Constructing Semantic Web Form from Unstructured Web Page

Amira AbdEl-atey
Faculty of computers and Information,
Menofia University.

Sherif El-etriby
Faculty of computers and Information,
Menofia University.

Arabi kishk
Faculty of computers and Information,
Menofia University.

ABSTRACT
Semantic web is a kind of webs that is able to describe things to be understood by computers. Automatically answering any query without human interactions is one of the key challenges in computer science area. Semantics can help in answering such queries. Consequently, extracting information from unstructured documents and transforming them into semantic web form is an important trend. Semantic web mining is a combination of two trends; semantic web and web mining. Our extracting and structuring system clarify the meaning of the web mining. The obtained data converted to the semantic web format. And so, the semantic web mining trend was illustrated. This paper concentrates on extracting data from the web page tables. Data on the Web in the HTML tables are mostly structured. However; we usually do not know the structure in advance. Thus, data of interest cannot be directly queried. Data extraction and structuring system is proposed to put data extracted into the semantic web form. After putting extracted data in the semantic web format, it can be queried using semantic web query language. Experimental results show that the data of interest can be located and build its new structure using semantic web.

Keywords
semantic web; information extraction; information structuring; natural language processing; wrapper generation; semantic web mining; extracting and structuring data.

1. INTRODUCTION
The Semantic Web is a mesh of information linked up to be easily processable by machines as a globally linked data base [1]. "If HTML and the Web made all the online documents look like one huge book, RDF, schema, and inference languages will make all the data in the world look like one huge database" Tim Berners-Lee [5].The Semantic Web is a kind of a web that is able to describe things to be understood by computers. Statements are built with syntax rules. The syntax of a language defines the rules of building the language statements. This is what the Semantic Web is all about describing things in such a way that computer applications can understand. The Semantic Web is not about links between web pages [7]. To know the main purpose of semantic web, a search example on the current web will be driven. Humans are capable of using the current Web to carryout tasks such as searching for a low price car. However, machines cannot accomplish all of these tasks without human directions. Web pages are designed to be read by people, not machines. The semantic web is a vision of information that can be interpreted by machines. Machines can perform more of the work involved in finding, combining, and acting upon information on the web [1]. Semantic web has more standard and non standard technologies .The main semantic web technology for building semantic web is Resource Description Framework (RDF) [4]. RDF is capable of describing information and resources on the web. The main semantic web technology for querying semantic web is SPARQL which stands for SPARQL Protocol and RDF Query Language.

The research reported here relates to recent efforts in several areas including semantic web extraction of data, Web data modeling, wrapper generation, natural-language processing, semi-structured data, and Web queries. Semantic web attempts to build systems that find and link relevant information in natural language documents while ignoring extraneous, irrelevant information [12]. Also, query extracted data quickly and accurately. The most common approach to information extraction from the Web is through wrappers [2], which parse source documents to provide a layer to map source data into a structured or semi-structured form. If the mapped form is fully structured, standard query languages such as SQL can be used to query the extracted information. Otherwise, special semi-structured query languages may be used.

Several works have been done on the web page. The previous work concentrates on a specified domains and not a general view [8], [9], [10]. Furthermore, some systems are being developed on Wikipedia. Wikipedia extraction framework concentrates on infobox template [2].

The work in this paper is focused on the HTML tables. We consider these HTML tables to be our sources. The target of the proposed work is to extract data, put them into the semantic web form and querying these data. A system for converting current web page to semantic web form is proposed. The proposed system doesn't works on a specified domain. It structures data extracted from web page tables. First, the data from web page tables is extracted using pythonic methods. Then apply structuring framework. This approach doesn't guarantee to work well for all the unstructured HTML table documents. The approach works well for unstructured HTML table documents that contain one table on the HTML page.

The target is to extract data from the web tables and form the semantic web structure and then query these data. The extracted data can be converted to the RDF format using the D2R server [6]. We test the system for the HTML page that contains any number of tables. Using the proposed system Data extracted from each table, then stored in a data structure in the triple format. Furthermore, the data converted to the RDF format. After that, The RDF formatted data queried by the SPARQL. The data in the database converted to the RDF triple format with the D2R server. The D2R server is used to convert data to semantic web format [6].

