalize path expression. In case of the provision of recursive built-in function, this query language is not so safe as compared to other query languages and has no support for aggregation and nesting.

3.5. XQuery-style Query Language for RDF (XsRQL)

XsRQL is another RDF query language used for RDF graph. The style and syntax of XsRQL is similar to the XML query language, Xquery, with more simplicity and readability [17]. The two reasons behind the popularity of this query language are the use of already functional and widespread used syntaxes of W3C XML Query working group, and the capability to skip built-in and complex statements that are unique for XML. XQuery provides simple and stylish interface to the users. Due to this property users can choose query style of their own choice.

3.6. SQL like Query Language (SquishQL)

The basic structure of SquishQL is the same as SQL which can efficiently and consistently navigate RDF graph [17]. The future of this query language is not so bright from standardization point of view, because there are other standardized query languages in the market and most of them are based on Java powerful features. However, due to its single query mechanism used for sub-graph matching and simple filtering mechanism, this query language is suitable for beginners. SquishQL query language is implemented with the help of Jena application and users of SquishQL can access RDF data quickly and easily. Reifications and containers are one of the most important features of RDF graph but there is a lack of support for such features in SquishQL.

3.7. Tabulated Representation and Analysis of RDF Query Languages

Table 5 shows the analysis of RDF Query Languages based on the discussions in sections 3.1, 3.2, 3.3, 3.4, 3.5 and 3.6. In Table 5, RDF query languages are analyzed based on the parameters: organization/ projects under which each query language developed, its implementation, results and accuracy of each language, how much scalable each query language is and the scalability/extensibility of each query language. One can easily analyzed from the table that almost each type of query constructs is similar to SQL along with logic and inference capabilities and SPARQL query language is one of the most popular among them.

Table 5. Analysis of RDF Query Languages

RDF Query Languages	Query Language	Organization/ Project	Implementation	Query Result Accuracy	Scalability/ Extension	Remarks
	SPARQL	W3C RDF Data Access Working Group (DAWG)	Yes	Very High	Yes	• Transfers SPARQL query to another query format • Commands used in SPARQL are similar to SQL • No support for negative statements and path expression
	RDQL	Hewlett Packard	Yes	High	Yes	• Data-oriented and declarative • Performance issues, restriction of OR operation

	RQL	ICS-FORTH Research Project	Yes	Medium	Yes	• Generalized path expression • Support for both data and schema • Arithmetic operations • Aggregate functions • Name spaces • Recursive traversal • Enforce constraints on domain and range of property
	SeRQL	Sesame	Yes	Low	Yes	• Can easily parse • Boolean constraints • Operations on set theories, nested queries • Default name spaces • Blank node identification • Support of generalized path expression • Not safe
	XsRQL	RDF Data Access Working Group	No	Medium	No	• Uses existing functional syntaxes of XML Query • Simple and stylish interface • Cannot support closure operation, descendant like operators
	SquishQL	----	No	Low	No	• Single query mechanism used for sub graph matching and simple filtering mechanism • No support for reification and containers

4. QUERY LANGUAGE SUPPORTED BY BOTH TOPIC MAPS AND RDF

RDF and Topic Maps are the two alternative technologies used for Semantic Web, and each has its own tools for storing, navigating, editing and browsing data. Both the technologies have their own query languages, but the query languages for RDF are excessive in number and popular than Topic Maps Query languages. Due to this reason, some interoperable systems were developed recently by Semantic Web community to extract Topic Maps data with RDF/RDFS query languages. For these techniques to work efficiently; first the Topic Maps data should be mapped to RDF schema vocabulary, because in RDF, each resource has been identified by a single URI; while in Topic Maps, each topic has multiple identifiers.

SPARQL is one of the Query language and protocol through which we can query and manipulate Topic Map data. It was originally developed as a query language for RDF, however, further enhanced to view and manipulate Topic Maps data in terms of RDF schema [22]. The triple mapping technique called TMSPARQL takes SPARQL query and translates it into a set of matches against a Topic Maps data store.

