ADVANCES IN CYBER-PHYSICAL SYSTEMS

Vol. 5, No. 2, 2020

ALGORITHMIC AND SOFTWARE OF MIGRATION OF DATABASES IN HYPERMARKET NETWORK

Volodymyr Mruts, Marta Mashevska

Lviv Polytechnic National University, 12, S. Bandery Str., Lviv, 79013, Ukraine
Authors' e-mail*: volodymyr.mruts@gmail.com, marta.v.mashevska@lpnu.ua*

https://doi.org/10.23939/acps2020.02.077

Submitted on 17.12.2020

***Abstract*: Possible ways and options of migrating data from existing database management systems to new ones have been analyzed in the article. Also, the main advantages and disadvantages of these methods have been considered, the common problems that may arise during migration and the standard requirements for this type of system have been given. In the practical part, the Exact Transform Load (ETL) system has been developed with the implementation of all its functional and non-functional requirements, migrating data from the old system to the new one.**

***Index Terms*: data migration, automation, Exact Transform Load (ETL), transformation**

I. INTRODUCTION

The modern world, standing firmly on both feet in the information age, is extremely effective in generating huge amounts of information of all kinds and varieties, including both man-made data (music, art, movies, literature, photographs, emails, tweets, program code, and even plain handwriting) and machine-generated data (measurements of various things or phenomena, genomic codes), the so-called "logs" – historical data on human or program activity). And all this information needs to be processed and stored somewhere. It comes to the aid of Database Management Systems – DBMS.

II. THE CURRENT STATE OF THE PROBLEM

Now on the market there are many different DBMS and, therefore, the client can choose the system most suitable to him depending on a lot of parameters. One of the most important parameters, from the point of view of business, is the price for the license for use of a concrete DBMS. Currently, there are also a large number of so-called "Open Source" solutions. It's open-source projects, which are provided with free licenses, such as GNU / GPL (General Public License) or BSD (Berkeley Software Distribution license). This software is free to use, sometimes even on a commercial basis, which allows the customer to save money when choosing such software. But sometimes it happens that a certain business for a long time used expensive solutions, not using their capabilities to the full and it was decided to switch to free / cheaper software, in our case a database. But how to properly transfer the huge amount of data stored on the current system?

To transfer this data manually, filling new DB with new records, is irrational from the point of view of use of resources. That is why ETL solutions (Extract, Transform, Load or Extract, Convert and Download) are used [1, 2].

With ETL, data is migrated from one system to another, following certain rules set by the client. As a part of this thesis, an ETL solution was designed and developed, the purpose of which is to ensure the migration of data from one relational database to another, with their subsequent transformation, validation and logging (recording a detailed chronology) of the process. Scheme of ETL process is shown in Fig. 1.

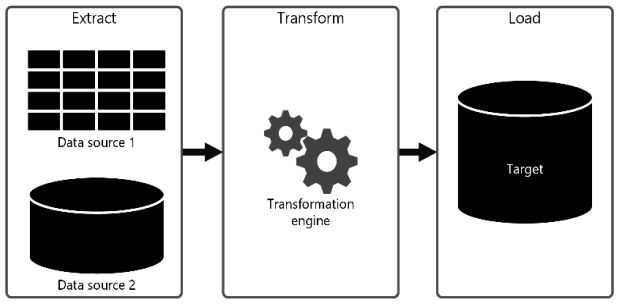

Fig. 1. Scheme of ETL process

III. STATEMENT OF THE MAIN MATERIAL

The purpose of the work is to develop automation software of migrations of databases in hypermarket network

As it was mentioned earlier, the so-called "digitalization" now prevails in the world. Everyday processes familiar to us are increasingly going online. And the more such digitized processes there are, the more certain information that needs to be stored they generate. In addition, data is increasingly being transferred from the usual paper form to electronic. On the one hand, it significantly speeds up and simplifies the interaction with this information, but, on the other hand, it creates a single point of failure – a certain information system through which this data is translated into electronic form, processed, changed and stored. This information system (IS) also includes an integral component – DBMS (database management system) [3].

It depends on how reliably in terms of fault tolerance or hacking security data is stored, how much of

this data can be processed and stored by the system and, ultimately, the ease of use is also a very strong argument in choosing a particular database. Currently, there are 2 leading types of databases (DB) on the market: relational and non-relational. The popularity of specific databases for 2019 is presented in Fig. 2.

