EAI Endorsed Transactions
on Scalable Information Systems
Research Article EAI.EU

Optimised Transformation Algorithm For Hadoop Data Loading in Web ETL Framework

Gaurav Gupta[1]*, Neelesh Kumar[2] and Indu Chhabra[3]

[1] Research Planning & Project Management, CSIR-Indian Institute of Petroleum, Dehradun, India, gaurav.gupta@iip.res.in
[2] BioMedical Instrumentation, CSIR-Central Scientific Instruments Organisation, Chandigarh, India, neel5278@gmail.com
[3] Department of Computer Science & Applicaions, Panjab University, Chandigarh, India, chhabra_i@rediffmail.com

Abstract

Web ETL unlike conventional ETL framework requires considerable improvements in all the three layers i.e. Extraction, Transformation and Loading due to the inherent nature of web input data. Websites are huge and are unique source of information, out of such huge information available on the websites, finding and analysing the required and relevant data is critical as the data may be foul consisting of redundant data or misspelled. Determining integrated record that stands for identical real world entities in abundant ways is the major problem to be analysed for any database. Hence, Web ETL transformation layer functionality of data transformation becomes mandatory in determining the pertinent information to be examined. Since the data on the web is "very voluminous" hence loading only clean data in data warehouse is necessary for fast processing to achieve accurate result. The present research focuses on data transformation in web ETL framework and proposes a modified technique to employ token wise sentence sorting to remove redundant records from the patent database along with Levenshtein distance used for string matching. Afterwards the cleaned data is transformed and loaded from this staging area to hadoop environment. The integration of proposed transformation technique with hadoop system delimits the constraint of data processing, storage and retrieval of large data structure from conventional data warehouse system.

Keywords: Redundant Data, Data Transformation, Data Loading, Levenshtein Distance Matching, Hadoop

Received on 11 May 2019, accepted on 01 October 2019, published on 02 October 2019

doi: 10.4108/eai.13-7-2018.160600

*Corresponding author. Email:gaurav.gupta@iip.res.in

1. Introduction

Web ETL Framework requires web documents mainly web-pages as input for extraction of data. This existing voluminous data is highly unstructured and heterogeneous due to high expectations and needs of user thus requires pre-treatment through multiple transformation techniques. After discussing the major limitations of the existing scenario, this research has contributed into a modified transformation technique for the data being loaded in the distributed nodes under Hadoop file system. Web documents are enormous source of knowledge and information. These contain information which is used by public, private and government sectors to observe the progress going on their respective domains. Data which is acquired and processed for mining is identified and cleaned before loading it to the data warehouse. Massive amount of data is generated every minute of the day and hence there is copious data present in the world. Data in the databases of the firms and social media is usually in petabyte and zetabyte and obtaining useful data is the real task because without accurate data correct result extraction is a major issue. Hence the need of data transformation arises. Many independent sources of data are merged into a gigantic vault known as data warehouse for querying and analysis purpose [1]. The data present in the staging area can be heterogeneous. Since our focus is on web data which is not based on conceptual schema and therefore explicit use of semantic model is required for web based data

processing [2].There are possibilities that data may not be clean data [3][4] where errors such as spelling mistakes, typographical errors and name variants may present. Also database share different schemas to store information and sometimes multiple datasets are merged which again creates more difficulty to deal with the data [5]. The aim is to remove such duplicity from the database [6][7] . Redundant records removal process works in two ways; initially it identifies the exactness, whether two records are exact copies of each other, if so it detects the redundant records and then removes one copy out of the two to make the data unique and accurate.

To load the data in the data warehouse, data pre-processing is required to discover the knowledge [8]. There are several companies and organizations that spend a lot of money on identification and removal of errors [9]. Handling the correctness of data manually is time consuming and arduous, still there remain chances for errors. Therefore the need of automation in the cleaning process is required. For this purpose research is made to filter the quality data in the data warehouse [10]. Unless the exact and accurate information about redundant data is not known, processing of the data for further analysis cannot be validated due to the production of untrustworthy data. Hence the present work is divided into two parts first, data cleaning and second, to load quality data in the data warehouse.