From the study, we can see that we can draw the vision of the semantic web. Generally, we convert the web page table to the semantic web form. The proposed system considered a general system and work on any web page table. First we could extract the important data and store it in a data structure. Then, store the extracted data in the database, subsequently we use the D2R server to build the RDF triples from stored database, therefore we can query these RDF triples. As a result of the system, we can build semantic web application and query this application as a database. And making the web like a data base is the vision of the new form of web which is called semantic web. We present the details for our paper as follows. We give an overview of semantic web in section II. The paper provides some details about semantic web technologies, RDF, SPQRQL query language and semantic web mining. In section III the paper demonstrates the proposed extraction framework, how to build RDF and how to query this RDF data. In section IV the proposed framework problems is discussed. In section V the paper describes our implementation status. Finally, Conclusion and future directions discussed in section VI.

2. SEMANTIC WEB

The Semantic web is a new vision of the current web; it has more standard and non-standard technologies. The Semantic Web describes the relationships between things (E.g., A is a part of B and Y is a member of Z) and the properties of things (size, weight, age, and price). Semantic web depends on current web standards and technologies. In this section important semantic web technologies such as The RDF and the SPARQL will be discussed in more details.

2.1 Semantic web technologies

There are many candidate technologies in the semantic web. The semantic web technologies described in the layered approach as shown in Figure 1.This figure shows the semantic web architecture. The structure starts with the hypertext web technologies which are the URI (Unified Resource Identifier) and Unicode. The URI provides methods for uniquely identifying any web resources. Unicode is used to represent and manipulate text in many languages. Above the hypertext web technologies the XML (Extendable Markup Language) is existed. The XML is the markup language that enables creation of documents with structured data. It is important language for the semantic web. Semantic web gives semantics to the structured data using XML. The following layer is Resource Description Framework (RDF), which is the basic technology for building semantic web. It is a language for describing information and resources on the web. Putting information into the RDF files, makes it possible for computer programs ("web spiders") to search, discover, pick up, collect, analyze and process information from the web. The Semantic Web uses the RDF to describe web resources. The next layer is the RDFS which is the RDF Schema. It provides basic vocabulary for the RDF. Using RDFS it is for example possible to create hierarchies of classes and properties. The next layer is the Web Ontology Language (OWL). It extends the RDFS by adding more advanced constructs to describe semantics of the RDF statements. It is knowledge representation language for representing ontology. Also, the SPARQL Protocol and RDF Query Language (SPARQL) which is the RDF query language. It is used to query any RDF-based data. It is necessary to retrieve information for semantic web applications. Logic, Proof and Trust layers are undergoing active research. User interface is the final layer that will enable humans to use semantic web applications. Semantic web layers are shown in Figure 1.

The shaded rectangle contains finally standard technologies. These technologies used in the current web. The solid rectangle contains technologies standardized by W3C. The doted rectangle contains technologies still in experimental view.

