Indonesian Journal of Electrical Engineering and Computer Science
Vol. 26, No. 3, June 2022, pp. 1520~1529
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v26.i3.pp1520-1529

An approach for metadata extraction and transformation for various data sources using R programming language

Forat Falih Hasan[1,2], Muhamad Shahbani Abu Bakar[1]
[1]School of Computing, Universiti Utara Malaysia, Sintok, Malaysia
[2]Department of Computer Engineering Techniques, College of Engineering Technology, Alkitab University, Kirkuk, Iraq

Article Info

Article history:

Received Oct 20, 2021
Revised Mar 20, 2022
Accepted Apr 2, 2022

Keywords:

Data transformation
Information systems
Metadata information
Non-relational databases
SABR algorithm

ABSTRACT

The metadata system is the key system for sharing and transforming data between various information systems (IS), and each database system has its own structure for storing and retrieving metadata information. Metadata information must be extracted for data transformation. Furthermore, these procedures were required to communicate with each type of database system and retrieve the stored metadata; these processes required much information and effort. To overcome the challenge of accessing and extracting metadata from any type of data source, a uniformed method must be developed and integrated into any organization's information systems. The semi-structured data extraction method (SeDEM) is a developed method that includes three main operations logical structure operation, unique key operation, and relationships operation. Finally, the accurate information obtained using the SeDEM addressed data quality issues concerning the integrity and completeness of the data transformation.

Corresponding Author:

Forat Falih Hasan
School of Computing, Universiti Utara Malaysia
Sintok, 06010 Bukit Kayu Hitam, Kedah, Malaysia
Email: forat.db@gmail.com, forat@uoalkitab.edu.iq

1. INTRODUCTION

Many organizations and businesses now have their own information management system. Metadata, which can be defined as data that describes other data, is the key for sharing and transforming data between various information systems. The primary goal of metadata is to provide structured information that can be used to explain, manage, and describe information sources. Furthermore, this term correct and effective design and management contribute to the overall efficiency of any organization information systems [1]–[4]. The metadata of a database describes the schema as well as all other stored data in a database system, such as table details, data types, constraints, and relationships. Metadata is essential for database reporting, query creation, and data transformation [5]–[9].

Notably, many tools, such as cross-database studio, database-bridge, and data-management center, are available for re-engineering and data transformation between various database systems. All of mentioned tools are based on the metadata stored in the source databases [10]–[12]. In light of this, extract, transform, load (ETL) refers to the process of extracting metadata information from source databases, transforming the obtained data, and loading it to the target system. It can also be defined as data transformation processes between source and target databases [13]–[16]. Furthermore, the structure and access methods for metadata information differ between different types of database systems; for example, in relational databases, in the Oracle system, there are two locations used to store metadata information called data dictionary and metadata-registry, which built-in-views can access. The number of these views varies by Oracle version; the main built-in-views are "all objects", "all tables", and "all views" [17], [18]. Furthermore, in structured query language (MySQL), a specific database called "information schema" is used to store metadata information related to all stored databases in a MySQL instance. This database contains a large number of read-only tables that view but not base tables. Therefore, just the data can be readable without any operations such as delete, insert, or update. All of the stored data in this database can be viewed using SQL commands; many tables related to various functions are stored in this category, including "information schema columns", "information schema tables", and "information schema. Key column usage". In addition, there is no metadata to describe the stored data in the semi-structured spreadsheet [19], [20]. Metadata in a database system efficiently describes the stored data. Many approaches to sharing and transforming data from one system to another have been developed, and almost all of these approaches are based on the stored metadata in source systems [3]. Basically, researchers [21]–[24] developed methods for transforming data between two relational databases. These approaches were based on the concept of extracting the structure of the source database from the stored metadata, then using the metadata information to generate the structures of the database objects in the target database using the developed methods. Once the structures of the database objects in the target database have been successfully created, the final step is to move the data from the source to the defined locations in the target database. These methods have provided effective solutions for transferring and transforming data between two relational databases. However, the structure of all of these methods is built on the metadata information stored in the source database. As a result, tracking and analyzing each metadata structure in various relational databases is difficult. It is worth noting that the following method differs from the previous approaches in that it uses the source metadata to design a data repository. Xiao *et al.* [25] presented a methodology for transforming databases from the current system to the target data warehousing using metadata stored in the source database. This method contributed to the development of a framework for constructing an organization repository using stored metadata information as a guide, which can support both relational and non-relational information systems. The data transformation processes from the source database to the target data repository are primarily determined using this method's extracted metadata. Moreover, the developed system must be updated on a regular basis in accordance with the structure of the information sources, and these are potentially difficult procedures to analyze each system metadata.