Data cleaning is the technique for data transformation to scrub the data and deals with missing, incorrect values and removal of replicated data [11] and store it in warehouse for the better analysis. It is required in several fields such as telecommunication, banking, transportation, retailing, IP analysis and social media sites. Data cleaning is a vast term and its definition depends upon the area it is applied, the major domain on which it is applied are data warehousing, data knowledge discovery and data quality management. The aim is to achieve quality data efficiently [12]. It is performed on the data before moving it to data warehouse so that further data mining techniques and queries can be applied on the data set to achieve better and efficient results. The existing techniques of RDBMS sorts data in pre-defined sequence for identifying duplicate tuples in records and therefore depends on word sequence for identification. For example a field name “person_name” consisting of three words for first name, middle name and last name will be dependent on the sequence of order of its name and therefore will consider the same name reappearing in difference sequence with last name, first name, middle name as different although both belongs to same person. The present research is able to identify such dissimilar sequence of words for identification of duplication. Finally, Loading phase is the last step in which cleaned de-duplicated data is transferred into target source in hadoop storage due to its inherent advantages.

In the road map of this research, section 2 explains the related work and section 3 describes the proposed work, section 4 has covered the experimental results and finally section 5 concludes the research outputs.

2. Literature survey

There has been some work done previously on duplicity removal and de-duplication of the records in the database table. The problem of representation of same objects in different forms is known as field matching problem. Buckles et al. [13] have studied fuzzy relational database, to make out alike occurrences of the existences of objects through inexact matching. Fuzzy relational database focuses on the relevant and efficient results by firing a fuzzy query on the fuzzy relational database, but this paper concentrates more on the pre-processing of the data even before it is ready for the query. Although some work is done in field matching but that is only on the specific area. T.F. Smith and M.S. Waterman [14] have proposed algorithm for identification of common molecular subsequence’s for matching DNA and protein sequence. C.Jacqemin and J.Royaute [15] proposed technique to retrieve terms and their variants in a lexicalized unification-based framework. A.E.Monge and C.P.Elkan [16] had developed the algorithm to approximate duplication of the records and properties that must satisfy pair-wise records property for the successful removal of the redundant. The easy and best way to remove redundant from the database is to sort the database and identify the redundant records adjacent to each other; this technique of cleaning the database by removing redundant is given by D.Bitton and D.J.Dewitt [17].

Development of ETL process framework is a complex activity and requires specific techniques to conceptualize its activities for a particular domain. Model driven ETL approach is based on domain specific technique of particular platform and emphasis on the automatic generation of code from specific models [18].

Despite the emphasis of automated code development from ETL model, the researchers still hold that code based ETL is better that GUI based approach [19].

The present work focuses on transformation of web data in the staging area of Web ETL framework by identifying and removing duplicate tuples occurring in different sequence using Levenshtein matching metric and subsequently loading loaded in hadoop based distributed database nodes for retrieval and storage.

3. Proposed framework

The classical ETL system as shown in figure 1 is unable to handle large voluminous data sets. With the expansion of web, there is tremendous increase in generation of data, and most of the data is available in the form of web pages. There is a requirement for improvement of classical ETL process model for such type of highly unstructured data and to be able to process such huge amount of structured or unstructured data efficiently, correctly and within the given time frame.

Data accuracy and responsiveness of an ETL system plays a vital role in the generation of knowledge from the data warehouse. These multiple heterogeneous sources of data vary morphologically; thus it becomes much more evident that extraction and transformation algorithms should be able to extract and transform large data sets efficiently and accurately

The Web ETL framework is proposed for precision and optimization of ETL processes in knowledge discovery to extract, transform and present large data sets from multiple public domains in distributed environment as shown in figure 2. The result is performance optimization of ETL in handling large data sets. This performance optimization is validated experimentally by applying ETL system algorithm on USPTO patent domain to prove the authenticity of the proposed approach. However, the present scope of the research paper is on transformation and loading phases only.