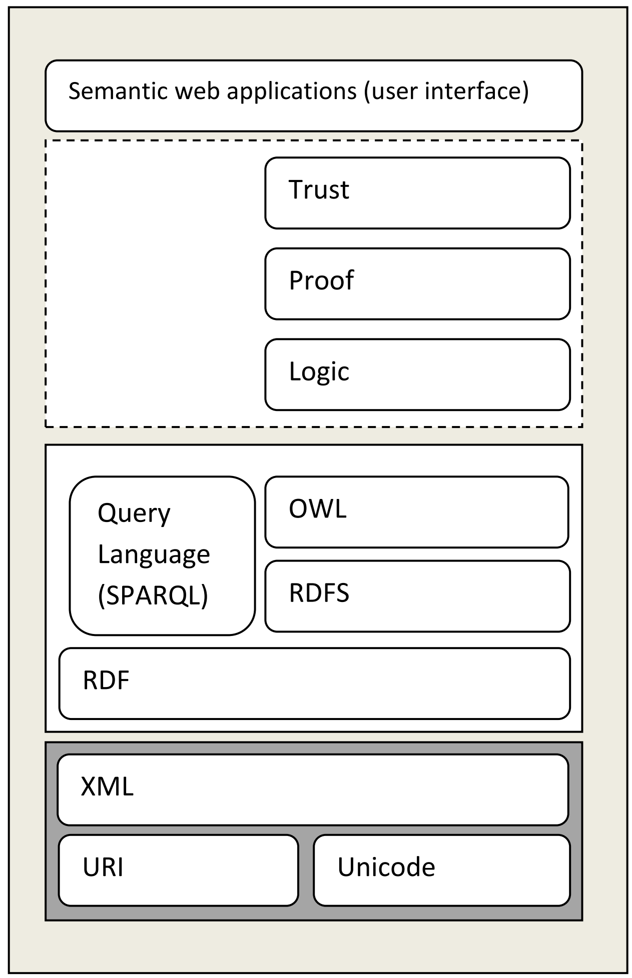

Figure 1. Semantic web layers

The illustration and the view of the semantic web architecture was created by Tim Berners-Lee [1],[5]. The most important technologies in the semantic web are the RDF and the SPARQL. The RDF is used for building the semantic web. The SPARQL language is used for querying semantic web. We will illustrate these two technologies in the next subsections.

2.2 The Resource Description Framework technology

The RDF (Resource Description Framework) is a language for describing information and resources on the web. Putting information into the RDF files, makes it possible for the computer programs ("web spiders") to search, discover, pick up, collect, analyze and process the information from the web [11]. The Semantic Web uses the RDF to describe web resources. RDF usually displayed as A Subject-Predicate-Object. If you used the RDF for representing data, you need a way for accessing information that mirrors the flexibility of the RDF information model. The RDF query languages such as SPARQL query language.

The RDF model for the web can be considered as the equivalent of the ER (Entity-Relationship) model for the RDBMS (Relational Database Management System). Let's look at a simple example. Consider the fact that "The book was written by Jane Doe." In a traditional ER model, this information would be expressed as shown in Figure 2.A. An RDF graph of the same concept would be represented as in Figure 2.B.

The RDF graph represents a node for the subject "book", a node for the object "author", and an arc/line for the predicate relationship between them. The goal of both the ER model and the RDF graph is to provide a human-readable picture that can then be translated into machine-readable format. Where a physical model in the relational database world can create the DDL (Data Definition Language) to execute on a database, the RDF graph can be translated into "triples" where each node and predicate is represented by the URI (Uniform Resource Identifier) which provides the location of the information on the web network. For example, the above RDF graph can be represented as a triple shown in Figure2.C.

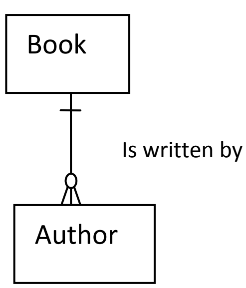

Figure 2.A. ERD model

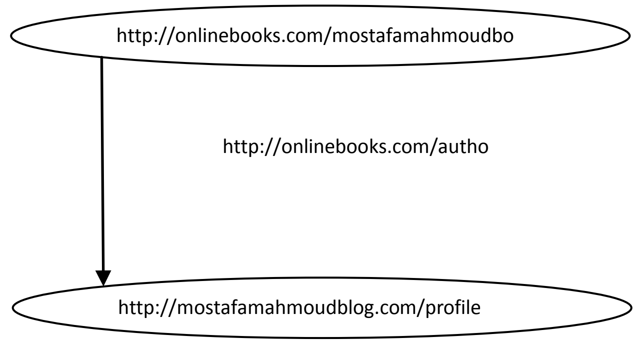

Figure 2.B. RDF model

<http://onlinebooks.com/mostafamahmoudbook>
<http://onlinebooks.com/authors>
<http://mostafamahmoudblog.com/profile>

Figure 2.C. RDF triple

2.3 SPARQL Protocol and RDF Query Language

We provide the SPARQL endpoint for querying semantic web or knowledge base. Client applications can send queries over the SPARQL protocol to the endpoint. In addition to the standard SPARQL, the endpoint supports several extensions of the query language that have proved useful for developing client applications, such as full text search over the selected RDF predicates, and aggregate functions, notably COUNT(). To protect the service from overload, limits on query complexity and result size are in place. The endpoint is hosted using Virtuoso Universal Server [11].