In the case of system redesigning and query management, the following two methods made use of the term metadata. Based on the stored metadata, Ristić *et al* [11] proposed a method for reverse database engineering. This method contributes to improving information systems by redesigning the storage of existing relational databases using metadata. According to this method, the old system's metadata must be extracted by users, and the new system must be assigned by users, with no flexibility for auto extraction of the stored metadata. This method used comparisons based on stored data and stored metadata data to redraw the steps of reverse engineering processes. Finally, Anitha and Mukherjee [26] developed a methodology for managing database queries that made use of the stored metadata. The developed framework is used for efficient data retrieval from cloud databases. Based on the processes of this method, the system begins by reading the user's query and then investigates the stored database metadata to reduce the time spent searching for and retrieving information. This method provided a good solution for dealing with queries. Each type of database system must design processes for analyzing and extracting information from stored metadata, and these processes differ from one system to another. Furthermore, it isn't easy to be interacting with each type of data source system and efficiently extract metadata information.

In this paper, we propose a uniformed method for obtaining metadata information from various types of data sources systems based on the R programming language. The first step in this method deals with scanning the entire data in the selected data source, then generating accurate metadata information and the relationship between tables based on the developed algorithms. The following are the main contributions of current research: a standardized method for obtaining metadata information that can be used with all types of data source systems. The developed method reduces the amount of time, the number of applications, and the effort required for metadata management. Finally, it improves an organization's information systems by sharing data, migrating data, and upgrading systems. This research is structured as follows: The developed methodology and method are discussed in the second section, and the developed method is tested with the chosen case study. Section three contains the results with discussions, and section four includes the conclusion and future work.

2. THE PROPOSED METHOD

This study proposed work is concerned with the extraction of metadata using the R programming language. In general, the semi-structured data extraction method (SeDEM) consists of three steps: the first involves connecting to a selected data source; the second step is used to extract the required data from the selected data source. The third step entails generating metadata for the extracted data. In SeDEM, all data from relational, non-relational, and semi-structured spreadsheets are stored in the R environment and considered the source system. It is worth mentioning that after the extraction processes, each dataset will be saved in the R environment, and each file will have the R program extension that ends with the character "filename R". All tables in the R environment are semi-structured, and there is no metadata to describe this dataset. In addition, the SeDEM was developed to deal with this type of data and generate metadata for any desired dataset from any source within the R environment. The second step was carried out after the data had been transformed and saved in R table format. The developed algorithms in the R environment proposed in this step to extract metadata information from the stored data and save it in a predefined location. Finally, the step includes displaying information that explains the logical structure of the stored data (the data describe other data).

The general processes in this method begin with connecting to the desired data source, followed by extracting and saving the required data within the R environment. The obtained data are saved as a table within the R system. Essentially, the R program stores each dataset as an R data frame with a unique name. Furthermore, three types of algorithms exist in SeDEM and are presented as follows: The first algorithm, "Algorithm.1", is used to connect with the desired source system. The second algorithm, "Algorithm.2", is used to extract and load the desired dataset from the source system into the R environment. The third algorithm, "Algorithm.3", deals with logical structure extraction (metadata extraction) for the predefined dataset and includes three sub-algorithms, "Algorithm 3.1", "Algorithm 3.2", and "Algorithm 3.3". Figure 1 depicts the general processes of the developed method.

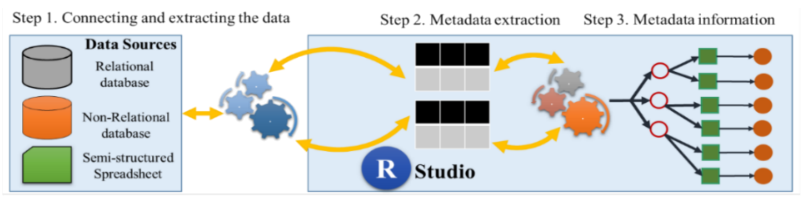
Step 1. Connecting and extracting the data
Step 2. Metadata extraction
Step 3. Metadata information
Data Sources
Relational database
Non-Relational database
Semi-structured Spreadsheet
R Studio

Figure 1. The general processes of the developed method

2.1. Datasets extracting

The data extraction method is the process of obtaining the necessary data from a predefined data source. The first algorithm, as previously stated, was used to generate the connection name with the predefined data source. Furthermore, the SABR algorithm based on R programming was used with all of the semi-structured tables to set up the connection and create the connection name [27]. The second algorithm used the connection name from the first algorithm to extract the required data from the predefined data source. Each source system has its own set of properties and models for storing and managing data. The data in relational databases is stored in a relational format in a table, and the table is organized into columns. In light of this, relationships exist between database tables in relational databases. There are no relationships between the data in non-relational databases, and the data is mostly stored in a collection [28], [29]. Each defined dataset in R studio is saved in table format in the R environment [27]. The table is a data structure that is organized into attributes and rows [30]. The second algorithm will transform each selected data source to the R environment. As a result, the first step in the second algorithm is to determine the names of the necessary datasets from the source system. In the second step, each dataset will be chosen based on its assigned name. The selected dataset will be transformed to the R environment and saved in the table based on the R data format. Furthermore, each table will be treated in R as columns and rows (R data frame), if there is more than one table in the R environment, each table is isolated, and there is no relationship between these objects.

