



AN INTERACTIVE TOOL FOR EXTRACTING LOW-QUALITY SPREADSHEET TABLES AND CONVERTING INTO RELATIONAL DATABASE

Arwa Awad*

Information System
Department,
Faculty of Computer
and Information
Sciences, Ain Shams
University,
Cairo, Egypt
arwaawad91@yahoo.com

Rania ElGohary

Information System
Department, Faculty of
Computer and Information
Sciences, Ain Shams
University,
Cairo, Egypt
rania.elgohary@cis.asu.edu.eg

Ibrahim Moawad

Information System
Department, Faculty of
Computer and Information
Sciences, Ain Shams
University,
Cairo, Egypt
ibrahim_moawad@cis.asu.edu.eg

Mohamed Roushdy

Computer Science
Department,
Faculty of Computer and
Information Technology,
Future University in Egypt,
Cairo, Egypt
Mohamed.Roushdy@fue.edu.eg

Received 2020-11-26; Revised 2021-1-29; Accepted 2021-02-01
Available online 2021-02-10

Abstract: Spreadsheets are contained critical information on various topics and most broadly utilized in numerous spaces. There are a huge amount of spreadsheets clients around the world. As a result of their convenience, support for announcing and portrayal as diagrams and graphs and gives their makers an enormous level of opportunity in encoding their data as it simple to utilize. Tables produce in a large amount of spreadsheet data. The expansion in volume and complexity of tables has prompted expanded necessities to preserve this data and reuse it. However, spreadsheets are hard to arrange with other data sources. As a result, it makes data stored in spreadsheets with low-quality.

We exhibited an automated extractor tool that gives the standard client a chance to concentrate on extracted relational tables from spreadsheets without experience in any programming language besides high-quality data extraction. The paper executed novel algorithms based on a heuristic approach for table extraction from a spreadsheet and implemented data improvement and quality rules using domain ontology for changing over between low-quality semi-structured data to high-quality relational data for reusability and integration as a Java program interfacing with SQL server database. The paper does experiments on 2 real public datasets. The percentage of improving the performance using the proposed approach on the 2 datasets are 100 % for extracting duplicated records and the percentage of successfully table identified are 100% and 85% respectively.

* Corresponding author: Arwa Awad

Information System Department, Faculty of Computer and Information Sciences, Ain Shams University, Cairo, Egypt

E-mail address: arwaawad91@yahoo.com

Keywords: *Spreadsheet Low-Quality, Data Cleaning, domain ontology, Relational Database, Spreadsheet Conversion.*

1. Introduction

Spreadsheets are utilized by a huge number of clients as a standard generally useful data management tool. It is currently increasingly necessary for external applications and services to consume spreadsheet data. A spreadsheet is one of the most utilized type's forms of representation for datasets of similar type. Spreadsheets provide considerable flexibility for data structure organization. Because of this flexibility, tables with very complex data structures could be generated. Thusly, such complexity makes automatic table processing and data extraction a difficult task. Therefore, the table preprocessing step is regularly needed in the data extraction pipeline [1]. In this section, we will examine the issue of spreadsheets low-quality besides the importance need for changing over arbitrary tables in spreadsheets into structured formats required by these applications and services. In addition, the paper proposes a novel methodology in which transformation logic is embedded in a spreadsheet.

The issues of low-quality for a spreadsheet whose object is data stockpiling are divided into two primary parts. The initial segment originates from the perspective on the spreadsheet itself, for example, multiple worksheets, hidden content, and various tables in the same sheet. In addition, the spreadsheet contains tables and text and diagrams on the same sheet. And so on. The subsequent part is explicit to the data store in a spreadsheet in particularly table content of the Excel spreadsheet. For example, blank cell or empty cell, no primary key, different formats: like distinctive date organizes in Excel spreadsheets, diverse text style or shading and duplicated data.

Since the increasing interest in ontology and semantics in recent years has led to the creation of ontology for different purposes and with different feature systems. There are many applications using different ontologies such as the intelligent semantic educational system [2] that is built based on ontology. Also, there is a fuzzy ontology [3] which is proposed based on OWL2 to represent the Obesity-Related Cancer domain knowledge. In addition, electronic learning management systems [4] is used ontology to improve the learning management system. Also, the authors of [5] were extracted ontology from Arabic textual resources based on Arabic natural language processing, machine learning, and text mining techniques. We differ from these all past applications as a point of view in using ontology. The paper introduced a tool that can extract tables from spreadsheets and convert them automatically into a relational database. The tool uses the ontology structure itself to identify duplicates in spreadsheet tables.

Motivation: arbitrary tables in spreadsheets

- A huge volume of important information for science and business applications
- A major assortment of format, style, and content features
- Different structure for spreadsheet tables
- No explicit semantics for understanding by machine-readable

Challenges

- How to detect tables from spreadsheets
- How to identified and recognize cell structure
- How to extract required tables for automatic interpretation
- How to improve the quality of data stored in spreadsheets

Since our main objective was to overcome these restrictions by introducing a tool that can extract tables from spreadsheet and convert them automatically into a relational database.

The rest of this paper is produced as follows. Section 2 presented the previous related work approaches of spreadsheets conversion. Section 3 reviews the experimental methods and algorithms used in the spreadsheet conversion. While section 4 introduces a case study scenario. Results and analysis are produced in section 5. Finally, the paper is finished up in section 6.