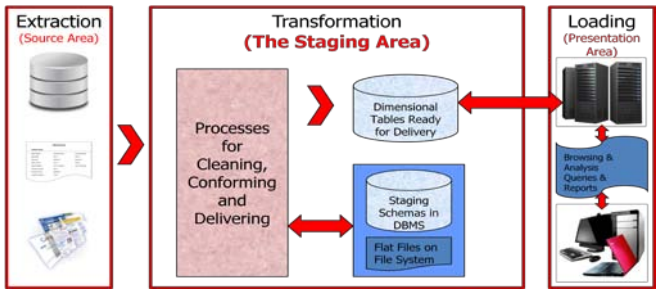

Figure 1. Classical ETL Framework

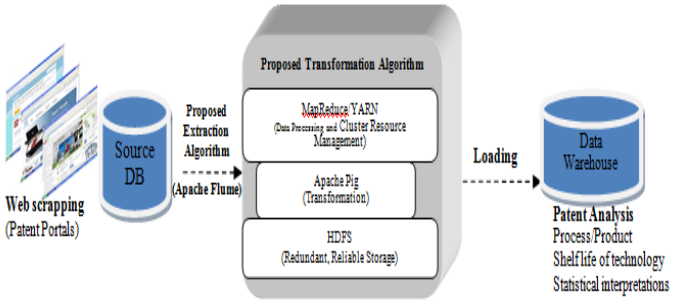

Figure 2. Web ETL Framework

3.1. Transformation Process

Dealing with Web ETL has many complexities due to the inherent nature of heterogeneity in web data and therefore dealing with such data requires robust techniques not authorized with conventional schema. The techniques involve cleaning, transformation and integration [20]. Transformation in data mining is performed before storing the data in the data warehouse so that subsequently data mining techniques can be applied on the data to get trustworthy result in lesser time. The following steps for data cleaning are modified for the present proposal to remove duplicity in the records so that it can be stored in data warehouse for further usage and easy access.

Transformation Algorithm

1. Establish the database connectivity, tokenize the elements within the field and perform token wise sorting within the field.

2. Choose a key to sort the database and sort it through the chosen key.

3. Execute the Levenshtein matching metric to remove the records which are not unique.

3.1.1 Token-wise Field Sorting

The main issue in database is the different representations of the same real world entity which creates the redundant records. To deal with this problem first the components within the field are tokenized. Tokenization is carried out by using delimiter such as colons, semi- colons, space, punctuations etc. The sample data is taken from patent portal like tokens in the inventors field {Aoki; Shigenori (Kawasaki, JP), Kato; Masayuki (Kawasaki, JP)}. After sorting the tokens result is {Aoki; Masayuki (Kawasaki, JP); Shigenori (Kawasaki, JP) as shown in Table 1.

Table 1. Dataset from patent database

PATENT INVENTORS
Sofet; Marco(RivaroloCanavese,IT)
Takamastsu; Shuji(Kanagawa,jp)
De Dobbelaere; Peter(San Diego,CA)
Kawamoto; Hirnori(Kawaguchi,jp)
Marco(RivaroloCanavese,IT); Sofet

Table 2. Token wise field sorting

PATENT INVENTORS
Marco(RivaroloCanavese,IT); Sofet
Shuji(Kanagawa,jp); Takamastsu
De Dobbelaere; Peter(San Diego,CA)
Hirnori(Kawaguchi,jp); Kawamoto
Marco(RivaroloCanavese,IT); Sofet

3.1.2 Database Sorting

Database sorting is performed after token wise sorting. Database sorting depends on the key chosen for the sorting. Choosing the key also plays a vital role to group records and thus helps in the identification of duplication. For example,

a table having fields patent number, patent name and patent gender. Now, choosing patent gender as a key to sort database will not bring the records close to each other as there are many patents having same gender. So choosing the key like “patent inventors” attribute to sort database should be a wise decision as shown in table 3.

Table 3. Database after applying database sorting

PATENT INVENTORS
De Dobbelaere; Peter(San Diego,CA)
Hirnori(Kawaguchi,jp); Kawamoto
Marco(RivaroloCanavese,IT); Sofet
Marco(RivaroloCanavese,IT); Sofet
Shuji(Kanagawa,jp); Takamastsu

3.1.3 Matching Records

Token wise sorting is followed by the database sorting, and becomes easy where records are bought together or nearby to find out the redundant records. For matching the records, there are several string matching algorithms such as exact string matching, single error string matching and Levenshtein string matching.