2.2. Metadata extraction

All datasets extracted and saved from source systems into the R environment will be saved as semi-structured tables in the R data frame. As a result, there is no metadata for this type of object. The metadata data generation processes are based on the stored data of each table in the R environment. The R studio can provide a variety of functions for reading the data in each table. Moreover, the third algorithm was created in order to generate metadata for the desired datasets. All defined and undefined operations can be applied to the extracted data at this stage [31]. This work has developed a set of operations to generate metadata information from any chosen dataset. Furthermore, to support various functions, all operations are based on the concept of a relational database [32], [33] and the codes are written based on the R programming language. The first operation is also the logical structure operation. We will perform this operation if we need to extract the logical structure information for the assigned tables. This operation categorizes the attributes of group tables based on the criteria specified and saves the results in "Table 1". Then, based on this operation, we can begin to address the following issue: "Given a logical structure definition for each attribute in each table". In addition, each attribute logical structure information contains the following characteristics:

– Name $\in$ {The column name in the original table}
– Ordinal-position $\in$ {The column position in the original table}
– Data-Type $\in$ {Number, Varchar, Date}
– Size of data type $\in$ {Number = 30, Varchar = 40}
– Attribute null able $\in$ {Yes, No}

To clear the above rules, Algorithm 3.1 is invoked, which is used to obtain "Table_1" that includes the table name, column name, ordinal position, data type, and maximum length. Furthermore, Algorithm 3.1 was used to assign the nullable attribute feature. Algorithm 3.1 was divided into two parts to provide a clear picture. The first part is to assign the nullable feature to the datasets. The second part addresses attribute keys and relationships. Notably, Algorithm 3.2 was developed to obtain information about the attribute key constraint to extend the work of Algorithm 3.1. The unique key operation occurs when we need to test and register whether or not the column contains duplicated values. The workflow of this operation begins by selecting the attribute and comparing the number of null values, all tuples, and distinct tuples for the same attribute, and all the results of Algorithm 3.2 will be saved in "Table 2". We address the problem of "identifying the unique table keys" using this operation. In this study, each attribute that is considered a unique key must adhere to the following rules:

– Rule1: no duplicates in tuples.
– Rule2: accept only one null value.
– Rule3: the number of distinct tuples is equal to number of all tuples.
– Rule4: in case of one null value, the number of distinct tuples is equal to number of all tuples minus one.
– Rule5: each table can hold more than one unique key.

Based on the above results of Algorithm 3.2, the operation of the relationship can be identified using the developed Algorithm 3.3. This type of operation occurred in case we needed to extract the relationships between a group of tables. In this operation, the conversion criteria are defined in order to extract the relationship between various assigned tables. The current operation processes start selecting an attribute from one table and make the comparisons with other attributes saved in other tables. All the results of Algorithm 3.3 will be saved in "Table 3". Based on this operation, we address the problem of: "Identifying the tables relationships". In this research, to extract the relationship between the defined tables, the following rules are used for that purpose:

– Rule6: the data types must be the same in the connected attributes.
– Rule7: no null value.
– Rule8: can accept the duplicate tuples.
– Rule9: each attribute is unique and does not include null value and has a relationship with any attribute is called the primary key.
– Rule10: each attribute is not unique and does not include null value and has a relationship with any attribute is called the foreign key.

The pseudocode of Algorithm.1, Algorithm 2, Algorithm 3.1, Algorithm 3.2, and Algorithm 3.3 based on R programming language are presented in the next section.

2.3. Modelling the developed method with the real case study

To validate the developed method and its algorithms, we use a connection to a relational database as an example. MySQL is a relational database management system for managing and storing relational data. The proposed example in this study deals with tables stored in the MySQL database. This database is known as the human resources (HR) database and is used to store and manage human resources information. It consists of seven tables with various types of relationships between them [34]. In light of this, to verify the first Algorithm 1 in the developed SeDEM method, the following pseudocode was used to set up the connection with the MySQL database based on the R programming language.

As a result of the preceding code, the connection name is regarded as the gateway to accessing the predefined data source and meeting the data transformation requirements. The next step is to deal with the extracted desired datasets from the connected data source. Furthermore, the list of the tables required to be extracted from the MySQL database needs to be assigned, as shown in the Algorithm 2.