We conduct simple query on simple RDF triple of figure 3. This simple example shows a SPARQL query to find the author of a book from the given data graph. The query is shown in figure 3. The query consists of two parts: the SELECT clause identifies the variables to appear in the query results, and the WHERE clause provides the basic graph pattern to match against the data graph. The basic graph pattern in this example consists of a single triple pattern with a single variable (?author) in the object position. This simple query means what is the author of mostafamahmoudbook. The query result will be http://mostafamahmoudblog.com/profile

The SPARQL allows users to write globally unambiguous queries. For example, assume we have data contains name and mail of some persons. These data is shown in figure 4.A. We need to conduct query to display all names and mail .The query in Figure 4.B. The query returns names and emails of every person in the world as shown in Figure 4.C.

```
SELECT ?author
WHERE
{
<http://onlinebooks.com/mostafamahmoudbook>
<http://onlinebooks.com/authors > ?author.
}
```

Figure 3. RDF query of RDF triple of Figure 2.C.

```
@prefix foaf:  <http://xmlns.com/foaf/0.1/> .

_:a  foaf:name   "Johnny Lee Outlaw" .
_:a  foaf:mbox   <mailto:jlow@example.com> .
_:b  foaf:name   "Peter Goodguy" .
_:b  foaf:mbox   <mailto:peter@example.org> .
_:c  foaf:mbox   <mailto:carol@example.org> .
```

Figure 4.A. Data contains names and mails of two persons.

```
PREFIX foaf:   <http://xmlns.com/foaf/0.1/>
SELECT ?name ?mbox
WHERE
  {
 ?x foaf:name ?name .
   ?x foaf:mbox ?mbox
  }
```

Figure 4.B. SPARQL query for mails data.

```
 _:a foaf:name  "Johnny Lee Outlaw" .
  _:a foaf:box   <mailto:jlow@example.com> .


 _:b foaf:name  "Peter Goodguy" .
  _:b foaf:box   <mailto:peter@example.org> .
```

Figure 4.C. output of SPARQL query in figure 4.B.

2.4 Semantic web mining

Semantic web mining combines two research areas semantic web and web mining. In our previous terms we introduce semantic web. So we concentrate on describing web mining and it is relationship to semantic web. Web Mining is moving the World Wide Web toward a more useful environment in which users can quickly and easily find the information they need. Web mining is to automatically extract structured information from unstructured or semi-structured documents. The goal of web mining is to find desired pieces of information in natural language texts and store them in a form that is suitable for automatic querying and processing. The Semantic Web provides formats and standards for storing and processing such data. So web mining approaches are used for building semantic web [13], [3].

3. THE PROPOSED APPROACH

Proposed approach consists of two operations. First, we extract and structure data. Second, we build RDF triples of extracted data. And query these data. With our suggested approach we build a complete approach of a semantic web application.

3.1 Extracting and structuring framework

In the system we have two combined operations, extraction framework and structuring framework. Extraction framework is used to extract data, by parsing the HTML code. We use pythonic methods to parse the HTML source code. Before extraction we need to know which data to be extracted. We proposed a system for structuring web page table from our point of view. This structuring works well for most of the cases. During the exhaustive study, we try to use regular expressions but we do not get a good result.

The system is capable of extracting data and putting them into the RDF triple. The RDF triple consists of a subject, a predicate, and an object. Our approach consists of several steps as shown in Figure 5:

- Locate title of page as the subject: Using parser we can get title tag and the string after the title tag is the subject.
- Locate table in the page: It is easy for a human to locate the table of interest; we can get it using <HTML> tag
- Locate headers of the table as attributes: the field (column) name is RDF property
- Locate data as values: the data in fields (table cell) is a value.
- Form triple subject, predicate, object: after locating subject, predicate and object we get the triple.

Parsing and locating data or code is an important trend in natural language processing. In our system Beautiful Soup package provides a rich interface for parsing data. Parsing data using these pythonic idioms have several advantages.