(i) Exact String Matching - Exact string matching will return the value 1 if the strings are exact matching to one another else 0.

(ii) Single Error String Matching - Single error string matching checks for the single error between two strings. It compares two strings if there is some missing character, additional character or substitution of character. It allows only one error out of these three.

(iii) Levenshtein Distance Matching - Levenshtein distance matching checks the changes required to make two strings similar by performing the simple operations such as delete, insert or update. Mathematical equation working for Levensthein distance matching is –

$$lev_{a,b}(i,j) = \begin{cases} \max(i,j) & if\ \min(i,j) = 0; \\ \min \begin{cases} lev_{a,b}(i-1),j) + 1 \\ lev_{a,b}(i,j-1) + 1 \\ lev_{a,b}(i-1,j-1) + 1_{(a_i \neq b_j)} \end{cases} & otherwise. \end{cases} \quad (1)$$

where i= first i characters in string a

j= first j characters in string b

a,b = strings for comparison

Table 4. Database after applying database sorting

PATENT INVENTORS
De Dobbelaere; Peter(San Diego,CA)
Hirnori(Kawaguchi,jp); Kawamoto
Marco(RivaroloCanavese,IT); Sofet
Shuji(Kanagawa,jp); Takamastsu

3.2. Loading

Loading in ETL is the process to transfer the data from staging area to destination after transformation process. In loading phase, data from staging area is transferred to the data warehouse. There are periodic updates in data warehouse rather than continuous. Large number of records is loaded to multiple tables in single data load. Efficient loading process should be designed in such a way that offline time of data warehouse can be minimized.

With the tremendous growth of data almost exponentially, the role of big data has emerged as the continuous stream of data has to be processed through ETL system for loading in data warehouse and therefore the various techniques are proposed based on its system architecture, usability and its benefits and limitations [21].

Map Reduce is integrated with graphical tools like Pentaho and Talendby from ETL chains for rapid generation of ETL chain but still it is not the replacement for programming based tools created using Hadoop based environment [22].

As there is large data set, loading the data in hadoop environment is required for fast processing. Apart from hadoop, certain tools and services such as Sqoop tool, Apache HBase, Apache flume and Apache Yarn have been integrated into the environment according to their specific purpose.

4. Experiments and Results

The problems in experimentally validating the proposed web ETL framework and transformation of data through sorting is due to the restrictions either copyright or capcha protected imposed by most of the web portals especially patent portals. Therefore, to experimentally validate the proposed solution, a web portal has been selected for extraction of web pages and its transformation by removing duplicate values and subsequent storage in hadoop cluster for futher mining of data.

4.1. Experimental setup

Transformation and loading is implemented in eclipse on Linux. Processor used is Intel(R) core(TM) i5-2400, RAM 4GB. Eclipse version-platform is 3.8.1-5.1. Tool used is

sqoop tool, version 4.6. Number of nodes taken for parallel processing in hadoop cluster is two.

The benchmarked United States Patent and Trademark Office (USPTO) website is being considered as a case study for implementing the algorithm. The website consists of a multiple number of dynamic web pages, which may return number of pages corresponding to a single search query. The major strength of this work is that the refined search is carried out for searching the patents granted during between the dates 7/3/1979 and 8/3/1979.

4.2. Experimental Results

The result is based on two parameters namely noise reduction by removing duplicate values i.e. “Inventors names” occurring in different patents in USPTO portal and secondly, efficient loading of filtered or de-duplicated data in the hadoop cluster. The overall result is highly efficient web ETL framework that can process continuous stream of incoming web data which further can be extended from GBs to TBs without redesigning the ETL framework. The result is verified on USPTO portal.

4.2.1 Transformation

Transformation is performed on 11365863 (2.7144 GB) of patent records retrieved from USPTO patent database. The original table is patent inventor table 1 which contains various redundant records. These redundant records are due to different formats of records i.e. some patent inventors name are written in the form first, middle, last name while other record are in form middle, first and last name. Also there are some names in patent inventor field that are written in different serial order.

Initially, the elements in the field are tokenized. For tokenizing the elements of the field, semicolon delimiter is used to separate elements. After tokenization, sorting is performed on the field values. Subsequently, token wise sorting on database is performed to bring records close to each other. Patent inventor key is used to transform and clean the database and to sort database.