All of the tables assigned in the second algorithm will be extracted and saved as an R data frame in the R environment. The seven tables are extracted as a result of the second algorithm and used in the remainder of this research to validate the developed method and generate metadata information from the selected data source. In addition, Algorithm 3.1 was used to generate the logical structure information for each table's attributes, as shown in section 2.2.

As a result of Algorithm 3.1, all of the columns in each table are extracted and collected in a single table called "Table_1". From the table name, we assign the source of the attributes and attribute ordinal position. All the other information related to data types, such as the size of data and null-able features, will be extracted from analyzing the stored data in each attribute. Table 1 displays the results of Algorithm 3.1.

Algorithm 3.2 was added to determine the unique key constraints to extend the functionality of Algorithm 3.1. Algorithm 3.2 work is based on both the original tables stored as R data frames and the output of Algorithm 3.1 "Table_1". As a result of this algorithm, all the information coming from the unique key of each attribute are extracted and registered. The processes of this step are functionally based on the original tables and the output of the Algorithm 3.1. Table 2 displays the results of Algorithm 3.2.

Algorithm 1. Database source connection
Algorithm: MySQL connection based on R programming
Input: Database name, MSQL
Output: Connection name "MSQL"
Variables: Connection_name

```
begin
set "MSQL" is the connection name
username ="root"
password = "MySQL2020"
host = "127.0.0.1"
/ * host based on the local connection */
loading (DBI) library
loading (RMySQL) library
MSQL = starts to connecting based on (username, password, host)
/ * "MYSQL" the output of current algorithm */
End
```

Algorithm 2. Datasets extracting
Algorithm: Extracting the objects of the connected data source
Input: Connection name, tables names
Output: Seven tables (countries, departments, employees, jobs, job history, locations, regions) extracted and saved in R environment

```
begin
use the "MSQL" connection name
/ * host based on the local connection */
loading (DBI) library
loading (RMySQL) library
MSQL = Extract (countries, departments, employees, jobs, job_history, locations, regions)
Save the tables in R environment
End
```

Algorithm 3.1. Datasets analyzing
Algorithm: Logical structure generating
Input: List of tables names
Output: Attributes properties (Table_1)
Variables: i, j

```
begin
i = {all the tuples ϵ" List of tables names "}
/ * List of tables in this research from the second algorithm*/
j = {number of columns for each table ϵ (table i)}
for all the tuples ϵ (table i)
/ * Select the first table till the end of the tables list */
 for all the columns ϵ j
/ * Select the first column till the last column in the current table i */
read the column properties for (table i)
register table name
extract & register column name (table i)
extract & register the ordinal position (table i)
extract & register column data types (table i)
extract & register maximum length (table i)
extract & register if the column nullable or not (table i)
save all the results in Table_1
 next j / * Next column in current (table i) */
next i / * Next table in List of tables */
Report ← get message (Generate the final Table_1)
end
```

Table 1. The result of Algorithm 3.1 “Table_1”

No	Table_name	Column_name	Ordinal_position	Data_type	Maximum_length	Is_nullable
1	countries	country_id	1	Varchar	40	No
2	countries	country_name	2	Varchar	40	No
3	countries	region_id	3	Int	30	No
4	departments	department_id	1	Int	30	No
5	departments	department_name	2	varchar	40	No
6	departments	manager_id	3	int	30	Yes
7	departments	location_id	4	int	30	No
8	employees	employee_id	1	int	30	No
9	employees	first_name	2	varchar	40	No
10	employees	last_name	3	varchar	40	No
11	employees	email	4	varchar	40	No
12	employees	phone_number	5	varchar	40	No
13	employees	hire_date	6	date	NA	No
14	employees	job_id	7	varchar	40	No
15	employees	salary	8	int	30	No
16	employees	commission_pct	9	int	30	Yes
17	employees	manager_id	10	int	30	Yes
18	employees	department_id	11	int	30	Yes
19	jobs	job_id	1	varchar	40	No
20	jobs	job_title	2	varchar	40	No
21	jobs	min_salary	3	int	30	No
22	jobs	max_salary	4	int	30	No
23	job_history	employee_id	1	int	30	No
24	job_history	start_date	2	date	NA	No
25	job_history	end_date	3	date	NA	No
26	job_history	job_id	4	varchar	40	No
27	job_history	department_id	5	int	30	No
28	locations	location_id	1	int	30	No
29	locations	street_address	2	varchar	40	No
30	locations	postal_code	3	varchar	40	Yes
31	locations	city	4	varchar	40	No
32	locations	state_province	5	varchar	40	Yes
33	locations	country_id	6	varchar	40	No
34	regions	region_id	1	int	30	No
35	regions	region_name	2	varchar	40	No