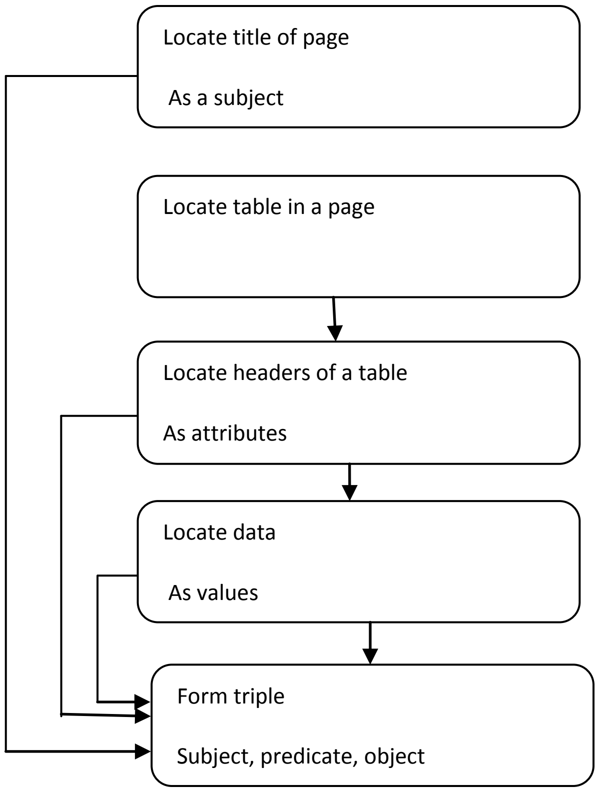

Figure 5. Structuring framework

Beautiful Soup won't choke if you give it bad markup. It yields a parse tree that makes approximately as much sense as your original document. Using python tool, we can parse data or web page table code. And our structuring system decides which data we interested in. the end of the system will be a data structure contains all triples.

3.2 Build and query RDF triples

After applying the proposed system and getting the important data as triples. We can get the RDF triples and we will be able to query this RDF triples using the SPARQL query language. After getting data, we will pass it to the SQL (structured query language) database. After sending the data to the SQL database we can get the RDF triples by using the D2R server tool. The D2R means database to the RDF. After getting the RDF triples, we can query this triples using the SPARQL query language. Cycle of data which converted to the RDF, then queried and getting result is shown in Figure 6.

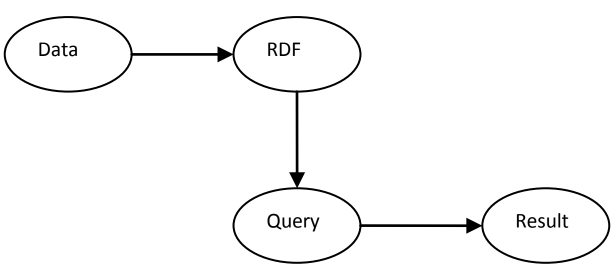

Figure 6. Cycle of building semantic web

3.3 Experimental results

Our Extracting framework and structuring framework concentrate on web page tables, as described above in the proposed system. Extraction framework interests in extracting data from the table. Structuring framework suggests a system for structuring data extracted from web page table. Assume we have web page table contains data about cars; as shown in figure 7.A. Each car described with car, year, make, model and price as columns in table. Extracting framework locates the data inside the table and ignores any other data. Extraction framework parses the HTML code of web page table. The HTML code of our example shows in figure 7.B. the extraction framework uses pythonic methods to parse the HTML and get the data. Structuring system tells us what we need from data, as shown in figure 5. We need to get title of a table as subject, columns name as predicate and data in table format as objects. Structured data must be in triple formats. Triple as we know consist of subject, predicate and object. Subject of all triples will be the title of the table. In our method we can extract it depends on <title> tag .Predicate for each triple depends on column. Predicate of all values in car column will be car. Predicate of all values in year column will be year. And so on for other columns. Predicates can be extracted by parsing the HTML code and parsing first <tr> tag and get data depends on <td> tags. Objects or values are different in each triple. Values exist in the <tr> tags except the first one. When parsing <td> tags included in <tr> tags we can get all values. With our example, extracted triples stored as lists. Each list contains three items subject, predicate and object. Some of output triples shown in figure 7.C. As shown the first list contains [u'cars info', u'Car', u'0001'] which means subject is cars info which is title of a table, predicate is car which is car column. Object is 0001 which is value. Another list is [u'cars info', u'Year', u'1999']. Which means cars info is subject; year is predicate and 1999 as value. And so on for other triples.