2. Related Work

The expansion in volume and multifaceted nature of data store in Excel spreadsheets has prompted expanded prerequisites to protect this data and reuse it. The greater part of the applications concentrated on extricating relational data from semi-structured data on the web by HTML and XML formats [6-9]. But a little research deals with extraction tables "Tabular Data Extraction" outside the web like us [10]. These days, there is a blast of semi-organized archives that create outside the web.

From the previous related work, the paper concluded that there are four main approaches to transforming from spreadsheet data into the database. The approaches are rule-based, visualization-based, semi-automatic based, and automatic based approaches. In this section, we will explain the four approaches with their advantages and drawbacks.

Firstly, rule-based methodologies [11-13] frequently expect clients to gain proficiency with a recently characterized language to depict the change procedure. The methodologies are adaptable however regularly require unequivocal change decisions that are troublesome and tedious for the client to create. Secondly, there is a scope of representation frameworks or visualization systems [13] that help the client explore and comprehend spreadsheets with perception methods, however, the systems are not ready to extract relational data from spreadsheets. Thirdly, mixed-initiative systems or semi-automatic systems [14-16]: many of these systems are driven by a programming language that the user must learn or need user intervention; in this paper, the system does not require the user to learn a programming language, just "open Excel Files" from the interface. After that, press "Create" to automatically convert spreadsheet tables into a repository of the database. There are three common drawbacks of these three main approaches: (A) challenges to handle hierarchal spreadsheets. (B) The transformation process cannot be accomplished automatically. (C) The approaches require users to learn a new language or predefined operators to describe the transformation rules. The opposite of that, the fourth approach: computerized methodologies or automatic approaches [17] that are the most like to us.

Some research has been done to extract table data from a spreadsheet [18, 19, 20]. However, they only focused on one of the subtasks such as spreadsheet table detection and the semantics in spreadsheets have largely been overlooked. We propose an approach to automate table detection, component recognition, and cell type recognition simultaneously on a large scale for capturing semantics in spreadsheets. In addition, improve the quality of spreadsheet table extraction.

Also, there are researches don't integrate with a relational database [21], semi-automatic and doesn't integrate with a relational database. From the paper perspective, such researches or extraction be a non-relational database while relational databases are fantastic, they do accompany exchange offs. In that circumstance, you may need to consider going with a non-relational database. A non-relational database just stores data without an organized instrument mechanism to connect data from various tables to each other.

The paper approach engages downstream spreadsheet integration applications. In this paper, an automatic extractor tool is displayed that gives the user a chance to concentrate on extracting data tables from spreadsheets without experience in any programming language. Furthermore, presents a structure that consequently removes the form of spreadsheets table content with low-quality data.

3. Framework Architecture

As showed up in Figure. 1, the framework configuration can recognize and focuses tables with their metadata from a spreadsheet. The metadata which called the function component of the table spoke to in (A) Table name which delineates the table substance (properties and qualities), (B) Attributes that depict the qualities store in a table and (C) Values which called a data cell. By then, the system applies the quality examination computation on the table extracted. Table content comprises of 1. Attributes or Properties spoke to in column names and 2.Values of cells.

The paper methodology of spreadsheet automatic conversion comprises three fundamental stages (Extract tables with their metadata, analyze quality and sort in a database) with a lot of procedures. Starting with detecting tables [22] from a spreadsheet and concentrate its metadata based on hierarchal clustering and heuristic approaches. The clustering approach refers to the strategy of grouping data (cells) into various groups (tables) depending on their characteristics (one cell or more than one cell). A heuristics approach to classify spreadsheet table cells (cell format: bold, italic, underline, font size greater than 11, any color except black, all letters capital). The heuristic rules are top, left, in the table.

At that point the paper executed quality rules for attributes name and values extracted from the table. The rules applied to the attributes name is to peruse their data type from the spreadsheet and watch that there is no missing [23] and no duplicated using knowledge base provided by ontology in the segment name. Missing implies a vacant cell and duplicated implies similar information rehashed. The examination done to the values is to watch that there is no absent, the values data type is associated with their properties data type which called valid data.

At long last is to inclusion into a relational database by deciding the data sorts of the table name and send to create a new table and deciding extracted attributes name and send to the database as columns name with their values in the wake of applying the quality management rules. The rules for table creation in a database are primary key and foreign key if found. The paper will present a brief baseline discussion of each procedure with the algorithms utilized in the following segment.

