HWPDE: Novel Approach for Data Extraction from Structured Web Pages

Manpreet Singh Sehgal
Department of information Technology, Apeejay College of Engineering, Sohna, Gurgaon

Anuradha
PhD, Department of Computer Engineering, YMCA University of Sc. & Technology, Faridabad

ABSTRACT
Diving into the World Wide Web for the purpose of fetching precious stones (relevant information) is a tedious task under the limitations of current diving equipments (Current Browsers). While a lot of work is being carried out to improve the quality of diving equipments, a related area of research is to devise a novel approach for mining. This paper describes a novel approach to extract the web data from the hidden websites so that it can be used as a free service to a user for a better and improved experience of searching relevant data. Through the proposed method, relevant data (Information) contained in the web pages of hidden websites is extracted by the crawler and stored in the local database so as to build a large repository of structured and indexed and ultimately relevant data. Such kind of extracted data has a potential to optimally satisfy the relevant Information starving end user.

Keywords
Hidden Web, Web page Extraction, Web Page Service.

1. INTRODUCTION
As per the survey [1, 2, 3] tremendous amount of the information in the World Wide Web is hidden behind the search query interfaces and is dynamically generated on user request from the search interfaces, The current web crawlers do not and cannot touch this repository as they only crawl *publically indexable web* [2] also called Surface web. Information in the surface web is purely unstructured and static and thereby unable to meet the most of the user requests. In an attempt to break the jinx 'the step' is taken in the direction of mining data for the better and improved experience of internet browsing. An approach in 'the step' has been explained in the following sections and subsections. Section 2 throws a light on the related work done in the same direction. Section 3 highlights the proposed work, beginning with a conceptual architecture and ending with a set of algorithms (supported by the snapshots) used in 'the step' this section succeeds in justifying the proposed work in a neat and simplified way. Section 4 shows the experimental results. Section 5 draws the conclusion and leaves a space for future work. The paper ends with a set of references referred in this wonderful journey.

2. RELATED WORK
A plenty of work has been done in this area. Liu and Grossman [4] proposed a novel method to mine data records in a Web page automatically which is called as MDR The technique is based on two observations about data records on the Web and a string matching algorithm. The technique of MDR is able to mine both contiguous and non-contiguous data records. Its experimental results show that the technique outperforms existing techniques substantially.

[5] Proposed effective policies for generating queries automatically. It gave a theoretical framework to investigate the query generation problem for the Hidden Web. This research is based on single keyword.

DESP [6] presents an automatic deep extractor on Deep web pages for book domain. which can extract data items and label attributes at the same time. The case of DESP is to extract books' information such as title, author, price and publisher from result pages returned from bookstore web sites. Although DESP is for a specific domain, the method used by DESP is highly adaptive and can suit other domains.

[7] Discussed the research that has been done in the area of data extraction from Hidden Web sources. It elaborates on the the advantages and disadvantages of currently existing techniques of data extraction from Hidden webpages

[8] Proposed VIPS (**VI**sion-based **P**age **S**egmentation) algorithm to extract the semantic structure for a web page. Such semantic structure is a hierarchical structure in which each node will correspond to a block. Each node will be assigned a value (*Degree of Coherence*) to indicate how coherent of the content in the block based on visual perception.

[9] Proposed a novel approach that identifies Web page templates and the tag structures of a document in order to extract structured data from hidden web sources as the results returned in response to a user query are typically presented using template generated Web pages

Yalin Wang and Jianying Hu [10] proposed a machine learning approach to detect data rich tables on a web page. Tables are used to represent relational information in web documents. Web designer choose <TABLE> tag not only for relational information display but also to create any type of multiple-column layout for easy viewing, thus the presence of the <TABLE> tag does not necessarily indicate the presence of a relational table. In their work,they defined *genuine* tables to be documents where a two dimensional grid is used for the logical relations among the cells.

3. PROPOSED WORK
In this paper a novel technique to extract table data from structured databases has been proposed. Architecture and detail of components is given in next section.

3.1 Architecture
Figure 3.1 shows the architecture of HWPDE. The data extraction process is divided into eight steps. There are multiple steps involved in filtering the desired data and formatting in a format a database can accept. Subsections in this section Illustrate these steps in a chronological order as depicted in Figure 3.1

3.1.1 Hidden Web Data Miner
The role of a Hidden Web Miner is to recognize the relevant data out of the web page and extract two type of data out of it, one as an HTML (source code) and another as a TEXT (plain

text displayed on web page). The approach used In this paper for this module is static, that is user has to select a relevant data with the help of a mouse.