Alphabetical sorting on patent inventor is performed on the database which brings similar records together.

The final step in the data transformation is to identify redundant records and remove them. Matching is performed to figure out the replicated records. For record matching, string matching algorithm Levenshtein string matching is used. In Levenshtein string matching, percentage of redundant records is calculated from the result records which are unique and are not 100% matched and are stored in the hash table from where afterwards the records are moved to the database table.

4.2.2 Loading

Loading the data to the warehouse is the final step. In this phase, transformed records are inserted from database to data warehouse. In this step, extracted and transformed data are written into the dimensional structures accessed by the end users and application systems. Sqoop tool is used for loading purpose and to transfer the data between mySQL to hadoop distributed file system. Patent table in mysql is shown in Figure 3. Before moving the data to HDFS, patent_uspto table is created in the HDFS shown in Figure 4 to transfer user data and sqoop command is executed to load the final data.

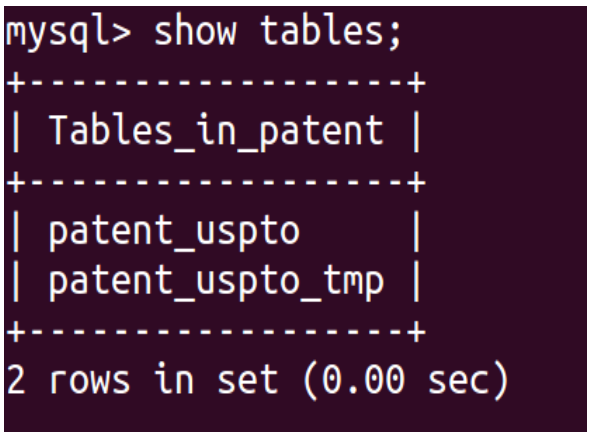

Figure 3. Patent database in staging area

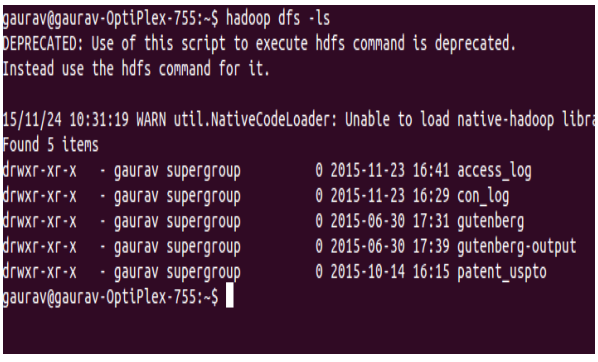

Figure 4. Patent database in HDFS

Heavier the dataset, the larger number of nodes in Hadoop cluster, hence multinode hadoop cluster setup is done. In our setup, multinode clusters composed of one master and one slave named as HadoopMaster and HadoopSlave1. With the increase in dataset, more nodes can be added in the cluster. The present cluster setup of

two nodes is shown as web view of hadoop multinode in Figure 5 with two Active nodes.

The nodes are able to retrieve data after being populated through staging area where transformation is performed on the input extracted data. As we are using hadoop multimode file system for loading of data thus large dataset can be retrieved efficiently. Moreover, a failsafe system is an advantage in case any node is down due to any network failure.

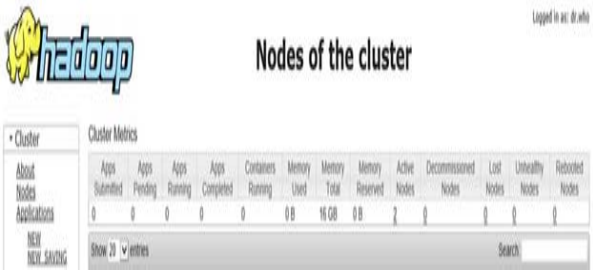

Figure 5. Web view of hadoop multinode cluster

```
hduser@HadoopMaster:-~$ jps
2764    NameNode
2980    Jps
4046    ResourceManager
```

Figure 6a. Hadoop Services on master node

Hadoop services on master and slave nodes are shown in Figure 6a and 6b respectively.