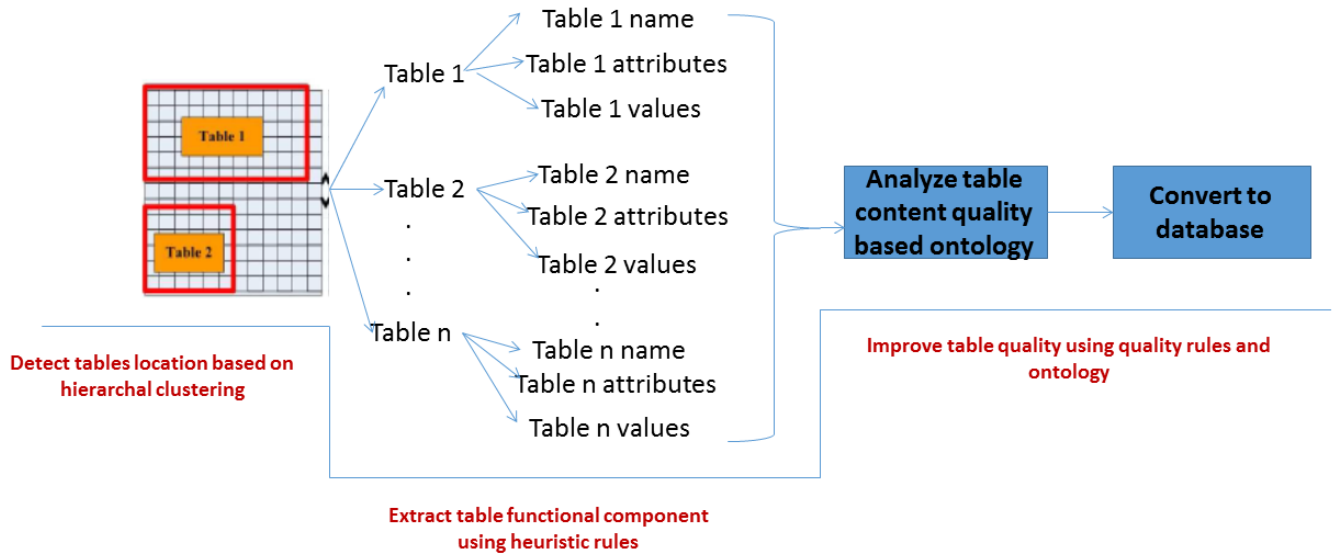


Figure. 1: Framework of spreadsheet tables conversion

3.1. The implemented approach and algorithms used

3.1.1. Approaches

Clustering Approach: Data clustering refers to the methodology of collecting data into various groups relying upon their attributes. This grouping will demand the data and thusly, further treatment of this data is made simpler. The paper utilizes a clustering approach to deal with determine the area of data cells. Each set of cells in a bunch of lines or sections are gathered one next to the other, be a group of cells called a table.

Heuristic Approach: The functional component of table detection is based on a heuristic-features approach and rules that explained in Table 1 to classify spreadsheet table cells. The heuristic features will be explained in details in the next section.

Table 1 Heuristic Rules

Rules	Area Must
Top	Top row
Left	Left column
In Table	In table rows

Ontology for Duplicate Detection: The development of an effective mechanism to detect duplicated values in spreadsheets or in a database is a critical problem for data integration. This research addresses this problem by proposing an ontology-based system for high-quality data

extraction from spreadsheet data. The ontology comprises domain knowledge and a set of rules that establish an expert system. The native reasoning support in ontology is used to produce the output knowledge from the predefined rules about duplicated detection in the application phase. The novelty of the approach lies in the use of the ontology technique that not only minimizes the human efforts cost and data modeling but also makes the data extracted extendable and reusable for different applications.

The algorithm presents for identifying duplicates in databases that are various portrayals of the same one true substance and in this manner a significant issue pertinent for example in the situation of incorporating a databases. The algorithm utilizes a classification calculation from the data mining area will be clarified in the following.

A duplicate detection algorithm using a classification algorithm that consists of two phases respectively. In the process phase, the algorithm's inputs are the distance similarity between the correspondent attributes and the knowledge about whether or not the attributes or values are duplicates, in the application phase that knowledge is produced through the classifier that is the process phase's output. Similarity characteristics are the same name and the same data type of two objects. Similarities are measured in the range 0 to 1 [0, 1], this score in the range of [0, 1] is called the similarity score. Two main consideration of similarity:

- Similarity = 1 if $X = Y$ (Where X, Y are two objects)
- Similarity = 0 if $X \neq Y$

The classification algorithm distinguishes between if there are duplicated or non-duplicated records in the spreadsheet data extracted in the knowledge phase output.

3.1.2. Algorithms

Algorithm 1: Detect and extract tables with its metadata

As appeared in Algorithm 1, the calculation is made to distinguish the table name and the section headers and values. There is a reserved place for each table to plan this fundamental information like the following:

Case 1: Table name

The algorithm starts to extract from a cell by cell. If there is not any more one filled a cell with the format of the header (check the subsequent case), that implies this information has a place with the table name.

Case 2: Table headers

To consider the found data in any cell as a piece of the table header, it ought to carry on, in any event, one of the formats of the headers (Bold, Italic, Underline, Font size greater than 11, any color except black, all letters capital). The calculation will add this information to the header's queue.

Case 3: Detect left attribute

If the algorithm found only one cell with the format of the header and there is still data in the rest of the row. That's mean this table has its headers on the left and this table should be transposed after extraction. The algorithm will get the first cell of the row as a header and the rest of the row

as data. After extracting the whole headers and data table, the table content should be transposed each row will be a column and each column should be a row.

Case 4: In table attribute

This case the same as the second case except the headers of the table do not exist at the beginning of the data. It can be in any row of the table. In this case, the queue of the table headers will be empty until the headers found during the extraction process.

Extract table values

This algorithm is designed to extract Excel spreadsheets metadata and attribute data. It considers only filled data sheets and each sheet looks for filled rows. Empty rows and columns are the main separators between tables if one data sheet contains more than one table. For example, if a data sheet contains two tables placed vertically, undoubtedly they are separated by one or more empty rows. Similarly, if a data sheet contains two tables placed horizontally, undoubtedly they are separated by one or more empty columns [21]. This algorithm will be discussed in detail in the next sections.