3.1.2 Comma Remover and Tag replacer (with Comma)

This module takes source code copy (HTML) as an input and removes all existing commas in the source code .

those already present in the plain text. The same algorithm in the form of Line Trimmer[2] is applied at the output of Line Picker (explained in section 3.1.5)

3.1.4 Boundary Extractor

After getting commas and white spaces removed from plaintext (TEXT) form of data, the top row (first boundary) and the bottom row (second boundary) are picked up for later comparison from the other copy (HTML) after processing.

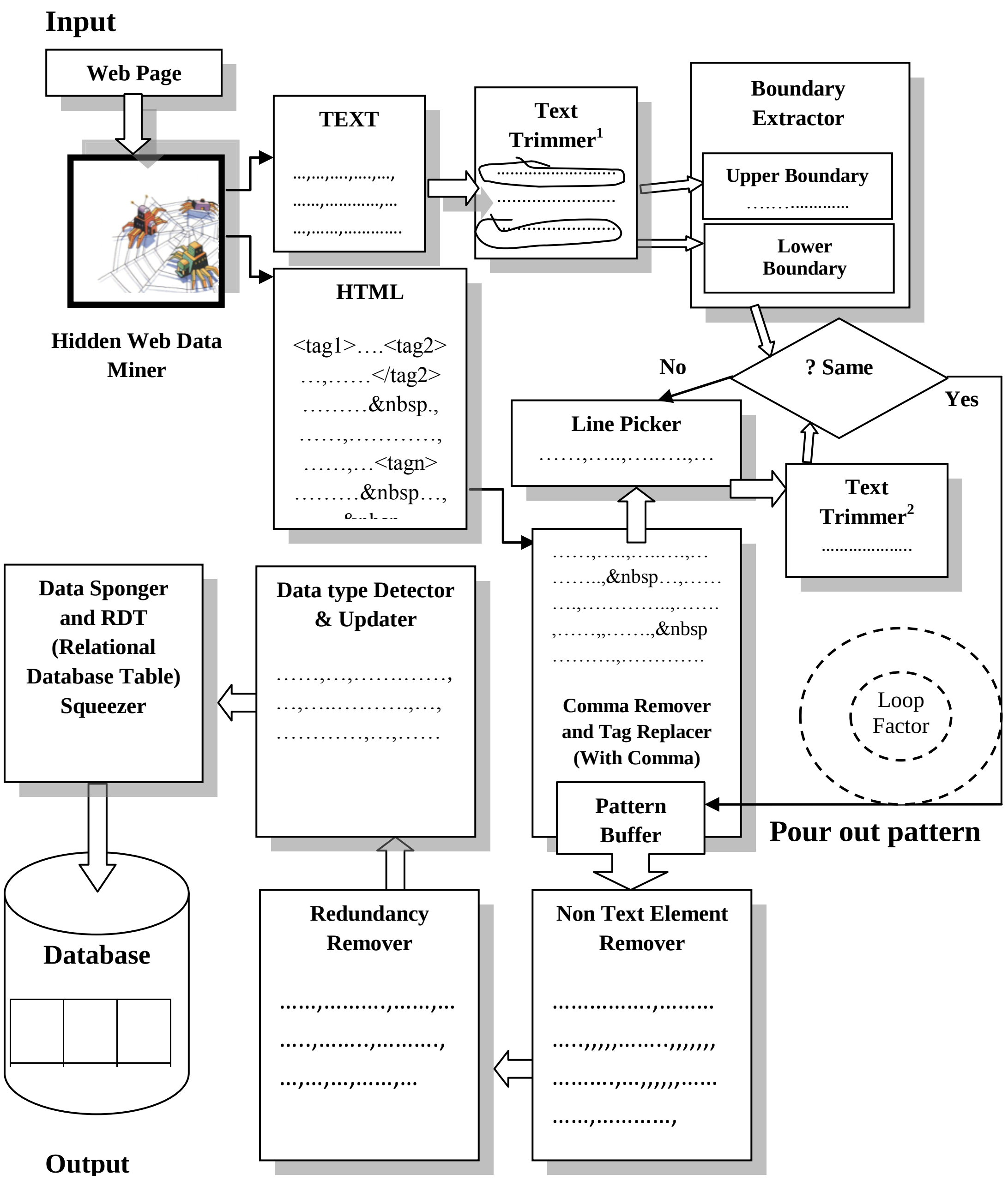


Figure 3.1 HWPDE: Hidden Web Page Data Extraction

3.1.3 Text Trimmer[1,2]

Text Trimmer[1] takes plaintext copy of the extracted data and does the same thing that Comma Remover does but alongside it also removes white spaces. It first replaces all whitespaces with commas and then removes all commas generated and