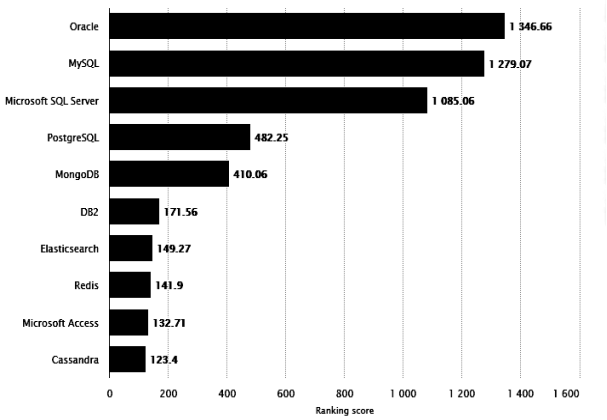

Fig. 2. DBMS division in the market by popularity [4]

As you can see, the key positions in this ranking are occupied by relational databases [5, 6], that is why we choose them as a basis.

However, sometimes there are situations when the client needs to change the basic DBMS due to certain factors:

- data security issues;
- problems with scaling certain components;
- high cost of use;
- obsolescence;
- difficulty in maintenance;
- performance issues.

But if the decision to change the main database was made after a long time of its work [7] – then to do it without the help of third-party software, most likely, will not work, because the system stores and operates large amounts of data, and transfer them to a new system manually – inefficient and irrational.

In this case, the best solution is to develop and implement an ETL system.

Extract, Transform, Load (ETL) – a process used in databases and, especially, in data warehouses (DW – Data Warehouse) to ensure automatic data migration [8]. The ETL process can be divided into the following stages:

- data extraction;
- conversion of data for further storage in the appropriate structure, or for further analysis;
- loading data into the target structure of their storage. This can be a regular database, data warehouse, or the so-called data showcase (DataMart);

Data extraction. The first stage of the ETL process involves extracting data from their sources or other source systems. This is one of the most important and responsible aspects of ETL, because without the correct extraction of data from sources, the further functioning of other processes is not guaranteed. Typically, data warehouses combine data from different sources, while each other system may use its own form of organization or storage of data or even have a different file format. Data sources can consist of relational databases, JSON, XML files, and so-called "flat" databases, and so on. Sometimes, the streaming of the source data source is used, and it is downloaded on the fly to the target database. It is a method that does not require an intermediate data warehouse, the so-called "staging". In general, the extraction phase is aimed at reducing the data to a single standardized format, which is required in the next step of information conversion.

The other part of the extraction process involves data validation to verify that the data extracted from the sources has the expected values in a particular business domain (as the price of the product cannot be negative). Such rules are provided by the customer and they depend on the real nature of this information. If the data does not comply with any verification rules, they may deviate in whole or in part. Here are some ways to further use this data: it can form separate data structures that will be further analyzed to fix incorrect records, or sometimes the ETL process itself can be extended by some logic, rules of correction or substitution of incorrect records to accept and validate this data target system and could move on to the next stage of processing.

Transformation. At the stage of data transformation, a series of rules provided by the customer based on real business logics is applied to them. Some of these data do not need to be converted at all, if the business logic has not changed when moving them to the new system, and there were no technical errors in its implementation in the old system.

Keep in mind that an important function of the conversion is to clean the data, which requires only "correct" data to be transferred, so as not to transfer so-called "garbage" to the new system – data that violates consistency or does not meet certain rules. The problem with the interaction of different systems involves many possible factors, such as character sets that are available for use in one system may not be available in other systems. In some cases, the following types of transformations may be required to meet business and technical needs:

- selecting only certain columns to load: the system can ignore rows that fall under one of the rules, such as those where a field is NULL, empty or equal to 0;
- sorting data based on a list of columns to improve search efficiency: for example, on the target system, the search is often conducted by the ID field – then it would be best to sort the data in the same order;
- conversion of the encoded values: for example, in the data source the logical field is made as type bit, where value 1 – true, 0 – false, and in the final system it is type char (1), which takes T – true or F – false;
- value of the free form of coding: the conversion of the value of "M" into "Male";
- creation of surrogate key values: if you want to represent a certain form of data historicity, or the history of their modification – then business keys that have a

value associated with a business process and uniquely identify the record, should be supplemented with a surrogate key that will not carry some attached meaning;

- splitting a column into several columns: for example, the full name column into 3 columns:
- first name, middle name, last name;
- derivation of field values calculated on the basis of other fields: for example, total amount = price_ for_goods *quantity.

Load. This phase loads the data into the final target system. Depending on the needs of the organization, this process can vary greatly. Some data warehouses can overwrite existing information, while others can add historicity and store both data sets. More complex systems can support the audit of changes in the database source and reflect the same changes in the final database

IV. PRACTICAL IMPLEMENTATION

Development and implementation of ETL solution is a complex process, and before it starts you need to clearly define the features of the system, necessary future functionality, agree on a specific set of business rules with the customer, which will be transformed during the ETL process.