Extract one table from one datasheet

The algorithm starts with the first non-empty sheet and gets the maximum number of written rows and columns. After that, the algorithm checks the row data condition to consider if this data will be classified as a table name, a header or attributes (Discussed in detail in the Table name and headers extraction algorithm section). If the active row is found an attribute data, the algorithm will start to detect each cell data type if it's string, numeric or empty cell. If a table includes an empty cell, the algorithm will fill it with an appropriate value. In the case of a string, the value will be 'Unknown'; otherwise, it will be the mean of the rest of the column data in case the data type is numeric.

After the extraction process, the data will be separated into two main categories. The table name, headers, and values for each table. The values are now ready to be checked for duplication and insert it into a database. The table name used to create a new table in the database, the headers used to create the table columns considering the column type with its interconnected data types.

Extract more than One Table from One Datasheet

In the case of the existence of more than one table in the same datasheet, there are two possible causes for the positioning of them. Case1, if these tables were placed side by side separated by one or more empty columns. Case2, if these tables were placed over each other separated by one or more empty rows.

Case (1): Horizontal synchronization

In this case, if the algorithm found an empty column before the maximum number of the filled data columns, that means there is another table in the same rows. The algorithm will implement a new matrix to store the new table.

Case (2): Vertical synchronization

In this case, if the algorithm found an empty row before the maximum number of the filled data rows that mean there is another table below. The algorithm will finalize and send the current table then starts to explore the new one.

Algorithm 1: Excel Data Extraction Algorithm

Input: Excel workbook with multiple sheets $A = \{S_1, S_2, \dots, S_n\}$

Result: Separate tables each include TBL: Table name as a variable TblNm and Headers as an array list
 Head = $\{h_1, h_2, \dots, h_n\}$, values as $val_{[m,n]} = \{val_{m1n1}, val_{m1n2}, \dots, val_{m,n}\}$

Initiate A;

For each sheet in A do

 Initiate new TBL ;

 For each filled row do

 If cell has Headers format and no other data in this row Then

 TblNm<- cell data;

 else

 If a cell has Headers format and there is other data in this row; Then

 Head<- cell data.

 If the cell does not have headers format Then

$val_{[m,n]}$ <- cell data

 End If

 End If

 End If

 Else if empty row Then

 go to Initiate new TBL

 End If

 End For

End For

Algorithm 2: Analyze table quality

The data quality process can be summarized as shown in Algorithm 2, like the following: Detect if there are duplicated rows or columns then delete it. Detect missing data founded and in this rule, there are two cases: (A) Numeric data: replace the empty cell with the average of the rest of the data included in the same column. (B) String data: replace the empty cell with "Unknown". Check different formats that represented in visual impacts like different font styles or colors.

Algorithm 2: Data Quality Algorithm

Input: Attribute data $Att_{[m,n]} = \{att_{m1n1}, att_{m1n2}, \dots, att_{m,n}\}$

Result: Attribute data after check values $Att[m,n] = \{att_{m1n1}, att_{m1n2}, \dots, att_{m,n}\}$

Initialization;

For each row m do

For each column n do

if

cell have no data ; // missing

data type is string ; then

cell data<- print 'Unknown ;

else

if data type is Numeric then

cell data<- calculate column average ;

End if

End if

if entire column exist before then

Delete // duplicated ;

else

if entire Row exist before then

Delete;

end

end

Algorithm 3: Automatic Creation and Insertion into a Database

To make a table inside the database, you should initially talk about the primary and foreign keys [24] algorithm as shown in Algorithm 3. Determine an essential key for each table to avoid duplication of data inside the database. A foreign key is to connect tables inside the database without copying information in the essential key table. After finishing the procedure of value examination of the data store inside the Excel spreadsheet, the transformation period of this information starts to sort into a database of high-quality by, First: check the attribute name and its type to pick an essential key and send to create another table with sections names inside the database. Second: insert the qualities of the cells in the wake of applying the quality calculation on them withstanding checking the kind of this information and contrast and the sort of segment name.

Primary key sitting:

The client of the system can pick any recorded of the table section, new personality field or without essential key (PK). On account of the current segment that ought to be checked to be

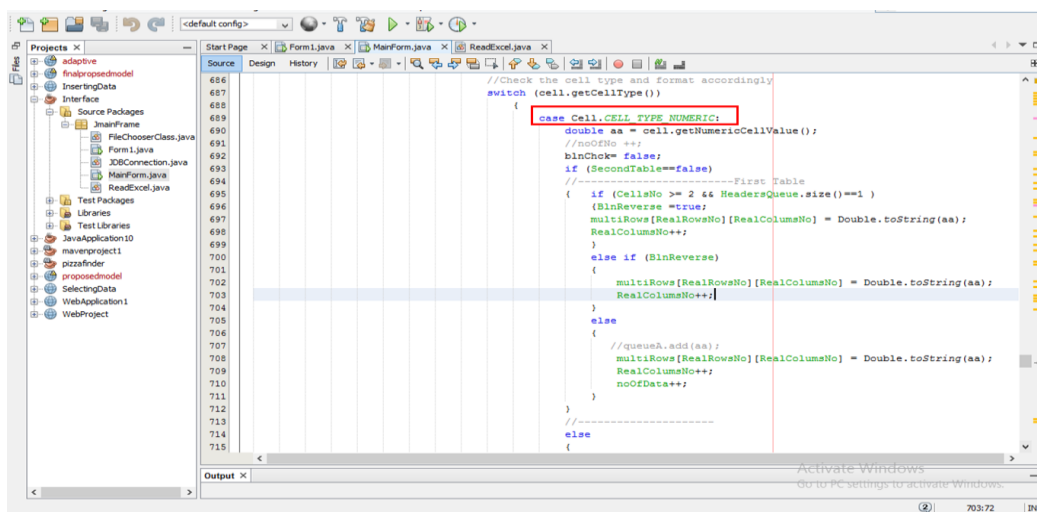
certain that the segment is processable with the essential key obliges, generally the framework will request that the client pick another field or fix the information. The PK limitations act in there is no copied or missing inside the field picked to be a PK, which makes it unique and complete.