Algorithm 3.2. Unique key
Algorithm: Unique key information extraction
Input: Table_1
Output: Attributes unique key information (Table_2)
Variables: i, Table_1, p1, p2, p3, p4, p5, p6

```
1.  begin
2.  i = {all the tuples ϵ " Table_1 "}
3.  for all the tuples ϵ (table i)
4.   P1 = Table_1 (i, 1)
5.   / * p1 Used to record table name*/
6.   P2 = Table_1 (i, 2)
7.   / * p2 Used to record column name*/
8.   P3 = Table_1 (i, 6)
9.   / * p3 Used to record if the column is null or not*/
10.  P4 = select the number of the alltable tuples of the current table based on (p1)
11.  P5 = select the number of all the distinct table tuples of the current table based on
    (p1)
12.  P6 = p4 -1
13.  If ((p4 = p5) & (p6 = p4)) then
14.  Column_Unique = "YES"
15.  Else
16.  Column_Unique = "NO"
17.  End if
18.  Tabe_2 = (p1, p2, p3, Column_Unique)
19. next i
20. Report ← get message (Generate the final Table_2)
21. End
```

Notably, we assigned the rules (rules 6–10) to identify the table relationships mentioned in section 2.2 for Algorithm 3.3 in order to generate information about the relationships between the attributes of a group of tables. The Algorithm 3.3 processes are based on both the results of Algorithms 3.1 "Table_1" and 3.2 "Table_2," as well as original tables stored in the R environment as R data frames. As a result of this algorithm, all the relationships between tables attributes are extracted and registered. This step act based on the original tables, and the output of Algorithm 3.1 with Algorithm 3.2 as shown in Table 3, displays the outcomes of Algorithm 3.3 (In Appendix).

Table 2. The result of Algorithm 3.2 “Table_2”

No	Table_name	Column_name	Column_Null	Column_Unique
1	countries	country_id	No	YES
2	countries	country_name	No	YES
3	countries	region_id	No	NO
4	departments	department_id	No	YES
5	departments	department_name	No	YES
6	departments	manager_id	Yes	NO
7	departments	location_id	No	NO
8	employees	employee_id	No	YES
9	employees	first_name	No	NO
10	employees	last_name	No	NO
11	employees	email	No	YES
12	employees	phone_number	No	YES
13	employees	hire_date	No	NO
14	employees	job_id	No	NO
15	employees	salary	No	NO
16	employees	commission_pct	Yes	NO
17	employees	manager_id	Yes	NO
18	employees	department_id	Yes	NO
19	jobs	job_id	No	YES
20	jobs	job_title	No	YES
21	jobs	min_salary	No	NO
22	jobs	max_salary	No	NO
23	job_history	employee_id	No	NO
24	job_history	start_date	No	NO
25	job_history	end_date	No	NO
26	job_history	job_id	No	NO
27	job_history	department_id	No	NO
28	locations	location_id	No	YES
29	locations	street_address	No	YES
30	locations	postal_code	Yes	NO
31	locations	city	No	YES
32	locations	state_province	Yes	NO
33	locations	country_id	No	NO
34	regions	region_id	No	YES
35	regions	region_name	No	YES

Table 3. The result of Algorithm 3.3 “Table_3”

Table_ name_1 (P3)	Column_ name_1 (P5)	Data_ type_1 (P1)	Column_ unique_1 (P8)	Key_1 (P10)	Table_ name_2 (P4)	Column_ name_2 (P6)	Data_ type_2 (P2)	Column_ unique_2 (P9)	Key_2 (P11)
employees	employee_id	int	YES	P.K	departments	manager_id	int	NO	F.K
departments	department_id	int	YES	P.K	employees	department_id	int	NO	F.K
departments	location_id	int	NO	F.K	locations	location_id	int	YES	P.K
employees	employee_id	int	YES	P.K	employees	manager_id	int	NO	F.K
employees	job_id	varchar	NO	F.K	jobs	job_id	varchar	YES	P.K
job_history	employee_id	int	NO	F.K	employees	employee_id	int	YES	P.K
job_history	job_id	varchar	NO	F.K	jobs	job_id	varchar	YES	P.K
job_history	department_id	int	NO	F.K	departments	department_id	int	YES	P.K
locations	country_id	varchar	NO	F.K	countries	country_id	varchar	YES	P.K
regions	region_id	int	YES	P.K	countries	region_id	int	NO	F.K

3. RESULTS AND DISCUSSION

In this reseach, the adaptable method for dealing with various data storage systems has been presented. The SeDEM is used in conjunction with multiple data sources to generate metadata for single or multiple datasets. Experiments in the chosen case study confirmed that SeDEM improved the accuracy of describing storage data. Furthermore, the SeDEM's accurate information increases the integrity of data sharing between various information systems. Moreover, this accuracy leads to the completeness of data transformation from one system to another. According to the comparison of three parts, it can be deduced that the first part is the original database system, which is used in the research case study. The second component is the R programming data frame, which uses the str() function to explain and provide information for any dataset in the R environment. The second section describes the developed method (SeDEM) in this study. The SeDEM results are shown in the Table 4.