The development and implementation of this ETL solution involves ensuring the following functional requirements:

1. ensuring the quality of migrated information;
2. translation of information into certain formats;
3. correct loading of data into the target system;
4. the impossibility of losing any data during the migration process;
5. full logging of the process and saving information about the ETL migration;
6. determination of all possible sources of the so-called "hardcode" and providing the ability to centrally change certain parameters of the migration ETL (DBMS source, target DBMS, table schemes, etc.);
7. ensuring verification of data integrity as well as possible correction of data consistency on the target database for future overlaying of foreign keys;
8. automatic preparation of the defined scheme of tables and relations between them on the target DBMS within the ETL process;
9. post-migration validation and verification of the target DBMS scheme and data in it for non-compliance;

Non-functional requirements determine the general principles and criteria of work, without delving into individual scenarios of behavior, including reliability, performance, ease of use.

The development and implementation of this ETL solution involves ensuring the following non-functional requirements:

1. the decision should look logical and clear;
2. ETL process must work productively (run less than 60 minutes, with a total data of 1.000.000 records in tables);

After that UML-diagram of future system was developed (Fig. 3)

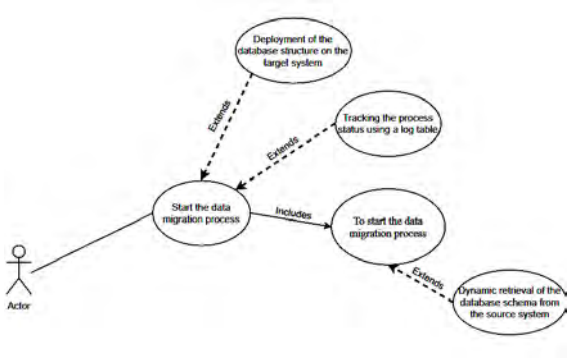

Fig. 3. UML – diagram of future system

Then we need to determine the logic of the data migration process. Within this work, it looks like this:

1. the data enters the source system;
2. at the beginning of the migration, the data is checked for compliance with all restrictions, formats, and quality. If any data is not tested in this way, it is written to a file with incorrect data and does not enter the target system;
3. then, if necessary, the data is transferred and converted into the desired format;
4. after that the data is checked for integrity and, if certain discrepancies are found, the process of restoring integrity is started;
5. in the end, the verified, validated and reduced to the required format data enter the target system;

Graphical representation of logic can be seen in Fig. 4.

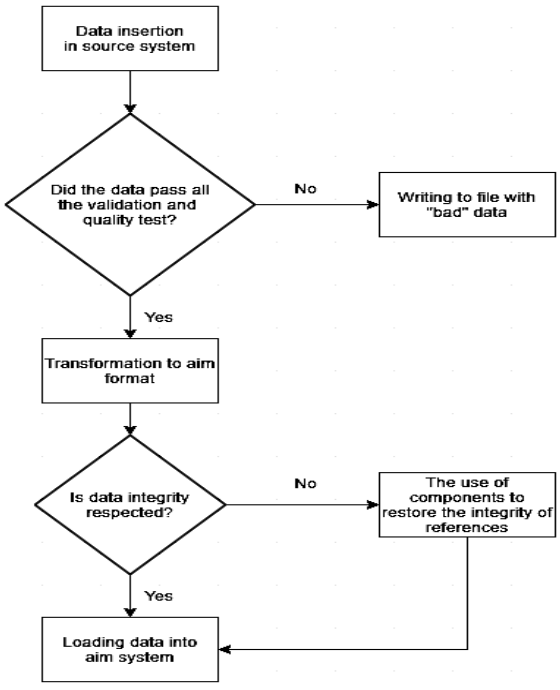

Fig. 4. Graphical representation of logic of ETL process

We will migrate data from the database of the hypermarket network developed within the framework of this article. The logic diagram of this database is presented in Fig. 5.

When all the previous steps have been successfully completed, we proceed to the development of the data migration process itself.

After analyzing various methods, technologies and environments for the development of ETL-processes, Talend Open Studio was chosen as the main tool [9, 10]. The logical process of migration will be demonstrated on the example of ETL – Jobs, which provides migration of the Employee table from the database source to the target database (Fig. 6).