Structuring and extracting system get triples from web page table. If we use a page that doesn't contain a table, we get nothing according to extraction framework. Implementation status of our system described later in details .Problems exists also described later.

Data extracted from web page table in structuring and extracting system is not the RDF triple. It is only the data we interests in and we put it in a triple format. We send data to the SQL database as shown in figure 8.A. Using the D2R server tool, we convert data to the RDF triples. Output the RDF triples can be queried using the SPARQL query language. For example if we need to get all data about cars; in other words we needs to get all subjects, predicates and objects, the query will be look like figure 8.B. and sample result of the query shown in figure 8.C.

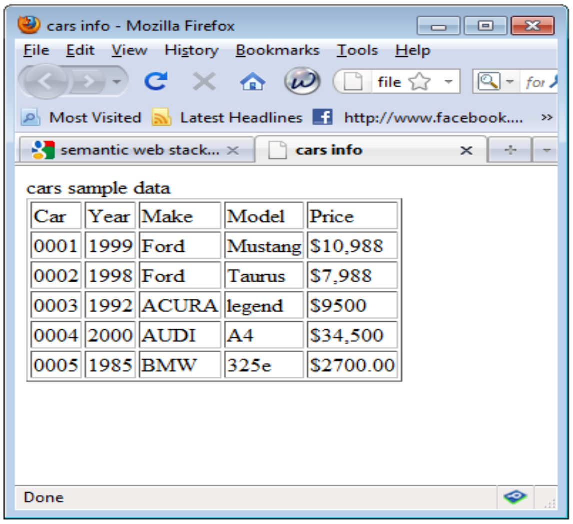

Figure 7.A. sample web page table

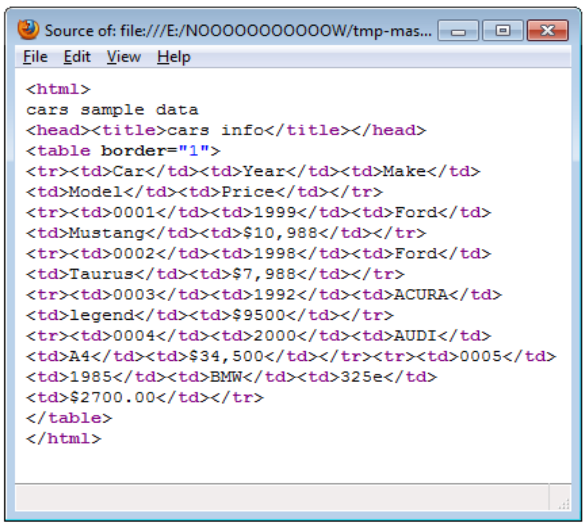

Figure 7.B. web page table HTML code

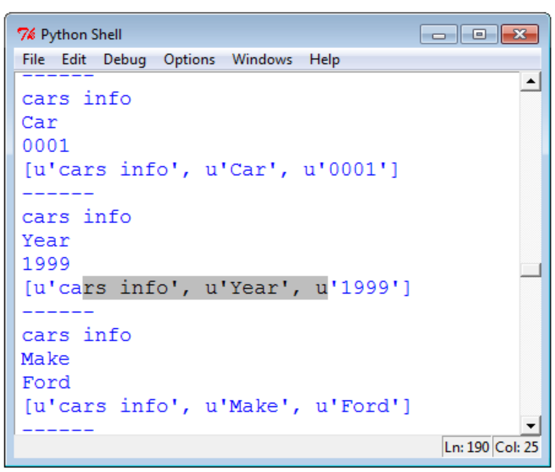

Figure 7.C. sample of data extracted (triples)