3.1.5 Line Picker and Pattern Buffer

Tag less copy of a source code is fed to the Line Picker. This algorithm parses the whole tag less copy as follows. It starts with picking up first line of the file, and provides it to the Text Trimmer which further converts it into a form comparable

With the output of the boundary extractor (Without Commas and White spaces) and gives it to the pattern comparer which compares the first boundary with the formatted line. Process goes on till the match is found. On the first match the corresponding pattern from the output of Comma remover and tag replacer (with commas) is poured into the Pattern Buffer. This Buffer will keep on accumulating the lines (Loop Factor) until the lower boundary gets matched with the trimmed textual output (from Text Trimmer[2]) of the Line Picker. On this match the corresponding pattern is fed into the pattern buffer and the process stops.

3.1.6 Non Text Element Remover

The unformatted line from Pattern Buffer might contain non text html elements such as '&nbsp;'. This module removes all such elements from the pattern buffer and hands over the resultant file to the Redundancy Remover

3.1.7 Redundancy Remover

This algorithm checks and replaces multiple occurrences of commas with a single comma and marks it as a field separator. This process also takes care of the fact that there should not be any leading and trailing commas so it detects such commas and removes them from the file

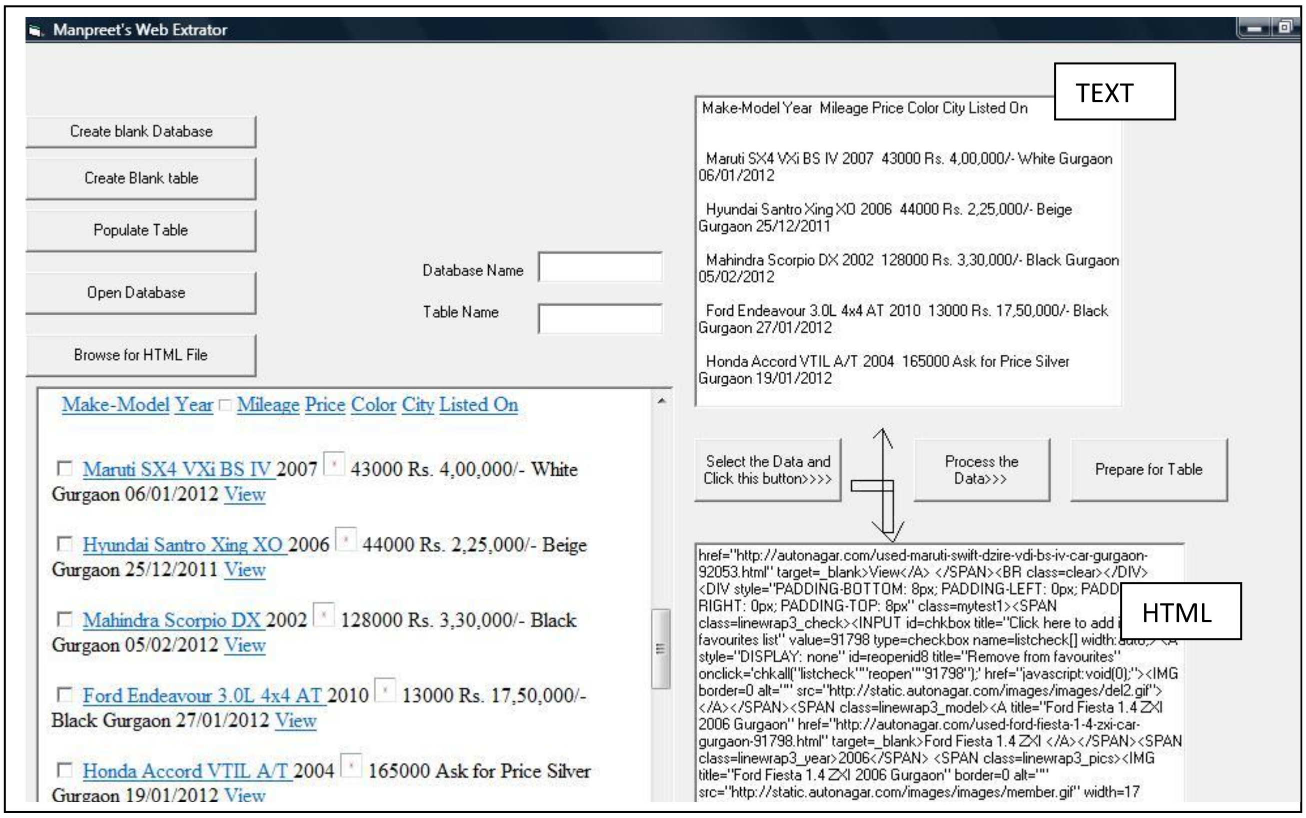
database table. Let's call it vdata (for valid data) for future reference