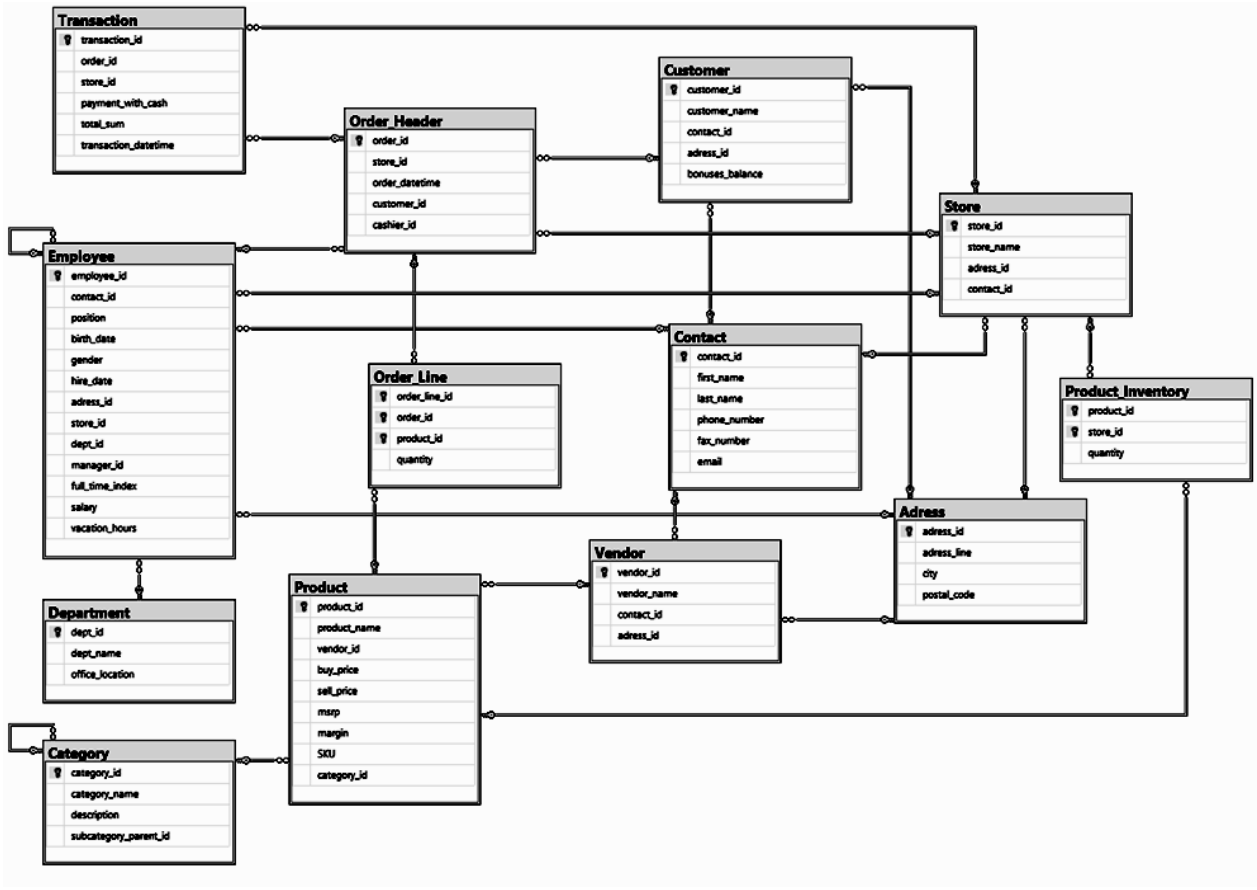

Fig. 5. Logical model of the developed DB schema

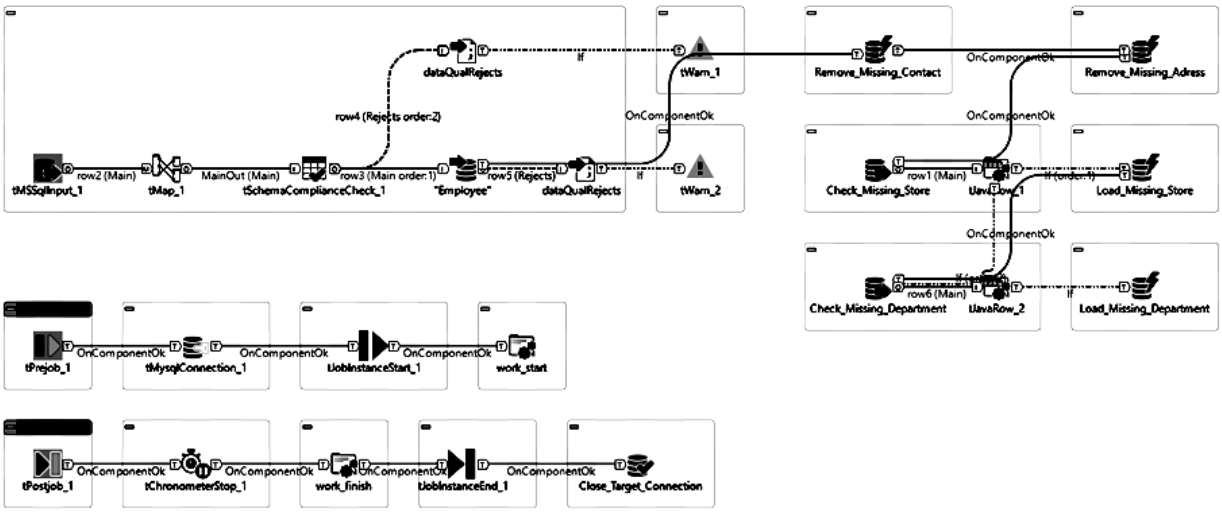

Fig. 6. ETL-job Employee

The process of executing ETL Jobs can be divided into 3 successive stages:

- preparatory actions that take place before the main part;
- main part;
- post-migration actions that take place after the main part.

1. Preparatory actions

Logical ordering in TOS is possible by constructing a "path" of the process and connecting the components in a certain way and under certain conditions. However, it is also good practice to use components – orchestrators. One of these is tPreJob. This is a trigger component that clearly indicates that other components connected to it must be executed before the main part

The whole preparatory part consists of 3 components (excluding tPreJob):

- tMySqlConnection – a component that explicitly creates a connection to the target database for further use in the migration process;
- tJobInstanceStart – a component used to start logging the data migration process;