To represent Topic Maps data accurately as RDF schema, a complex type of architecture was developed called TM-viewer architecture, shown in Figure 6. This architecture maps the Topic Maps data to RDF schema and then analyze it by SPARQL query processor [23]. The architecture in the Figure 6 consists of many modules, such as Topic Maps database in the form of XTM, TM Importer, Topic Map conceptual schema, RDF view generator, mapping rules, query processor and parser. Topic Map data is imported from Topic Maps file by the TM importer and this data is further populated to Topic Maps conceptual schema. Topic Maps conceptual schema is actually a functional model through which any Topic Maps database can be mapped automatically to RDFs view. After the automatic mapping is done by the programmer, it can issue SPARQL queries to the TM view module, through which Topic Maps contents in XTM format can easily be extracted. TM-viewer architecture is shown in the Figure 6 below.

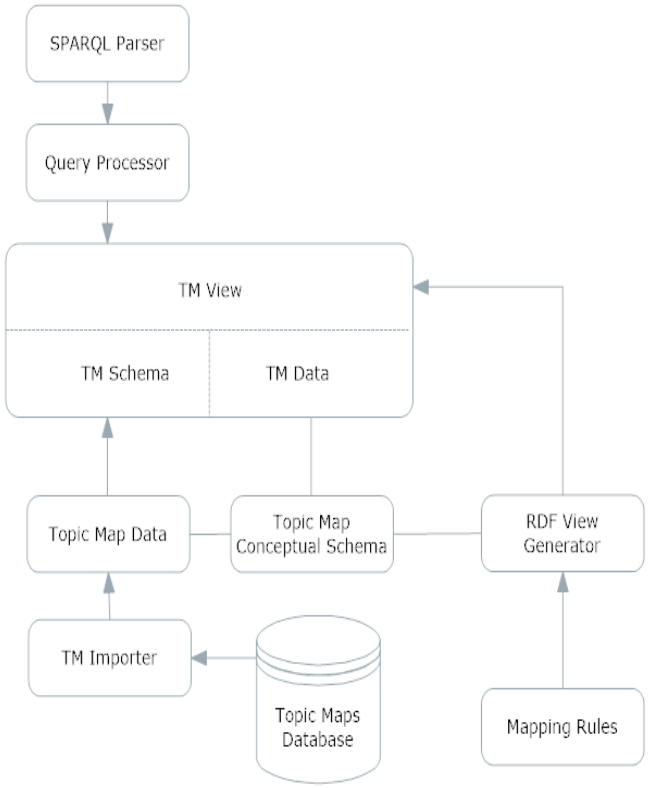

Figure 6.TM-viewer architecture [23]

5. Conclusion

In this paper, the available query languages for RDF and Topic Maps are highlighted. Several types of query languages using logic and inferencing capabilities have been investigated by the research community enabling Semantic Web users to easily retrieve required information from the underlying complex RDF and Topic Maps architectures. Semantic Web query languages make use of SQL-based and logic-based constructs and, therefore, can manipulate and present a large amount of information available on the Web in an organized format. Topic Maps query languages such as TMQL, tolog, TMRQL, AsTMA, and Toma can retrieve topics, associations between topics, and their occurrences effectively. On the other hand, RDF community has investigated a list of query languages including RDQL, RQL, SeRQL, XsRQL and the latest standard SPARQL. These query languages have the capability to effectively manipulate RDF metadata information available on different platforms. However, RDF query languages, especially SPARQL, have obtained high level of popularity due its resemblance with the other real world query languages. On the other hand Topic Maps query languages have gained less popularity as compared to RDF query languages. Therefore, a common set of protocols or standards needed to be investigated for applying RDF query language, SPARQL, equally to Topic Maps with the same ease, reliability, and higher performance.

REFERENCES

[1] World Wide Web. 24 December 2012; Available from: http://en.wikipedia.org/wiki/World_Wide_Web

[2] Topic Maps. 2 June 2012; Available from: http://en.wikipedia.org/wiki/Topic_Maps.