3.1.9 Data Sponger and RDT (Relational Database table) Squeezer

This algorithm acts as a sponger for the formatted textual data stored in a 'vdata', and squeezes the sponged data onto the table in a database. Before squeezing the data it first creates a blank relational database and than a blank table inside the database taking the first row of 'vdata' file as a column heading and from algorithm mentioned in 3.1.9 it forms a metadata of datatypes of the columns. Once blank relational table is generated it starts a loop from second line of sponged textual data, fetches it, forms an Insert Sql Query and then fires it. The result is populated table with squeezed data from a sponger.

4. EXPERIMENTAL RESULTS

This section presents the snapshots of the proposed solution. Figure 4.1 shows the front page of the proposed web extractor. With eight command buttons, four text boxes and one web browser it presents the simplistic view to the user. Following sub sections illustrates the functions of all the eight command buttons, four text boxes and a web browser in a



```
Make-Model Year  Mileage Price Color City Listed On

 Maruti SX4 VXi BS IV 2007  43000 Rs. 4,00,000/- White Gurgaon
06/01/2012

 Hyundai Santro Xing XO 2006  44000 Rs. 2,25,000/- Beige
Gurgaon 25/12/2011

 Mahindra Scorpio DX 2002  128000 Rs. 3,30,000/- Black Gurgaon
05/02/2012

 Ford Endeavour 3.0L 4x4 AT 2010  13000 Rs. 17,50,000/- Black
Gurgaon 27/01/2012

 Honda Accord VTIL A/T 2004  165000 Ask for Price Silver
Gurgaon 19/01/2012
```

```
Make-Model Year □ Mileage Price Color City Listed On

□ Maruti SX4 VXi BS IV 2007 43000 Rs. 4,00,000/- White
Gurgaon 06/01/2012 View

□ Hyundai Santro Xing XO 2006 44000 Rs. 2,25,000/- Beige
Gurgaon 25/12/2011 View

□ Mahindra Scorpio DX 2002 128000 Rs. 3,30,000/- Black
Gurgaon 05/02/2012 View

□ Ford Endeavour 3.0L 4x4 AT 2010 13000 Rs. 17,50,000/-
Black Gurgaon 27/01/2012 View

□ Honda Accord VTIL A/T 2004 165000 Ask for Price Silver
Gurgaon 19/01/2012 View
```

```
href="http://autonagar.com/used-maruti-swift-dzire-vdi-bs-iv-car-gurgaon-
92053.html" target=_blank>View</A> </SPAN><BR class=clear></DIV>
<DIV style="PADDING-BOTTOM: 8px; PADDING-LEFT: 0px; PADD
RIGHT: 0px; PADDING-TOP: 8px" class=mytest1><SPAN
class=linewrap3_check><INPUT id=chkbox title="Click here to add i
favourites list" value=91798 type=checkbox name=listcheck[] width:
style="DISPLAY: none" id=reopenid8 title="Remove from favourites"
onclick='chkall("listcheck""reopen""91798");' href="javascript:void(0);"><IMG
border=0 alt="" src="http://static.autonagar.com/images/images/del2.gif">
</A></SPAN><SPAN class=linewrap3_model><A title="Ford Fiesta 1.4 ZXI
2006 Gurgaon" href="http://autonagar.com/used-ford-fiesta-1-4-zxi-car-
gurgaon-91798.html" target=_blank>Ford Fiesta 1.4 ZXI </A></SPAN><SPAN
class=linewrap3_year>2006</SPAN> <SPAN class=linewrap3_pics><IMG
title="Ford Fiesta 1.4 ZXI 2006 Gurgaon" border=0 alt=""
src="http://static.autonagar.com/images/images/member.gif" width=17
```