- tJavaRow "work_start" – a component used to insert in the migration process of third-party program code written in Java. In this case, it is used to display in the command bar an inscription about the beginning of work. This is required for advanced log collection;

2. The main part

The main part of ETL Jobs consists of extracting data, transforming it and loading it into the target system, or into files for data that have not been validated

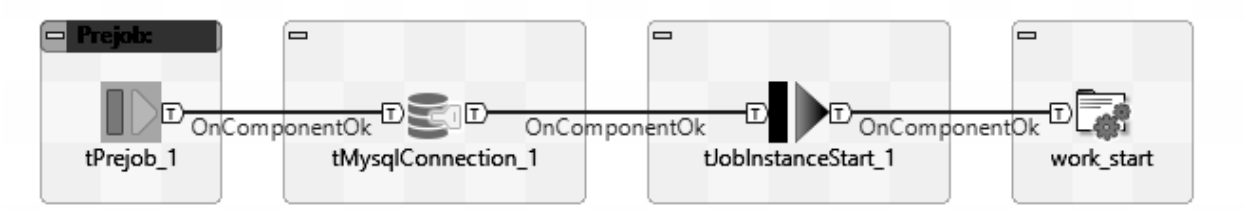

Fig. 7. tPrejob component which starts preparation part

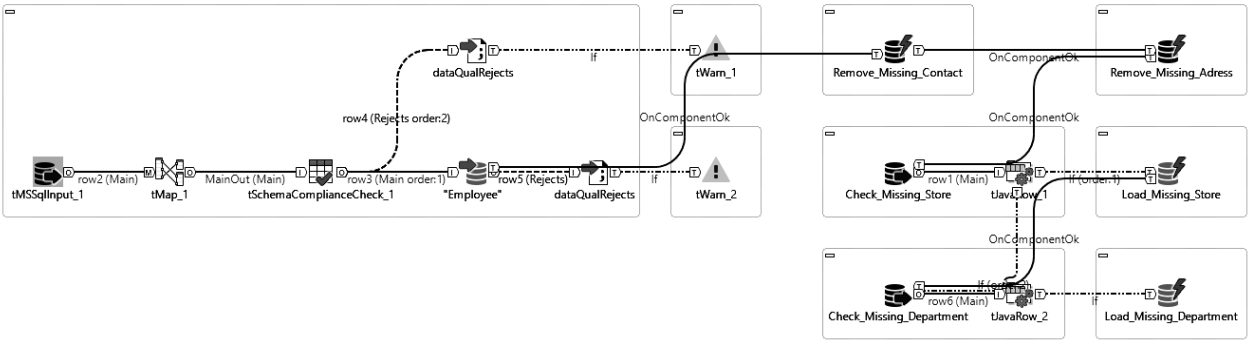

Fig. 8. Main part of ETL Jobs

The main part begins with the component tMSSqlInput, which is component responsible for downloading data from the source database and passing data on to the process. In this case, the component unloads the data of the Employee table.

Then the unloaded data is transferred to the tMap component, where they can be combined with other sources, perform various transformations, aggregations and decompositions. And also, to change data types, their size, possibility of insertion of NULL-values, etc.