[3] Maier, D., THE THEORY OF RELATIONAL DATABASES. 1982, COMPUTER SCIENCE PRESS.

[4] Z, L.J., O.M. T, and S. D, Query Languages in Multimedia Database Systems. 1995, Technical Report, The University of Alberta Edmonton.

[5]Krishnamurthy, R. and C. Zaniolo, Optimization in a Logic Based Language for Knowledge and Data Intensive Applications, in International Conference on Extending Database Technology Venice, Italy, March 14–18, 1988 Proceedings, J.W. Schmidt, S. Ceri, and M. Missikoff, Editors.

[6] ISO, TMQL -Topic Map Query Language, A New ISO Standards Initiative. . 2001, ISO JTC1 SC34.

[7] Garshol, L.M., tolog - a topic maps query language, In Proceedings of the First International Workshop on Topic Maps Research and Applications (TMRA) LNCS 3873. 2005., Ontopia AS, Oslo, Norway,.p. pages 183-196.

[8] Moore, G. and K. Ahmed, Topic Map Relational Query Language - TMRQL,, in Lecture Notes in Computer Science, . 2005, : Springer Berlin / Heidelberg, ISSN 0302-9743.

[9]Barta, R. AsTMa= Language Definition. 2003; Available from: http://james.bond.edu.au/courses/inft12235/092/astma-expl.html.

[10]Pinchuk, R., et al., Toma-TMQL, TMCL, TMML. , in Lecture Notes in Computer Science (Vol. 4438/2007, pp. 107-129).Springer. . 2007

[11] Miller, L., A. Seaborne, and A. Reggiori, Three Implementations of SquishQL, a Simple RDF Query Language,, in Proceedings of the First International Semantic Web Conference on The Semantic Web,. 2002. p. p.423-435.

[12] Karvounrakis, G. RDF Query Languages: A state-of-the-art. Available from: http://139.91.183.30:9090/RDF/publications/state.html.

[13]Prud'hommeaux, E. and A. Seaborne. SPARQL Query Language for RDF. 15 January 2008; Available from: http://www.w3.org/TR/rdf-sparql-query/.

[14] Beginner's Guide to RDF. 17 February 2012 1 Dec 2012; Available from: http://code.google.com/p/tdwg-rdf/wiki/Beginners6SPARQL#6.1._What_is_SPARQL?

[15] Tutorial 5: Querying Semantic Data. 2009; Available from: http://www.linkeddatatools.com/querying-semantic-data.

[16] Ho, R. The Limitations of SPARQL. AUGUST 28, 2010 16-11-2012]; Available from: http://horicky.blogspot.com/2010/08/limitations-of-sparql.html.

[17]Bhandari, A., Comparative Analysis of RDF Query Languages, in COMPUTER SCIENCE AND ENGINEERING DEPARTMENT. June 2011, THAPAR UNIVERSITY: PATIALA - 147004. p. 66.

[18]Danils, J. and J. Stark. SCAM.2003; Available from: http://scam.sourceforge.net/html/search.html.

[19]Laborda, C.P.e.d. and S. Conrad, Querying Relational Databases with RDQL. In Berliner XML Tage, 2005.: p. 161-172.

[20] Karvounarakis, G. and V. Christophides. The RDF Query Language (RQL). Oct 9, 2008 19-11-2012]; Available from: http://139.91.183.30:9090/RDF/RQL/index.html.

[21]Haase, P., et al., A Comparison of RDF Query Languages. ISWC 2004.LNCS, 2004. 3298: p. 502-517.

[22] Ahmed, K., Making Topic Maps SPARQL, in TMRA – the international conferences on Topic Maps Research and Applications 2009: Leipzig, Germany.

[23]S, S. and R. T, SPARQL queries to RDFS views of Topic MapsSource.International Journal of Metadata, Semantics and Ontologies, 2010. 5(N° 1): p. 1-16.