Foreign key automatic detection:

After information included in the database, the calculation begins to look at the current essential keys in the objective database, and contrast them and every section in the present table. If the calculation found a match between an essential key and a field in the present table, the coordinated segment will be a foreign key to the coordinated essential key. The coordinating procedure incorporates the fields' complete name and the data type. The purpose of the foreign key is to make a relationship between two tables and keeps up data integrity and allow navigation between two different instances of an element. It goes about as a cross-reference between two tables as it references the primary key of another table.

Datatypes detection:

Data types define what type of data the column contains. The proposed approach defines three types of cell-datatype in the Excel sheet, numeric, string, and blank cell. Numeric cell type contains numbers. String cell type contains text and numbers. Blank cell type contains nothing which means it is an empty cell. Figure. 2, and Figure. 3, shows the automatic detection of the three data types from the source code.



```

686 //Check the cell type and format accordingly
687 switch (cell.getCellType())
688 {
689     case Cell.CELL_TYPE_NUMERIC:
690         double aa = cell.getNumericCellValue();
691         //noOfNo++;
692         binChok= false;
693         if (SecondTable==false)
694             //-----First Table
695             {
696                 if (CellsNo >= 2 && HeadersQueue.size()==1)
697                 {
698                     {BinReverse =true;
699                     multiRows[RealRowsNo][RealColumnsNo] = Double.toString(aa);
700                     RealColumnsNo++;
701                 }
702                 else if (BinReverse)
703                 {
704                     multiRows[RealRowsNo][RealColumnsNo] = Double.toString(aa);
705                     RealColumnsNo++;}
706                 }
707             else
708             {
709                 //queueA.add(aa);
710                 multiRows[RealRowsNo][RealColumnsNo] = Double.toString(aa);
711                 RealColumnsNo++;
712                 noOfData++;
713             }
714         }
715         //-----
716         else
717         {
718         }
719     }
720 }