Fig 4.1 Front Panel of HWPDE

3.1.8 Data Type Detector and Updater

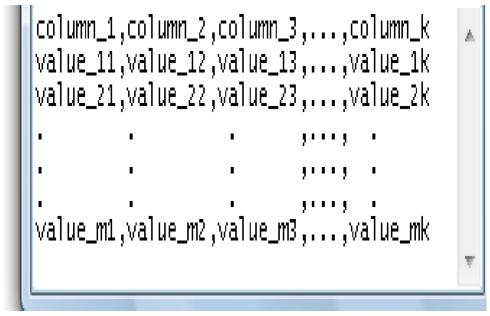
This algorithm assumes that the first line in a file is the perfect candidate for the list of column headings in the relational table in the relational database and rest of the lines are data to be populated. Building on this assumption it starts with second line and checks for the data type of each value and if it is found to be of *'string'* type or *'date'* type it encloses the field value in quotes so as to avoid any data type mismatch error with sql insert query. The output generated is the text file whose contents are guidelines and inputs to the target

quick fashion. For a reference from now on, Command Buttons will be referred as Commands. Text Boxes will be referred as Containers until or unless specified otherwise.

4.1 Commands (Eight)

For better readability, adopted Format (*italicized*) for describing the commands in this literature is *(Command Number)Name of Command. (1) Browse for HTML File'* lets a user to select a webpage to mine. *'(2) Select the Data and Click this Button >>>>'* will automatically extract the HTML and TEXT Part of the relevant data in two containers placed up and below this Command. *'(3) Process the Data>>>'* will run processing algorithms mentioned in architectural diagram in Figure 3.1 in partly sequential and partly concurrent order on both type of the relevant data for a pattern comparison motive. *'(4) Prepare for table'* command will generate a file we referred as 'vdata' (valid data) in section 3.1.9.

Once the 'vdata' is ready (Fig. 4.2 shows the format of the vdata file) the only thing left is database creation and table generation. The onus of doing this lies on the other four commands. '*(5) Create blank Database'* create a blank database in a specified directory. *'(6). Create Blank Table'* will create a table in the database. Columns of the table are picked up from the first line in 'vdata'. Figure 4.3 shows the snapshot for the Table Creation Form along with detected data types of field values of the table to be generated. Dimmed textbox entries are the detected data types of sponged data. The Create button at the bottom lets one to create a table in a database with the shown columns and data types. Here the example webpage to be extracted is the index page of www.autonagar.com which presents the query interface to the user for listing the cars.



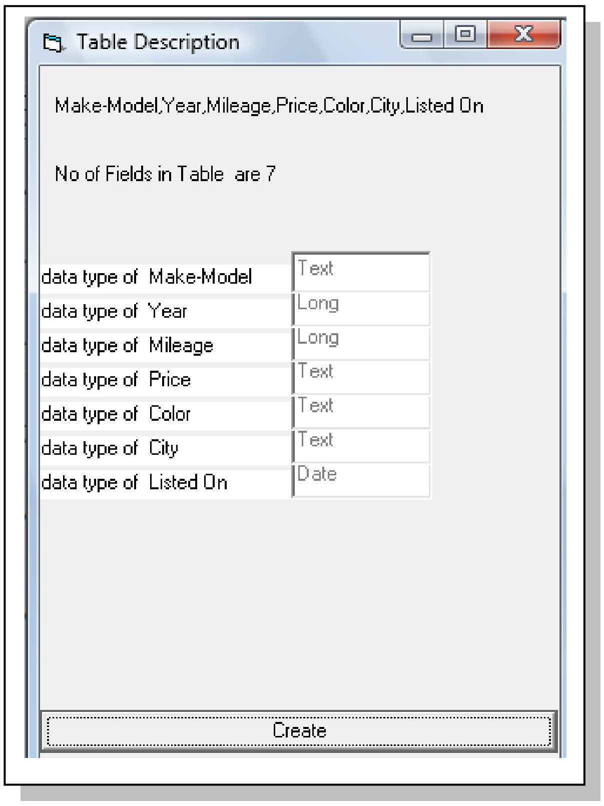
```
column_1,column_2,column_3,...,column_k
value_11,value_12,value_13,...,value_1k
value_21,value_22,value_23,...,value_2k
.        .        .       ,...,  .
.        .        .       ,...,  .
.        .        .       ,...,  .
value_m1,value_m2,value_m3,...,value_mk
```