Table 4. The obtained results by the SeDEM

No	The criteria	HR MySQL DB the original system	R data frame str()	SeDEM the developed method
1	The number of text value attributes	17	20	17
2	The number of numeric value attributes	15	15	15
3	The number of date value attributes	3	0	3
4	The numbers of primary keys attributes	6	0	6
5	The numbers of foreign keys attributes	10	0	10
6	The number of attributes that can accept null values	14	6	6
7	The number of relationships among the tables	10	0	10

4. CONCLUSION

This study developed a uniform method for generating metadata information from a variety of assigned source systems. Furthermore, the SeDEM-developed algorithms assisted in reducing the difficulties and time required to track each database system and obtain metadata information. Moreover, SeDEM can generate metadata information for data stored in a semi-structured file that lacks the ability to describe its contents. The developed method can extract the logical database structures, attribute constraints, and the relationship between all the tables' attributes from any selected datasets. All of the developed algorithms are written in the R programming language, which allows for greater flexibility in data analysis and support for various data visualization processes. The experimental results show that the SeDEM can improve information systems in any organization by providing a consistent method for extracting metadata data to assist in data transformation between different information systems. Furthermore, the SeDEM's accuracy in reading and defining data is equal or greater than the original system in the database system and greater than the R programming data frame data. This accuracy leads to data integrity and completeness, allowing data to be shared among various information systems. In the future, features for transforming extracted metadata and stored data to multiple database systems (SQL or NoSQL) can be added to SeDEM, extending the functionality of this developed method.

APPENDIX

Algorithm 3.3. Attributes relationships
Algorithm: Attributes relationships information extraction
Input: List of tables names, Table_1, Table_2
Output: Attributes relationships (Table_3)
Variables: m, n, Table_1, Table_2, p1, p2, p3, p4, p5, p6, p7, p8, p9, p10, p11, R1, R2

```
1.  begin
2.  m = {all the tuples ϵ " Table_1 "}
3.  n = {all the tuples ϵ " Table_1 "}
4.  for all the tuples ϵ (table m)
5.  for all the tuples ϵ (table n)
6.  P1 = Table_1 (m, 4)
7.  P2 = Table_1 (n, 4)
8.  / * p1 & p2 Used to record data types based on table_1*/
9.  P3 = Table_1 (m, 1)
10. P4 = Table_1 (n, 1)
11.  / * p3 & p4 Used to record tables names based on table_1*/
12. P5 = Table_1 (m, 2)
13. P6 = Table_1 (n, 2)
14.  / * p5 & p6 Used to record columns names based on table_1*/
15. P7 = Table_1 (m, 6)
16. / * p7 Used for null value comparison*/
17. R1 = The P5 holds any one of the current characters in column name (id or id or code or
    codes)
18. R2 = The P6 holds any one of the current characters in column name (id or id or code or
    codes)
19. / *R1 and R2 extendable by the user to unlimited words*/
20. If_1 ((p1= p2) & (R1 is True) & (R2 is True) & (p7 = 'NO')) then
21. / *The First level comparison*/
22. If_2 (if the content value of attribute P6 in P5)
23. / *To check if the data in P6 is in the P5*/
```

```
24. P8 = (Select column_unique from Table_2 where table_name = (p3) and column_name = (p5))
25. P9 = (Select column_unique from Table_2 where table_name = (p4) and column_name = (p6))
26. If_3 (p8 = 'yes') then P10 = 'P.K'
27. Else P10 = 'F.K'
28. End if_3
29. If_4 (p9 = 'yes') then P11 = 'P.K'
30. Else P11 = 'F.K'
31. End if_4
32. If_5 (p1=p2) & (p3=p4) & (p5=p6) & (p10=p11) & (p1=p2) then print "Error"
33. Else
34. Add to Table_3 the following new rows (p3, p5, p1, p8, p10, p4, p6, p2, p9, p11)
35. End if_5
36. End if_2
37. End if_1
38. next n
39. next m
40. Filter the duplicate records in Table 3
41. Report ← get message (Generate the final Table_3)
42. end
```

REFERENCES

[1] I. Cverdelj-Fogaraši, G. Sladić, S. Gostojić, M. Segedinac, and B. Milosavljević, "Semantic integration of enterprise information systems using meta-metadata ontology," *Information Systems and e-Business Management*, vol. 15, no. 2, pp. 257–304, 2017, doi: 10.1007/s10257-015-0303-6.