After setting the hadoop multinode cluster, loading patent_uspto table from mySQL to HDFS is performed. Time taken by Sqoop tool to load the data of size 2.7144 GB in 356.7999 seconds.

These redundant records are due to different formats of records i.e. some patent inventors name are written in the form first, middle, last name while other record are in form middle, first and last name. Also there are some names in patent inventor field that are written in different serial order.

The present work provides better performance of hadoop based web ETL framework than the traditional SQL RDBMS based ETL framework as the same is able to efficiently handle the stream of unstructured data.

```
hduser@HadoopSlave1:-~$ jps
4196    Jps
3918    NodeManager
2549    DataNode
hduser@ HadoopSlave1:-~$
```

Figure 6b. Hadoop Services on slave node

```
FILE:   Number of read operations=0
FILE:   Number of large read operations=0
FILE:   Number of write operations=0
HDFS:   Number of bytes read operations=87
HDFS:   Number of bytes written=66940
HDFS:   Number of read operations=4
HDFS:   Number of large read operations=0
HDFS:   Number of write operations=2
Job Counters
        Launched map tasks=1
        Other local map tasks=1
        Total time spent by all maps in occupied slots (ms)=7321
        Total time spent by all reduces in occupied slots (ms)=0
        Total time spent by all map tasks (ms)=7321
        Total vcore-milliseconds taken by all map tasks=7321
        Total megabytes-millieconds taken by all map tasks=1830250
Map-Reduce Framework
        Map input records=11365863
        Map output records=11365863
        Input split bytes=87
        Spilled Records=0
        Failed Shuffles=0
        Merged map outputs=0
        GC time elapsed (ms)=109
        CPU time spent (ms)=1380
        Physical memory (bytes) snapshot=133431296
        Virtual memory (bytes) snapshot=1927602176
        Total commited heap usage (bytes)=44040192
File Input Format Counters
        Bytes Read=0
File Output Format Counters
        Bytes Written=66940
15/11/29 10:24:43 INFO mapreduce.ImportJobBase:  Transferred 2.7144 GB in 356.7999 seconds (1.7942 MB/s)
15/11/29 10:24:43 INFO mapreduce.ImportJobBase:  Retrieved 11365863 records.
gaurav@gaurav-Optiplex-755:~$
```

Figure 7. Loading data in HDFS in HDFS

The results show that the pre-processing steps for data transformation using token wise sorting, database sorting and Levenshtein string matching bring more potentially matching records to a close neighbourhood for easy and fast detection and removal of redundant records. The transformation of data in the staging area is accelerated by the applicability of proposed hybrid algorithm as mentioned above. Furthermore, to deal with such a huge data, existing algorithm may fail for efficient performance because of structural dependency on RDBMS and conventional sorting methods based on indexing which becomes inefficient for string matching and in data warehouse hence, loading of data to hadoop environment is done.

5. Conclusion

Since the size of routine databases is becoming enormous day by day so to achieve performance efficiency the data cleaning, ETL process has become a necessity of this information age. Unless the exact and accurate information about redundant data is not known, processing of the data for further analysis cannot be validated due to the production of untrustworthy data. The aim is to achieve quality data efficiently hence the present work is divided into two parts first, data cleaning to scrub the data and to deal with missing, incorrect values and removal of replication and second, to load quality data in the data warehouse. Web ETL Framework requires web documents mainly web-pages as input for extraction of data. Various structured and unstructured resources are utilized from the benchmarked patent sites to create present study patent database. The work done in the field of data transformation to de-redundant the records is described as well. The present research aimed at finding the solution to clean the data in the database by removing the redundant records for similar entities and to load the cleaned data efficiently into distributed hadoop based data warehouse. It is performed on the data before moving it to data warehouse so that further data mining techniques and queries can be applied on the data set to achieve better and efficient results. The records are first sorted token wise then database sorting is committed based on the chosen key and finally redundancy is removed by Levenshtein distance matching to load unique and de-redundant records into multimode hadoop cluster based data warehouse.

Acknowledgement

The author would like to express United States Patent and Trademark Office, www.uspto.gov web pages source for providing access for extracting data from web pages. The authors further wishes to acknowledge the reviewers for their valuable and helpful comments.