```

Figure. 2: Automatic detection of numeric cell type

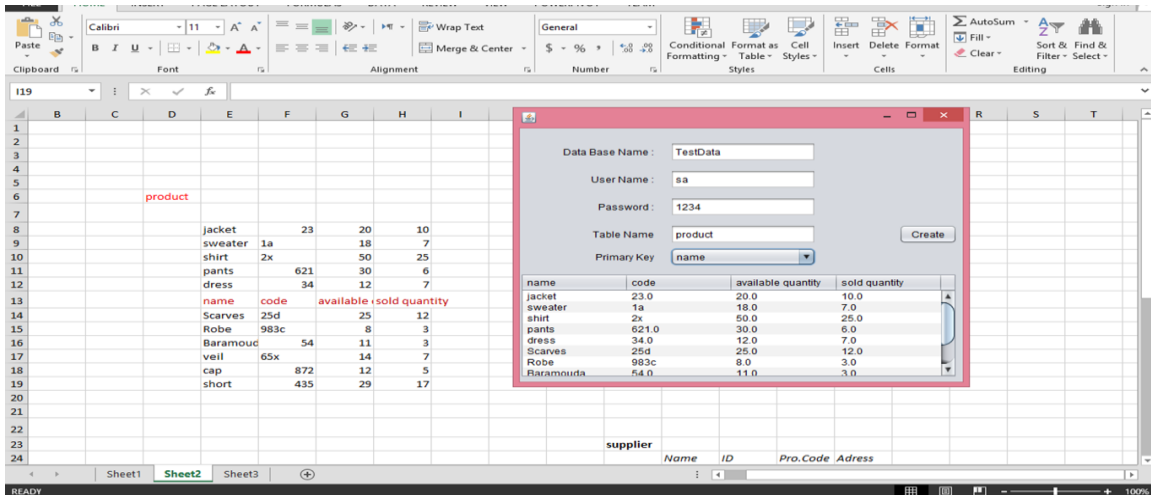


Figure. 5: Sample of data input inside the spreadsheet and its output

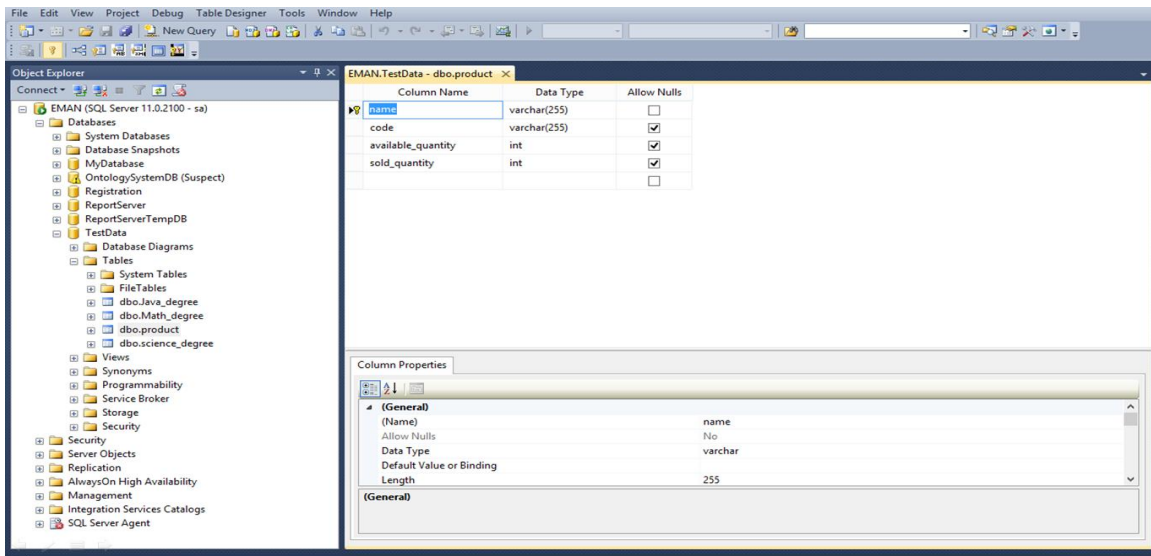


Figure. 6: The result of the input example with their datatypes

Algorithm 3: Database Creation Algorithm

Input: Separate tables each include TBL:Table name as a variable TblNm Headers as an array list Head = {h₁, h₂, ..., h_n}, column types as an array list ClmTyp = { Typ1, Typ2,}, Table primary key name TBLPKAttribute data Att[m,n] = {att_{m1n1}, att_{m1n2}, att_{m,n}} and existing Database DB

Result: Database table with its primary key and foreign key if required

Initialization;

For each table TBL do

Create a new Table TblNm with Head and ClmTyp;

Create a new constraint TBLPK data type is string ;

```

    For each Head do
    if Head has the same name and type
    of a Pk in DP then
        Set Head as FK;
        End if
    End For
    For each row do
        Insert data;
    end

    end
end
    
```

4. Case Study Scenarios

Given an example of an Excel spreadsheet document as input. The document contains multiple sheets. Each sheet contains multiple tables. The paper speaks to the extraction of one table for instance as appeared in Figure. 7. Each table, in any event, should contain his attributes in the left or top and their values and table name whenever found. The extraction calculation begins to identify and extract the tables with their metadata as appeared in Figure. 8. (Phase1).

	A	B	C	D	E	F	G	H
1	Empolyees							
2								
3	Emp ID	Emp Name	Job	Salary	Marital	Dep ID	Off ID	
4	20	Ibrahim	Eng	1800	Yes	1	2210	
5	40	Wael		1500	No	2	3330	
6	33	Ibrahim	Eng	2500	Yes	1	2210	
7	34	Fathi	Manager	3000	Yes	2	4343	
8	34	Fathi	Manager	3000	Yes	2	4343	
9								

Figure. 7: A sample of spreadsheet input

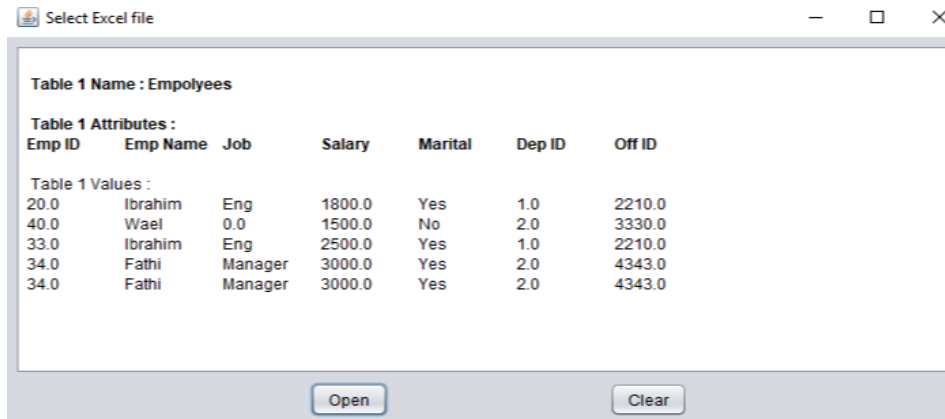


Figure. 8: Extract table with their metadata

Stage 2, the quality investigation of table substance. As appeared in Figure. 9, the calculation checks the PK choice, it can be any field which means a table can have multiple choices for a primary key but just one field can be set as the primary key. Also, the user can choose a new field to be a PK or can choose to create a table without any PK. Besides, the algorithm checks if there is a missing or copied in the information extracted from the spreadsheet at that point erase the rehashed data, and complete the missing cell as appeared in Figure. 10. What's more, in Figure. 11, if there should arise an occurrence of PK conditions doesn't accomplish as the PK setting must be unique and not null. So, the system will produce a warning message to fix the PK chosen if there is a missing value or duplicated values founded in the column of PK chosen. However, different remaining values in the rows.