[2] Y. Wang, J. Le, and D. Huang, "A metadata management framework for marine information based on XML," *IET Conference Publications*, vol. 2012, no. 636 CP, 2012, doi: 10.1049/cp.2012.2258.

[3] J. Zheng and X. Li, "Research and application of data modeling and integration based on metadata," in *2015 7th International Conference on Information Technology in Medicine and Education (ITME)*, Nov. 2015, pp. 525–528, doi: 10.1109/ITME.2015.160.

[4] S. Q. A. Al-Rahman, S. A. Jassim, and A. M. Sagheer, "Design a mobile application for vehicles managing of a transportation issue," *Bulletin of Electrical Engineering and Informatics*, vol. 10, no. 4, pp. 2263–2272, 2021, doi: 10.11591/EEI.V10I4.2918.

[5] A. Aspin, *Metadata*. 2012.

[6] M. R. Kogalovsky, "Metadata in computer systems," *Programming and Computer Software*, vol. 39, no. 4, pp. 182–193, 2013, doi: 10.1134/S0361768813040038.

[7] A. Nabli, S. Bouaziz, R. Yangui, and F. Gargouri, "Two-ETL phases for data warehouse creation: design and implementation," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9282, pp. 138–150, 2015, doi: 10.1007/978-3-319-23135-8_10.

[8] F. F. Hasan, "A Review study of information systems," *International Journal of Computer Applications*, vol. 179, no. 18, pp. 15–19, 2018.

[9] G. W. Sasmito, D. S. Wibowo, and D. Dairoh, "Implementation of rapid application development method in the development of geographic information systems of industrial centers," *Journal of Information and Communication Convergence Engineering*, vol. 18, no. 3, pp. 194–200, 2020, doi: 10.6109/jicce.2020.18.3.194.

[10] B. Walek and C. Klimes, "A methodology for data migration between different database management systems," *International Journal of Computer and Information Engineering*, vol. 6, no. 5, pp. 536–541, 2012.

[11] S. Ristić, S. Aleksić, M. Čeliković, V. Dimitrieski, and I. Luković, "Database reverse engineering based on meta-models," *Open Computer Science*, vol. 4, no. 3, pp. 150–159, 2014, doi: 10.2478/s13537-014-0218-1.

[12] K. N. Bhatt, S. Dessai, and V. S. Yerragudi, "Design and development of a parallelized algorithm for face recognition in mobile cloud environment," *International Journal of Reconfigurable and Embedded Systems*, vol. 10, no. 1, pp. 47–55, 2021, doi: 10.11591/ijres.v10.i1.pp47-55.

[13] N. Du, X. Ye, and J. Wang, "A schema aware ETL workflow generator," *Information Systems Frontiers*, vol. 16, no. 3, pp. 453–471, 2014, doi: 10.1007/s10796-012-9352-2.

[14] N. Biswas, A. Sarkar, and K. C. Mondal, "Efficient incremental loading in ETL processing for real-time data integration," *Innovations in Systems and Software Engineering*, vol. 16, no. 1, pp. 53–61, 2020, doi: 10.1007/s11334-019-00344-4.

[15] Z. A. Jaaz, S. S. Oleiwi, S. A. Sahy, and I. Albarazanchi, "Database techniques for resilient network monitoring and inspection," *Telkomnika (Telecommunication Computing Electronics and Control)*, vol. 18, no. 5, pp. 2412–2420, 2020, doi: 10.12928/TELKOMNIKA.V18I5.14305.

[16] A. H. Al-Hamami and A. A. Flayyih, "Enhancing big data analysis by using map-reduce technique," *Bulletin of Electrical Engineering and Informatics*, vol. 7, no. 1, pp. 113–116, 2018, doi: 10.11591/eei.v7i1.895.

[17] Y. V Ravikumar, K. M. Krishnakumar, and N. Basha, *Oracle database upgrade and migration methods*. Berkeley, CA: Apress, 2017.

[18] F. F. Hasan and M. S. A. Bakar, "Data transformation from SQL to NoSQL MongoDB based on R programming language," *ISMSIT 2021 - 5th International Symposium on Multidisciplinary Studies and Innovative Technologies, Proceedings*, pp. 399–403, 2021, doi: 10.1109/ISMSIT52890.2021.9604548.

[19] F. M. Kromann, *Beginning PHP and MySQL*. Berkeley, CA: Apress, 2018.

[20] C. Mahmoudi, F. Aymen, and S. Lassaad, "Smart database concept for Power Management in an electrical vehicle," *International Journal of Power Electronics and Drive Systems (IJPEDS)*, vol. 10, no. 1, p. 160, 2019, doi: 10.11591/ijpeds.v10.i1.pp160-169.