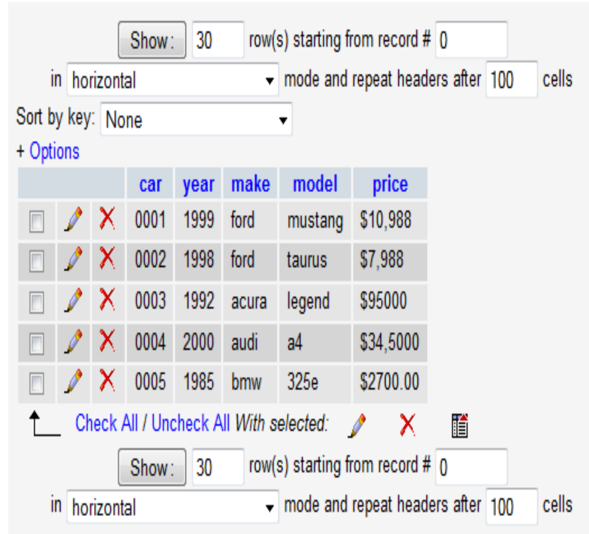

Figure 8.B. querying RDF triple

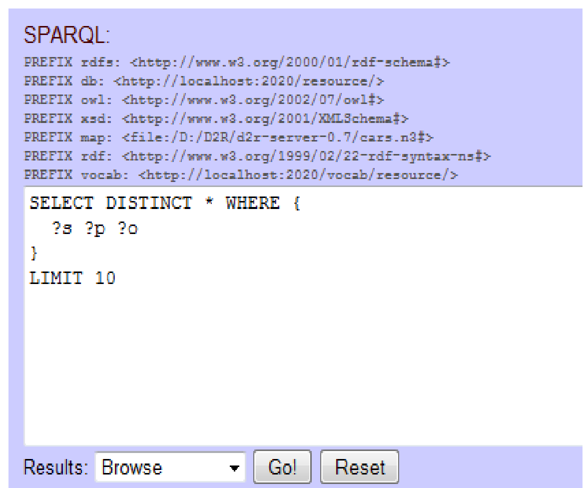

Figure 8.A. data stored in SQL database

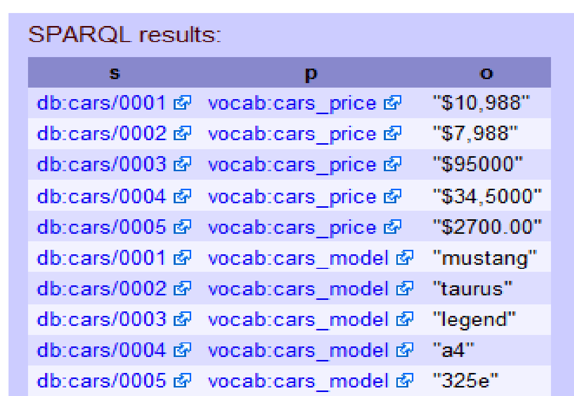

Figure 8.B. querying RDF triple

From the study, the vision of semantic web could be drawn. Generally, web page table converted to semantic web form. The proposed system considered a general system and can work on any web page table. First we could extract important data and store it in the data structure. Then, store extracted data in the database, subsequently we use the D2R server to build the RDF triples from stored database, therefore we can query these the RDF triples. As a result of our system, we could build semantic web application and query this application as a database. And making a web like a data base is the vision of the new form of web which is called semantic web.

4. EXTRACTING AND STRUCTURING FRAME WORK PROBLEMS

Extracting and structuring part of our proposed system parses data, locates important data and extracts it. Some problems occur during location and extraction problems. In this section we consider location problems and extraction problems.

4.1 Location problems

- Extract table from the page: It is easy for a human to locate the table in the page, but it is a non trivial task for the machine.
- Tables spanning multiple pages: table may contain data in the page and you have to click the next button to go to other page to get the reminder of data.
- Not complete tags: if the closed tag of <table> is not exists or there is error in and table tags.
- Using <ul> tag: some tables does not tag table with a <table> tag, but rather with a <ul> tag, making it an HTML list. Our system works only using the table tag.
- Extra information: some important data may be not in a table and exist as text under table.

4.2 Extraction problems

- Merged attributes/values: two attributes /values on the same cell, our system works general not in a specified domain. It works as a general system, so it is not easy to differ between data in cells.
- Linked information and Picture information: our system works only on a text data.
- Position of attributes: attributes may be in the left column or in the top row; our system can work if attributes are in the top row or in the left column.

5. IMPLEMENTATION STATUS

The proposed table-understanding system is implemented using python. We use the HTML parser, regular expression and beautiful soup interfaces. Status of system is shown in Table1. We characterize its current status as "prototype quality." It accomplishes a great deal, but does not handle every situation we have encountered. Table 1 gives a summary of the implementation status with respect to some problems we encounter. In Section IV we described a number of problems related to locating and extracting data from the HTML tables. Table 1 indicates our progress with each problem.