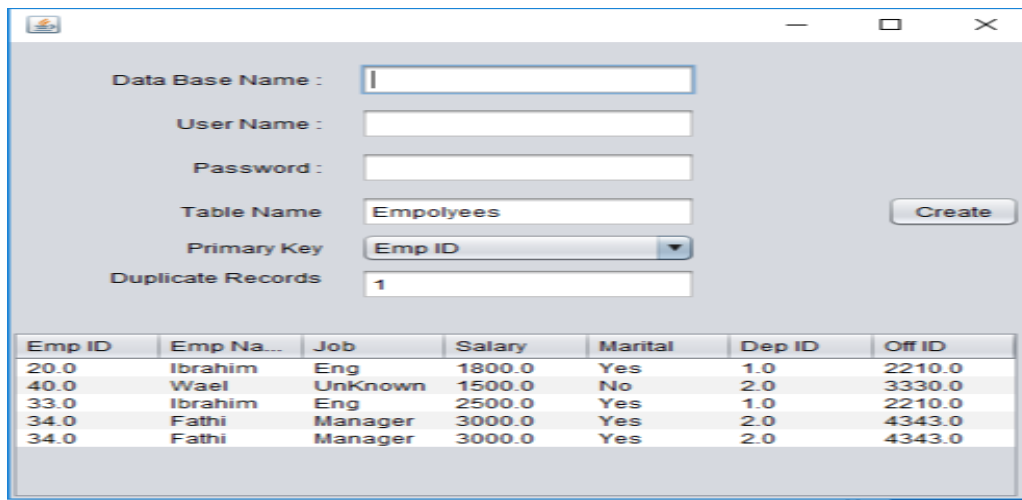


Figure. 9: The analysis quality of a spreadsheet input. The figure shows that there is a duplicate in the "Emp ID" column and PK error in that field, also missing in "Job" column as shown in figure. 2



Figure. 10: Analysis of quality figure. Notice that no duplicated or missing

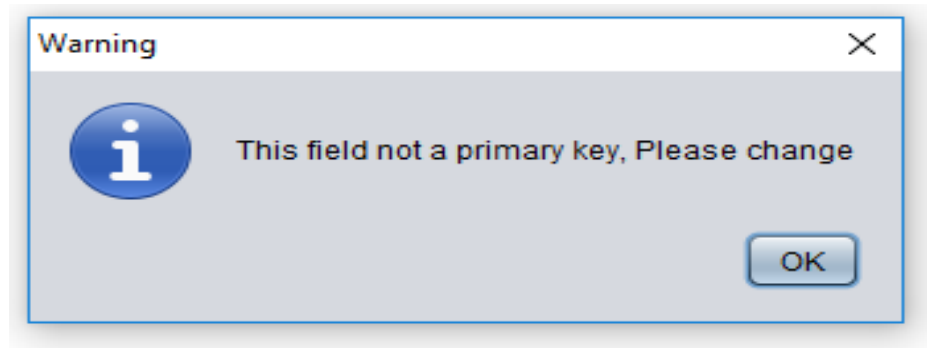


Figure. 11: A warning message to fix the primary key

Stage 3, is to the automatic creation of a table separated inside the database as appeared in Figure. 12, the client just pushes on the "Create" button.

emp_id	emp_name	Job	Salary	Marital	dep_id	off_id
20	Ibrahim	Eng	1800	Yes	1	2210
33	Ibrahim	Eng	2500	Yes	1	2210
34	Fathi	Manager	3000	Yes	2	4343
40	Wael	UnKnown	1500	No	2	3330
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Figure. 12: The representation of relation tables output inside the database

5. Experimental Results and Analysis

The evaluation of the accuracy of the approach is used and illustrate it by utilizing the gov.au.data dataset. We downloaded 2 genuine open datasets from [25], with Excel (.xlsx) Formats. Every dataset contains multiple sheets and every sheet may contain only one table or multiple tables as illustrated in Table 2. The evaluation supported table detected and extraction for attributes and values without a table name.

Table 2 Experimental Results

Datasets	Number of sheets	Number of randomly selected sheets, tables	Number of successfully identified tables	Percentage of successfully identified tables %	Percentage of successfully extracted duplicated records
1-APS Employment Data 30 June 2019 release	82 sheet	20 sheets with 29 tables	29 tables	100 %	100 %
2-Australian Public Service Statistical Bulletin - December 31, 2015	63 sheet	20 sheets with 20 tables	17 tables	85%	100 %

The experimental results showed that the accuracy of table detection and extraction is 100%, 85%, of datasets 1, 2 respectively with the average percentage of 85%. In summary, the majority of the spreadsheet tables contain some table characteristics. To extract a large corrected number of tables with relational form, you have to identify a variety of heuristic tables' cells feature and rule. However, the percentage of extracted duplicated records from a spreadsheet is 100% in the two datasets.

The application is written in JAVA. To test it, an executable jar file is made. The hardware specifications for the system are, Intel® Core™ 2 Duo CPU with 4 GB of RAM. The adaptation of Microsoft Excel introduced on the machine is 2013 however the application works with'.xlsx' format. The H2 database version used is 1.3.172.

6. Conclusion and Future work

The paper introduced an interactive automated extractor tool that gives the user a chance to concentrate on extracted relational tables from spreadsheets without experience in any programming language besides high-quality data extraction. The detailed algorithmic of automatically extracting a simple table structure from spreadsheets have been presented. Besides, the improvement of spreadsheet table content has been achieved by using the ontology-based system for identified and extracted duplicated data. Also, the proposed approach facilitates the missing values inside table content in the spreadsheet to transform from low-quality semi-structured data to a high-quality relational database. The proposed approach generated only the foreign keys inside the database. As future work, we attempt to add different keys generation such as compound keys or composite keys. Also, we aim to extract complex table structures from spreadsheets and improve performance accuracy.