After that, the uploaded data enters the component tSchemaComplianceCheck, where they are checked line by line for compliance with the scheme of the target database. The component checks the data for compatibility of data types, or their recording formats, for the absence of empty values, where this is not allowed by the target system, and for errors in the size of the data, or their accuracy.

If the string does not meet the conditions of the target system, it is sent to the component tFileOutput Delimited "DataQualRejects". This component writes incorrect data to a file that can then be analyzed. And then displays a message about the mismatch of data in the table, as well as the number of incorrect rows, in the system tray.

If the string satisfies the conditions of the target system, it is written to it using the tMySqlOutput component. If an attempt is made to insert a duplicate, such an entry is also redirected and written to DataQualRejects, but with an error code indicating that the entry was not inserted into the target system because it duplicates another entry.

Then the process moves on to the components that ensure the integrity of the data in the target system. Since foreign keys are then superimposed on the plates, they must be in a consistent form. According to the principle of operation, they are divided into 2 types: conversion to NULL values of data that are missing in the parent table, or uploading missing data to the parent table. For the tables of values that refer to the tables Contact and Address – this is a replacement, and for those that refer to the Store and Department – upload.

3. Post-migration part

The post-migration part of ETL Jobs is also marked with the tPostJob component, which clearly indicates that the next steps will be performed after the preparatory and main part.

- tChronometerStop – a component required for more detailed logging. It detects the execution time of Jobs and displays it in the command line;
- tJavaRow "work_finish" – a component that displays the status of the end of the process;
- tJobInstanceEnd – a component used to complete the logging of the data migration process. The log table records the duration of the process, time frame, number of loaded and unloaded rows;
- tMySqlCommit – a component that closes the global transaction that was revealed at the beginning of the process;

To check the correctness of the migration process, fill in the database – the source of test data generated by the service www.generatedata.com. In Fig. 10 you can see an example of test data in the Employee table

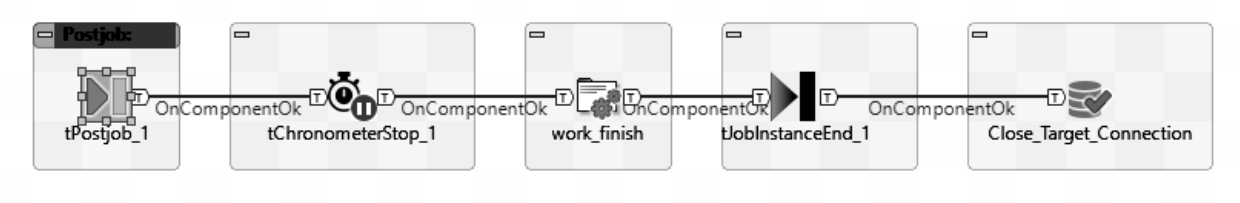

Fig. 9. PostJob component which starts post-migration part

	employee_id	contact_id	position	birth_date	gender	hire_date	adress_id	store_id	dept_id	manager_id	full_time_index	salary	vacation_hours
1	1	5	violet	1978-04-09	M	2013-03-26	1	7	1	[NULL]	Y	283,781	410
2	2	2	green	1973-01-03	F	2015-05-28	3	6	7	[NULL]	Y	184,164	425
3	3	7	orange	1984-06-27	M	2013-02-27	2	7	9	[NULL]	Y	196,588	350
4	4	8	blue	1961-08-21	F	2008-08-18	6	2	7	[NULL]	Y	153,036	504
5	5	6	red	2002-06-15	F	2016-04-08	3	5	4	[NULL]	Y	402,794	428
6	6	3	blue	2014-07-26	F	2011-07-14	2	10	2	[NULL]	N	267,443	208
7	7	9	red	1983-06-30	F	2011-11-18	2	9	1	[NULL]	Y	297,236	39
8	8	3	violet	2004-07-18	M	2015-05-12	9	1	5	[NULL]	Y	171,776	347
9	9	3	indigo	1978-12-20	M	2018-11-02	1	3	2	[NULL]	N	186,085	12
10	10	10	indigo	1964-05-07	M	2020-12-15	5	8	7	[NULL]	N	255,863	167
11	11	7	red	1991-08-03	F	2002-09-29	2	10	1	[NULL]	N	191,455	367
12	12	3	blue	2018-08-18	F	2005-08-23	3	10	3	[NULL]	Y	127,659	353
13	13	7	red	1973-09-15	M	2004-06-18	8	1	6	[NULL]	N	434,995	523
14	14	3	yellow	1976-01-19	M	2013-08-27	10	2	3	[NULL]	N	170,113	113
15	15	5	blue	1970-08-16	F	2009-03-27	2	4	1	[NULL]	N	187,724	252
16	16	6	yellow	1977-03-29	F	2016-05-28	5	4	2	[NULL]	N	282,597	159
17	17	7	red	1991-08-01	M	2021-04-18	10	2	4	[NULL]	Y	474,651	387
18	18	7	orange	1998-05-02	M	2016-10-06	10	4	5	[NULL]	N	419,161	327
19	19	1	yellow	1959-12-28	M	2004-04-14	7	7	7	[NULL]	N	432,582	320
20	20	3	yellow	1969-05-13	M	2012-05-01	2	6	8	[NULL]	Y	369,239	484
21	21	10	indigo	1955-12-06	M	2003-09-17	2	7	7	[NULL]	Y	335,057	57
22	22	10	orange	1980-07-18	M	2014-07-11	7	10	1	[NULL]	Y	209,191	474
23	23	3	indigo	1962-01-30	F	2008-07-26	10	7	8	[NULL]	Y	170,494	44
24	24	3	yellow	1950-09-27	M	2012-09-09	5	5	7	[NULL]	Y	234,162	123