Figure 4.2 Format for Textual Data Storage (vdata)

As an example query search string of city as Gurgaon is provided and the resultant page (hidden web page) is shown in the browser box (Fig 4.4). Fig 4.4 shows the source of this finding (a hidden web page) generated from the query interface provided by www.autonagar.com '(*7) Populate Table'* will pick up the rest of the lines from vdata one by one and will form insert sql queries to be fired into the database resulting in the table generation shown in Fig. 4.5 . *(8) Open Database* will open the created database for viewing and analyzing the result.

4.2 Containers (Four) and a Web browser

These are the input and output descriptors of the work done. Two Containers are for the Input (TEXT and HTML) from a hidden web page, while other two displays the database name and the table name of the output. Input Containers get their contents from the web browser when user browses for the hidden web file from command called 'Browse for HTML File'. Once hidden webpage gets loaded into the browser user

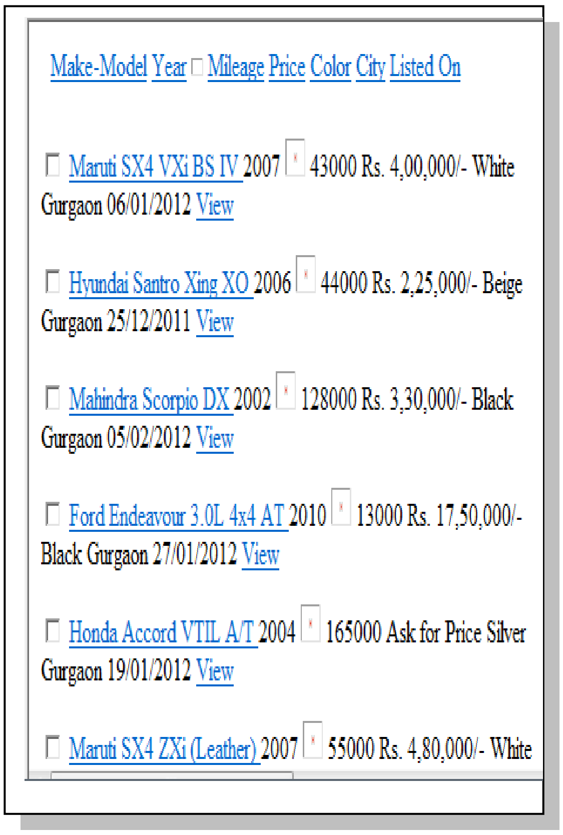


Make-Model,Year,Mileage,Price,Color,City,Listed On

No of Fields in Table are 7

data type of Make-Model	Text
data type of Year	Long
data type of Mileage	Long
data type of Price	Text
data type of Color	Text
data type of City	Text
data type of Listed On	Date