References

1. Paramonov, Viacheslav, Alexey Shigarov, and Varvara Vetrova. "Table Header Correction Algorithm Based on Heuristics for Improving Spreadsheet Data Extraction." International Conference on Information and Software Technologies. Springer, Cham, 2020.

2. Amin, Ahmed E. "Building Intelligent Semantic Educational System (ISES) Based on Ontology and Semantic Web Mining." *International Journal of Intelligent Computing and Information Sciences* 19.1: 38-49 (2019)
3. Elhefny, M., et al. "FOORC: A FUZZY ONTOLOGY-BASED REPRESENTATION FOR OBESITY RELATED CANCER KNOWLEDGE." *International Journal of Intelligent Computing and Information Sciences* 16.3: 15-36. (2016)
4. Amin, Ahmed E. "An Intelligent Synchronous E-Learning Management System Based on Multi-Agents of Linked Data, Ontology, and Semantic Service." *International Journal of Intelligent Computing and Information Sciences* 19.1: 25-37. (2019)
5. FaDI, Dalia, Safiaa Abas, and Mostafa Aref. "An Approach for Automatic Arabic Ontology Generation." *International Journal of Intelligent Computing and Information Sciences* 17.3: 1-10. (2017)
6. Glance, N. U.S. Patent Application No. 16/247,523. (2019).
7. Zanibbi, R., Blostein, D., & Cordy, J. R. A survey of table recognition. *Document Analysis and Recognition*, 7(1), 1-16. (2004).
8. Embley, D.W., Krishnamoorthy, M.S., Nagy, G., Seth, S.: Converting heterogeneous statistical tables on the web to searchable databases. *Int. J. Doc. Anal. Recogn.* 19, 1–20 (2016)
9. Galkin, M., Mouromtsev, D., Auer, S.: Identifying web tables: Supporting a neglected type of content on the web. In: *Proceedings of the 6th International Conference Knowledge Engineering and Semantic Web, Moscow, Russia. Communications in Computer and Information Science*, vol. 518, pp. 48–62 (2015)
10. Koci, E., Thiele, M., Lehner, W., & Romero, O. Table recognition in spreadsheets via a graph representation. In *2018 13th IAPR International Workshop on Document Analysis Systems (DAS)* (pp. 139-144). IEEE. (2018, April).
11. Shigarov, A.: Table understanding using a rule engine. *Expert Syst. Appl.* 42(2), 929–937 (2015).
12. Shigarov, A.: Rule-based table analysis and interpretation. In: *Proceedings of the 21st International Conference on Information and Software Technologies. Communications in Computer and Information Science*, vol. 538, pp. 175–186 (2015)
13. M. Spence, C. Beilken, and T. Berlage. Focus: The interactive table for product comparison and selection. In *UIST*, pages 41{50}, (1996).
14. Astrakhantsev, N., Turdakov, D., Vassilieva, N.: Semi-automatic data extraction from tables. In: *Selected Papers of the 15th All-Russian Scientific Conference on Digital Libraries: Advanced Methods and Technologies, Digital Collections*, pp. 14–20 (2013)
15. P. J. Guo, S. Kandel, J. M. Hellerstein, and J. Heer. Proactive wrangling: Mixed-initiative end-user programming of data transformation scripts. In *UIST*, pages 65{74}, 2011.
16. S. Kandel, A. Paepcke, J. Hellerstein, and J. Heer. Wrangler: interactive visual specification of data transformation scripts. In *CHI*, pages 3363{3372}, 2011.
17. Chen, Z., Cafarella, M.: Automatic web spreadsheet data extraction. In: *Proceedings 3rd International Workshop on Semantic Search over the Web*, pp. 1: 1–1: 8. ACM, New York, NY, USA (2013).
18. Haoyu Dong, Shijie Liu, Shi Han, Zhouyu Fu, and Dongmei Zhang. Tablesense: Spreadsheet table detection with convolutional neural networks. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2019.
19. Elvis Koci, Maik Thiele, Óscar Romero Moral, and Wolfgang Lehner. A machine learning approach for layout inference in spreadsheets. In *IC3K 2016: Proceedings of the 8th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management: volume 1: KDIR*, pages 77–88. SciTePress, 2016.

20. Zhe Chen and Michael Cafarella. Integrating spreadsheet data via accurate and low-effort extraction. In Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 1126–1135. ACM, 2014.
21. GULWANI, Sumit, et al. Extracting relational data from semi-structured spreadsheets. U.S. Patent Application No 14/044,063, 2015.
22. Pinto, David, et al. "Table extraction using conditional random fields." Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval. ACM, 2003.
23. Bhavik Doshi, Handling Missing Values in Data Mining. Data Cleaning and Preparation Term Paper.
24. Banks, Barbara Jane, Rajnish Kumar Chitkara, and Shiping Chen. "Protection of encryption keys in a database." U.S. Patent No. 9,158,933. 13 Oct. 2015.
25. https://data.gov.au/data/dataset?organization=australianpublicservicecommission&res_format=excel+%28.xlsx%29&_res_format_limit=0