References

[1] REDMAN, T., (1996) *Data Quality for the Information Age,* (Artech House).

[2] Stuckenschmidt H. (2012), Data Semantics on the Web. *Journal of Data Semant*, 1:1–9

[3] FOX C., LEVITIN A. and REDMAN T. (1994) The notion of Data and Its Quality Dimensions. *Information Processing and Management* **9**(19)

[4] KIMBALL R., (1996) KIMBALL R., (1996) DBMS *Dealing with dirty data*, **9**(10): 55-60

[5] LENZERINI M. C. and BATINI S. NAVATHE, (1986) A Comparative Analysis of Methodologies for Database Schema Integration. *ACM Computing Surveys* 18(4): 323–364.

[6] STOLFO S. and HERNANDEZ M. (1995) The merge/purge problem for large databases, In *Proceedings of* ACM SIGMOD Int. Conference on Management of Data pages, 127-138,.

[7] STONEBRAKER M. and ULLMAN J.D. and SIBERSCHATZ A. (1996), Database research: achievements and opportunities into the 21st century *A report of an NSF workshop on the future of database research*, SIGMOD Record

[8] U Fayyad, G Piatetsky-Shapiro, P Smyth (1996), From data mining to knowledge discovery in databases, *AI magazine* 17, 3, 37.

[9] MONGE A.E. and ELKAN CP. (1996) The field matching problem:Algorithm and applications. *Proc. of the 2nd Int. Conference on knowledge discovery and Data Mining*, 267-270.

[10] REDMAN, THOMAS C. (1998), The impact of poor data quality on the typical enterprise, *Communications of the ACM*, 41, 2, 79-82.

[11] DIEGO C., GIACOMO G. D., LENZERINI M., NARDI D., and ROSATI R (2001), Data integration in data warehousing, *International Journal of Cooperative Information Systems*, 10, 3, 237-271.

[12] SIMOUDIS, EVANGELOS, LIVEZEY B., and KERBER R. (1995), Using Recon for Data Cleaning, *In KDD*, 282-287.

[13] BARRY AND COTE L. D. (1996), Data warehouse: from architecture to implementation, *Addison-Wesley Longman Publishing Co. Inc., Boston, MA, USA*

[14] BILLY P., FREDERICK E. PETRY AND BUCKLES (1982), A fuzzy representation of data for relational databases. *Fuzzy sets and systems*, 7, 3, 213-226.

[15] SMITH T.F. AND WATERMAN M.S. (1981), Identification of common molecular subsequences, *Journal of Molecular Biology*, 147, 195-195.

[16] JACQEMIN C. AND ROYAUTE J. (1994), Retrieving terms and their variants in a lexicalized unification-based framework *In Proc.of the ACM_SIGIR Conference on Research and Development in information retrieval*, 132-141

[17] BITTON D. AND DEWITT D.J. (1983), Duplicate records elimination in large data files, *ACM Transaction on Database Systems (TODS)*8(2), 255-265.

[18] Marko Petrović, Nina Turajlić, Milica Vučković, Nenad Anicic, Sladjan Babarogić, Nenad Anicic, Development of ETL Processes Using the Domain-Specific Modeling Approach, *In book: Emerging Perspectives in Big Data Warehousing*, Jan 2019 DOI: 10.4018/978-1-5225-5516-2.ch010

[19] Neepa Biswas, Anamitra Sarkar, Kartick Chandra Mondal, Empirical Analysis of Programmable ETL Tools, *International Conference on Computational Intelligence, Communications, and Business Analytics CICBA 2018:*

Computational Intelligence, Communications, and Business Analytics pp 267-277

[20] Shaker H., Abdeltawab M. and Bastawissy H. (2011), A proposed model for data warehouse ETL processes *Journal of King Saud University – Computer and Information Sciences*, 23, 91–104

[21] Guang Sun, YingJie Song, ZiQin Gong, Xiya Zhou, Xinyi Zhou, YiLin Bi, Survey on streaming data computing system, Conference: the ACM Turing Celebration Conference - China, May 2019 DOI: 10.1145/3321408.3326687

[22] Michael Frampton, ETL with Hadoop, In book: Big Data Made Easy, December 2015 DOI: 10.1007/978-1-4842-0094-0_10