Fig 4.3 Data Type Detection and Updation Module



Make-Model Year Mileage Price Color City Listed On

Maruti SX4 VXi BS IV 2007 43000 Rs. 4,00,000/- White Gurgaon 06/01/2012 View

Hyundai Santro Xing XO 2006 44000 Rs. 2,25,000/- Beige Gurgaon 25/12/2011 View

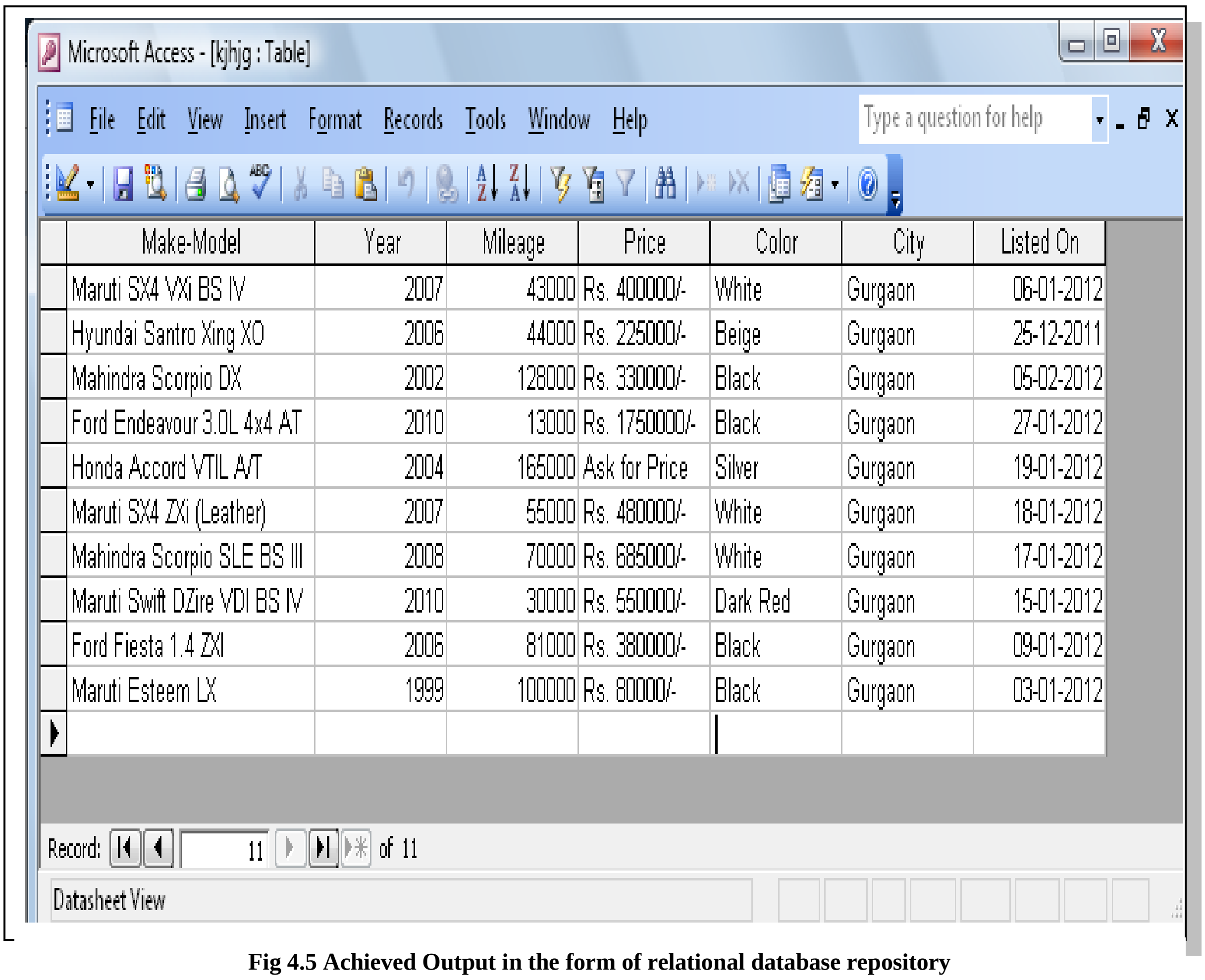
Mahindra Scorpio DX 2002 128000 Rs. 3,30,000/- Black Gurgaon 05/02/2012 View

Ford Endeavour 3.0L 4x4 AT 2010 13000 Rs. 17,50,000/- Black Gurgaon 27/01/2012 View

Honda Accord VTIL A/T 2004 165000 Ask for Price Silver Gurgaon 19/01/2012 View

Maruti SX4 ZXi (Leather) 2007 55000 Rs. 4,80,000/- White

Fig 4.4 Hidden Web Page


Microsoft Access - [kjhjg : Table]
File Edit View Insert Format Records Tools Window Help
Type a question for help
Record: 11 of 11
Datasheet View

Make-Model	Year	Mileage	Price	Color	City	Listed On
Maruti SX4 VXi BS IV	2007	43000	Rs. 400000/-	White	Gurgaon	06-01-2012
Hyundai Santro Xing XO	2006	44000	Rs. 225000/-	Beige	Gurgaon	25-12-2011
Mahindra Scorpio DX	2002	128000	Rs. 330000/-	Black	Gurgaon	05-02-2012
Ford Endeavour 3.0L 4x4 AT	2010	13000	Rs. 1750000/-	Black	Gurgaon	27-01-2012
Honda Accord VTIL A/T	2004	165000	Ask for Price	Silver	Gurgaon	19-01-2012
Maruti SX4 ZXi (Leather)	2007	55000	Rs. 480000/-	White	Gurgaon	18-01-2012
Mahindra Scorpio SLE BS III	2008	70000	Rs. 685000/-	White	Gurgaon	17-01-2012
Maruti Swift DZire VDI BS IV	2010	30000	Rs. 550000/-	Dark Red	Gurgaon	15-01-2012
Ford Fiesta 1.4 ZXI	2006	81000	Rs. 380000/-	Black	Gurgaon	09-01-2012
Maruti Esteem LX	1999	100000	Rs. 80000/-	Black	Gurgaon	03-01-2012