As shown in Table 1 we have the current work and the future work. The current work is the work which is already developed. The future work contains two states within scope as we want to create more general system and beyond scope that is outside scope of our system.

Table 1: Implementation Status and Future Work

Problem	**Current work**	**Future work**	
	Implemented	**Within scope**	**Beyond Scope**
Page contains more than one table	Yes		
Page contains nested tables		Yes	
Tables spanning multiple pages		Yes	
Not complete tags	Yes		
tag – list tag		Yes	
Merged attribute - value		Yes but it is not logical	
Position of attributes	yes		
Linked information			Yes
Picture information			Yes
Extra information that is not in table but it is important			Yes

6. CONCLUSIONS AND FUTURE DIRECTIONS

A simple system for structuring and extracting information from web page table is proposed. After that, convert the extracted data to the semantic web form. After getting RDF data, these data can be queried using SPARQL query language. The proposed system works well for web page tables.

As conclusion of the exhaustive study in this paper, the vision of semantic web can be drawn. Generally, the web page table is converted to the semantic web form. The proposed system considers a general system and can work on any web page table. important data can be extracted and stored in a data structure. Then, store the extracted data in the database, consequently, using the D2R server to built the RDF triples from stored database, and then these RDF triples can be queried. As a result of our system, we could build semantic web application and query this application as a database.

In structuring and extraction framework we want to implement the ideas we marked as being within scope in Table 1. As A future work, we want to parse all contents of HTML pages and convert them to the semantic web form using pythonic methods. From our study, Pythonic methods will be a good choice than java that is used today by a lot of researchers in semantic web area. Also, we need to consider web page table containing tables with related data. From database point of view, it is called Relational database. Table containing related data will be converted to Relational database then convert it to RDF which can be simply queried.

7. REFERENCES

[1] Antoniou, G. and Harmelen, F. V. 2008. A semantic Web primer - The MIT Press, second edition.

[2] Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Ives, Z., Cyganiak, R. 2007. DBpedia: A Nucleus for a Web of Open Data. 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference.

[3] Baumgartner, R. , Gatterbauer, W. and Gottlob, G. 2009. Web Data Extraction System," in encyclopedia of database systems.

[4] Berners-Lee, T. 2005. Primer: Getting into RDF & Semantic Web using N3. In WWW. http://www.w3.org/2000/10/swap/Primer.html.

[5] Berners-Lee, T. 1999. Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web by Its Inventor, ISBN 1402842937.

[6] Bizer, C. 2003. D2R MAP: A database to RDF mapping language. In WWW.

[7] Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R. and Hellmann, S. "DBpedia – A Crystallization Point for the Web of Data", Journal of Web Semantics: Science, Services and Agents on the World Wide Web, Issue 7, Pages 154–165, 2009.

[8] Embley, D. W., Tao, C. and Liddle, S. W. 2002. Automatically Extracting Ontologically Specified Data from HTML Tables with Unknown Structure. In Proceedings of the 21st International Conference on Conceptual Modeling.

[9] Embley, D. W., Tao, C. and Liddle, S. W. "Automating the extraction of data from HTML tables with unknown structure," in Data & Knowledge Engineering, vol.54, issue.1, 2005, pp.3-28.

[10] Embley, D. W., Campbell, D. M., Smith, R. D. and Liddle, S. W. 1998. Ontology-based extraction and structuring of information from data-rich unstructured documents. in Proceeding CIKM '98 Proceedings of the seventh international conference on Information and knowledge management.

[11] Prud'hommeaux, E. and Seaborne, A. January 2008. SPARQL Query Language for RDF. In WWW. http://www.w3.org/TR/rdf-sparql-query.

[12] Shadbolt, N., Hall, W. and Berners-Lee, T. June-2006. The Semantic Web Revisited. IEEE Intelligent Systems.

[13] Stumme, G., Hotho, A. and Berendt, B. "Semantic Web Mining: State of the art and future directions", Journal of Web Semantics: Science, Services and Agents on the World Wide Web, Volume 4, Issue 2, pages 124-143, 2006.