[21] N. M. Muddasir and K. Raghuveer, "Study of methods to achieve near real time ETL," *International Conference on Current Trends in Computer, Electrical, Electronics and Communication, CTCEEC 2017*, pp. 436–441, 2018, doi: 10.1109/CTCEEC.2017.8455002.

[22] P. Retelius and E. B. Persson, *Creating a customizable component based ETL solution for the consumer*. 2021.

[23] M. Radonic and I. Mekterovic, "ETLator - A scripting ETL framework," *2017 40th International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO 2017 - Proceedings*, pp. 1349–1354, 2017, doi: 10.23919/MIPRO.2017.7973632.

[24] B. Pan, G. Zhang, and X. Qin, "Design and realization of an ETL method in business intelligence project," *2018 3rd IEEE International Conference on Cloud Computing and Big Data Analysis, ICCCBDA 2018*, pp. 275–279, 2018, doi: 10.1109/ICCCBDA.2018.8386526.

[25] B. Xiao, C. Zhang, Y. Mao, and G. Qian, "Review and exploration of metadata management in data warehouse," *Proceedings of the 2015 10th IEEE Conference on Industrial Electronics and Applications, ICIEA 2015*, pp. 928–933, 2015, doi: 10.1109/ICIEA.2015.7334243.

[26] R. Anitha and S. Mukherjee, "'MaaS': fast retrieval of E-file in cloud using metadata as a service," *Journal of Intelligent Manufacturing*, vol. 28, no. 8, pp. 1871–1891, 2017, doi: 10.1007/s10845-015-1076-y.

[27] F. F. Hasan and Z. Jamaluddin, "An optimised method for fetching and transforming survey data based on SQL and R programming language," *Baghdad Science Journal*, vol. 16, no. 2, pp. 436–444, 2019, doi: 10.21123/bsj.2019.16.2(SI)0436.

[28] J. Yang, J. Ma, S. K. Howard, M. Ciao, and R. Srikhanta, "A big data analytic framework for investigating streaming educational data," *ACM International Conference Proceeding Series*, pp. 1–4, 2017, doi: 10.1145/3014812.3014869.

[29] R. P. Padhy, M. R. Patra, and S. C. Satapathy, "RDBMS to NoSQL: Reviewing Some Next-Generation Non-Relational Database's," *International Journal of Advanced Engineering Sciences and Technologies*, vol. 11, no. 11, pp. 15–30, 2011.

[30] S. W. Ambler, *Mapping objects to relational databases*. 2000.

[31] A. David, A. J. Saikat, and M. D. A. James, *The RMySQL package*. 2007.

[32] H. K. Dasararaju and P. Taori, "Data management—relational database systems (RDBMS)," *International Series in Operations Research and Management Science*, vol. 264, pp. 41–69, 2019, doi: 10.1007/978-3-319-68837-4_3.

[33] F. F. Hasan and M. S. Abu Bakar, "An Approach for Data Transformation in Homogeneous and Heterogeneous Information Systems," *HORA 2021 - 3rd International Congress on Human-Computer Interaction, Optimization and Robotic Applications, Proceedings*, 2021, doi: 10.1109/HORA52670.2021.9461287.

[34] A. Ciobanu, "No Title," *hr-schema-mysql. 2021*, 2021. https://github.com/nomemory/hr-schema-mysql/blob/0c3c8f322e607c5249de8adb8e43c0c08351d47c/hr-schema-mysql.sql (accessed Oct. 22, 2021).

BIOGRAPHIES OF AUTHOR

Forat Falih Hasan was born in Kirkuk, Iraq, in 1986. He received the B. Sc. in Manage. Information Systems in 2010, Master Degree in (Information Technology) from (IEC College Of Engineering and Technology/Mahamaya Technical University)-India in 2012, and now pursuing Ph.D. in (Information Technology) from the School of Computing, Universiti Utara Malaysia (UUM). His research interests include Information Systems, MIS, Database Systems, Big Data, Data warehouses, IoT, Data Quality, and Business Intelligence. He can be contacted at email: forat.db@gmail.com.

Muhamad Shahbani Abu Bakar received the Ph. D in Computer Science (Software Engineering), M. Sc (Information Technology) and B. Sc. Computer Science in 1990, 1999 and 2009, respectively. Currently, he is an Associate Professor in School of Computing, Universiti Utara Malaysia. After working as an analyst programmer and system analyst (from 1990- 2000) in private and government sector and a senior lecturer (from 2000-2017), he also served as Director of University Teaching and Learning, Universiti Utara Malaysia since 2018. His research interest includes Software Engineering, Big Data, Cloud Computing, Learning Analytic, Educational Technology, Data Warehouse, and Business Intelligence. He can be contacted at email: shahbani@uum.edu.my.