Fig. 10. Test data in the Employee table

	JOB_INSTANCE_ID	JOB_NAME	JOB_PROJECT	JOB_STARTED_AT	JOB_ENDED_AT	COUNT_INPUT	COUNT_OUTPUT	HOST_USER
1	1	ETL_Run	DIPLOMA	2020-04-29 16:29:39	2020-04-29 16:29:47	0	0	Volodymyr_Mruts
2	2	Create_DDL	DIPLOMA	2020-04-29 16:29:40	2020-04-29 16:29:40	0	0	Volodymyr_Mruts
3	3	Contact	DIPLOMA	2020-04-29 16:29:40	2020-04-29 16:29:40	90	90	Volodymyr_Mruts
4	4	Adress	DIPLOMA	2020-04-29 16:29:40	2020-04-29 16:29:41	100	100	Volodymyr_Mruts
5	5	Department	DIPLOMA	2020-04-29 16:29:41	2020-04-29 16:29:41	10	10	Volodymyr_Mruts
6	6	Category	DIPLOMA	2020-04-29 16:29:41	2020-04-29 16:29:41	10	10	Volodymyr_Mruts
7	7	Vendor	DIPLOMA	2020-04-29 16:29:41	2020-04-29 16:29:41	10	10	Volodymyr_Mruts
8	8	Product	DIPLOMA	2020-04-29 16:29:41	2020-04-29 16:29:42	100	100	Volodymyr_Mruts
9	9	Store	DIPLOMA	2020-04-29 16:29:42	2020-04-29 16:29:43	10	10	Volodymyr_Mruts
10	10	Employee	DIPLOMA	2020-04-29 16:29:43	2020-04-29 16:29:44	100	100	Volodymyr_Mruts
11	11	Customer	DIPLOMA	2020-04-29 16:29:44	2020-04-29 16:29:45	10	9	Volodymyr_Mruts
12	12	Product_Inventory	DIPLOMA	2020-04-29 16:29:45	2020-04-29 16:29:45	10	10	Volodymyr_Mruts
13	13	Order_Header	DIPLOMA	2020-04-29 16:29:45	2020-04-29 16:29:46	100	100	Volodymyr_Mruts
14	14	Order_Line	DIPLOMA	2020-04-29 16:29:46	2020-04-29 16:29:46	100	100	Volodymyr_Mruts
15	15	Transaction	DIPLOMA	2020-04-29 16:29:46	2020-04-29 16:29:46	100	100	Volodymyr_Mruts
16	16	Apply_FK	DIPLOMA	2020-04-29 16:29:46	2020-04-29 16:29:47	0	0	Volodymyr_Mruts

Fig. 11. Table of log sand chronology of the migration process

After that, configure the connection to the database, select the desired functionality and start the entire ETL process using the graphical interface Talend Open Studio for Data Integration [10]. At the end of the migration process, check the log table of the migration process, it is shown in Figure 11

The JOB_INSTANCE_ID column contains the sequence number of the ETL Jobs themselves.

The JOB_NAME column contains the name of the ETL Jobs that were running.

The JOB_PROJECT column contains the name of the project in which the ETL Job was executed.

The JOB_STARTED_AT column contains the start time of ETL Jobs.

The JOB_ENDED_AT column contains the completion time of ETL Jobs.

The COUNT_INPUT column contains the number of rows selected and retrieved from sources.

The COUNT_OUTPUT column contains the number of rows uploaded to the target system.

The HOST_USER column contains the name of the user who started the ETL Job.

Therefore, the logging functionality has been implemented successfully.