Fig 4.5 Achieved Output in the form of relational database repository

has to select the relevant data and then execute the command called 'Select the data and press this button.' The selected contents (Text only) gets as it is transferred into TEXT contents Container, while the source code goes to an HTML Container from where these are processed by the commands mentioned in section 4.1

5. CONCLUSION AND FUTURE SCOPE

In this paper a hidden web page data extractor has been implemented that successfully extracts the contents of users interest and stores it into the relational database. This system needs user to act as a data selector for Hidden Web Data Miner. After filling the query search interface of particular site, user will open up the hidden webpage and select the relevant data. Now, Hidden web data miner will fetch the table from webpage and store it into the local database for further use or analysis. In this the user selected text is scanned and processed to remove all the punctuation marks and spaces by the text trimmer module of type 1 (Text Trimmer^1). The source code of the hidden web page is captured and fed to the comma remover and tag replacer before it finds its place as an input to the Line Picker which picks up the processed line of html source and Text Trimmer version 2 (Text Trimmer^2) produces the line in the same format as that of the Text Trimmer^1. Boundary Extractor extracts the first line and the last line so that these can be compared with the Text Trimmer^2 Output. Getting a match is the hint to the Pattern Buffer to keep on buffering the contents from Comma Remover and Tag Replacer till (loop factor) the match with the last line from the boundary extractor does not happen and then pour out the pattern captured so far on to the Non Text Element Remover. This module filters out any &nbsp; if any. The output is checked for any redundancy of commas (commas in sequence) and the same is removed by redundancy remover. The output is a symmetrical text file containing entries separated by commas (Figure 4.2). Where first line is a column head and rest of the lines represent data to be fed to the database. Data type detection on these entries is done by simple format comparison and correspondingly the entries are updated (date and text entries are enclosed in single quotes) so that sql insert commands can be fired by the Data Sponger and RDT Squeezer without any syntax error.

Although this system works efficiently, this work could be made automated with no intervention of user. In future, this work can be extended for other domains and for multiple web pages also.

6. REFERENCES

[1] The Deep Web: Surfacing Hidden Value. http://www.completeplanet.com/Tutorials/DeepWeb/.

[2] S. Lawrence and C. L. Giles. Searching the World Wide Web. *Science*, 280(5360):98, 1998.

[3] S. Lawrence and C. L. Giles. Accessibility of information on the web. *Nature*, 400:107{109, 1999}

[4] Bing Liu, Robert Grossman, and Yanhong Zhai. Mining data records in web pages. In KDD '03: Proceedings of the ninth ACM SIGKDD international conference on

Knowledge discovery and data mining, pages 601–606, New York, NY, USA, 2003.ACM Press.

[5] Ntoulas, A., Zerfos, P., Cho, J. Downloading Textual Hidden Web Content Through Keyword Queries. In Proceedings of the 5th ACM/IEEE Joint Conference on Digital Libraries.

[6] Ji Ma; Derong Shen; TieZheng Nie DESP: An Automatic Data Extractor on Deep Web Pages Web Information Systems and Applications Conference (WISA), 2010 7th Publication Year: 2010, Page(s): 132 - 136

[7] Anuradha, A.K Sharma. "Structure based Data Extraction from Hidden Web Sources " Published in International Journal of Computer Applications (0975-8887) Volume 25-No. 3 July 2011 pages 32-37

[8] Cai, D., Yu, S., Wen, J.-R., and Ma, W.-Y. 2003. VIPS: a Vision-based Page Segmentation Algorithm. Tech. Rep. MSR-TR-2003-79, Microsoft Technical Report.

[9] Anuradha, A.K Sharma. "A Novel Technique for data extraction From Hidden Web Databases Published in International Journal of Computer Applications (0975-8887) Volume 15-No. 4 February 2011 pages 45-48

[10] YalinWang and Jianying Hu. A machine learning based approach for table detection on the web. In WWW '02: Proceedings of the 11th international conference on World Wide Web, pages