As you can see, all data was transferred successfully, except for one row from the Customer table. This means that this line should have ended up in a file with incorrect data. It can be seen in Figure 12

```
1 customer_id; customer_name;contact_id;adress_id;bonuses_balance;errorCode;errorMessage
2 10;Adipiscing Lacus Ut Associates;27;30;125.00;8;customer_id:exceed max length
```

Fig. 12. ETL-generated file with untested data

This file contains the line that did not pass validation, as well as the error code and error message. As you can see, in this case, the value in the customer_id column has exceeded the maximum length. Therefore, the functionality of tracking incorrect data is also implemented.

As one record did not get to the target DB, it could break data integrity in a DB, therefore, it is necessary to check up, whether the mechanism of preliminary loading of the lost data worked. The inserted string is shown in Fig. 13.

	customer_id	customer_name	contact_id	adress_id	bonuses_balance
1	1	Odio A LLP	24	22	703
2	2	Dolor Nonummy Limited	30	21	544
3	3	Non Institute	30	29	273
4	4	Risus In Mi Corporation	28	23	672
5	5	Vel Arcu Eu Ltd	30	20	722
6	6	Luctus Et Company	26	26	104
7	7	Quisque Varius Nam Institute	22	24	118
8	8	Orci Corp.	28	22	973
9	9	At Incorporated	25	23	543
10	10	Missing customer from Order_Header	[NULL]	[NULL]	[NULL]

Fig. 13. Moved data in the Customer table with a pre-inserted row to restore the integrity of the database

From this figure, we can see that there were rows in the Order_Header table that referenced a row in the Customer table that did not validate and did not reach the target system. Therefore, this string was artificially generated by ETL to ensure data integrity and to impose all foreign keys to further ensure it.

Therefore, as we see, the functionality of maintaining data integrity is also fully implemented.

V. CONCLUSIONS

The developed system and the scheme of a database are ready for use in real conditions and introduction in work of the enterprise. Automated data migration processes have been developed for a database management system for a hypermarket network. In this article, clear technical requirements and specifications for the ETL data migration system, as well as the requirements and necessary business logic for the database scheme of the target database were formed. The database schema that met all business and technical needs and met all the standards of relational database design were designed. Also, all functional and non-functional requirements were implemented, namely detailed logging of the process, validation of migrating data, automation, and parameterization of the process.

REFERENCES

[1] Implementation of the ETL system (ETL (Extract, Transform, Load) Sybsystem implementation of corporate data warehouse) [Online]. Available: https://www.prj-exp.ru/dwh/structure_ of_etl_process.php

[2] The main functions of ETL systems [Online]. Available: https://habr.com/ru/post/248231/

[3] Panneerselvam R "Database Management Systems", Prentice Hall India Learning Private Limited, 2011, 404 p.

[4] Ranking of the most popular database management systems worldwide [Online]. Available: https://www.statista.com/statistics/809750/worldwide-popularity-ranking-database-management-systems/

[5] Jan L. Harrington “Relational Database Design and Implementation, 4th Edition”, Morgan Kaufmann-Elsevier, 2016, 691 p.
[6] Malcolm Hamer “Relational Database Practices: Bridging the Gap Between the Theory of Database Design and Real-World Practices”, Malcolm Hamer, 2017, 242 p.
[7] MariaDB relational database management system [Online]. Available: https://web-creator.ru/articles/mariadb
[8] Детермінізм і програмне забезпечення: витяг, перетворення та завантаження (ETL) [Online]. Available: https://dou.ua/forums/topic/29284/
[9] 15 Best ETL Tools In 2020 [Online]. Available: https://www.softwaretestinghelp.com/best-etl-tools/
[10] What Is Talend? – An Unified Platform For Data Integration [Online]. Available: https://www.edureka.co/blog/what-is-talend-tool/

Volodymyr Mruts is a student of the Department of Computer Science of the Institute of Enterprise and Advanced Technologies of Lviv Polytechnic National University. Since 2019 he has worked in such a big IT company as DB/ETL Engineer. He is experienced in data migration, data warehousing, database designing and cloud services, especially AWS. In 2020 Volodymyr became a certified Amazon Web Services Associate Developer and now he is preparing for Google Cloud Platform Professional Data Engineer certification.

Marta Mashevska is a Senior lecturer of the Department of Computer Science of the Institute of Enterprise and Advanced Technologies of Lviv Polytechnic National University. In 2012 she obtained PhD in the field of "Systems and means of artificial intelligence" (degree of Candidate of Technical Sciences). Her research interest includes development of database software, the business requirements analysis and software specification document preparation, design and research of models based on fuzzy logic.