ACTA TECHNICA CORVINIENSIS – Bulletin of Engineering
Tome V (Year 2012). FASCICULE 1 [January–March]. ISSN 2067-3809

[1.] Robert HALENAR

MATLAB POSSIBILITIES FOR REAL TIME ETL METHOD

[1.] DEPARTMENT OF APPLIED INFORMATICS, UNIVERSITY OF SS CYRIL AND METHODIUS, TRNAVA, SLOVAKIA

ABSTRACT: *This article describes how to implement improved ETL process in Matlab environment. New architecture real time ETL process stills automated without human – database administrator interference, in cost of reduced accuracy rendered by level of trust. This method is constructed in Matlab environment, due to simple transformation and convert routines and functions. First we described ETL as a part of KDD, what is Real time ETL and problem how to achieve real – time in real world. In next part we present our improved near real time ETL model with new architecture containing equation for calculation the level of trust. And finally we shows how to use Matlab routines and toolkit for achieve simplicity in ETL phases.*
KEYWORDS: *Matlab; Extraction Transformation Loading; real time; data warehouse*

INTRODUCTION

Our research is about finding new methods and tools using in different stages of Knowledge Discovery in Databases. In the early stages of the KDD it is necessary to collect and preprocess data [5]. Especially, we improved the Near Real Time ETL phase (Extraction Transformation Loading), using the new architecture. Several approaches can be used, as described in [6]. Main rule is application of structures and processes in environment, which supports simple code and transformation to tasks performed in real time.

ETL PROCESS

The usual process of ETL-ing the data during the night in order to have updated reports in the morning is getting more complicated if we consider that an organization's branches may be spread in places with totally different time-zones. Based on such facts, data warehouses are evolving to "active" or "live" data producers for their users, as they are starting to resemble, operate, and react as independent operational systems. In this setting, different and advanced functionality that was previously unavailable (for example, on-demand requests for information) can be accessible to the end users. For now on, the freshness is determined on a scale of minutes of delay and not of hours or a whole day. As a result, the traditional ETL processes are changing and the notion of "real-time" or "near real-time" is getting into the game. Less data are moving from the source towards the data warehouse, more frequently, and at a faster rate. [1]

The ETL market has already made efforts to react to new requirements. The major ETL vendors have already shipped "real time" ETL solutions with their traditional platforms. In practice, such solutions involve software packages that allow the application of light-weight transformations on-the-fly in order to minimize the time needed for the creation of specific reports. Frequently, the delay between the moment a transaction occurs at the operational site and the time the change is propagated to the target site is a few minutes, usually, five to fifteen. Such a response should be characterized more as "near real time" reaction, rather than "real time", despite how appealing and promising can the latter be in business terms. Traditionally, ETL processes have been responsible for populating the data warehouse both for the bulk load at the initiation of the warehouse and incrementally, throughout the operation of the warehouse in an off-line mode. Still, it appears that data warehouses have fallen victims of their success: users are no more satisfied with data that are one day old and press for fresh data – if possible, with instant reporting. This kind of request is technically challenging for various reasons. First, the source systems cannot be overloaded with the extra task of propagating data towards the warehouse. Second, it is not obvious how the active propagation of data can be implemented, especially in the presence of legacy production systems. The problem becomes worse since it is rather improbable that the software configuration of the source systems can be significantly modified to cope with the new task, due to (a) the down-time for deployment and testing, and, (b) the cost to administrate, maintain, and monitor the execution of the new environment.

A more pragmatic approach involves a semi-automated environment, where user requests for freshness and completeness are balanced against the workload of all the involved sub-systems of the warehouse (sources, data staging area, warehouse, data marts) and a tunable, regulated flow of data is enabled to meet resource and workload thresholds set by the administrators of the involved systems [1].

The general ETL architecture of a near real time data warehouse consists of database sources of different types [7] including extraction tool, which pushes extracted data into temporary store.

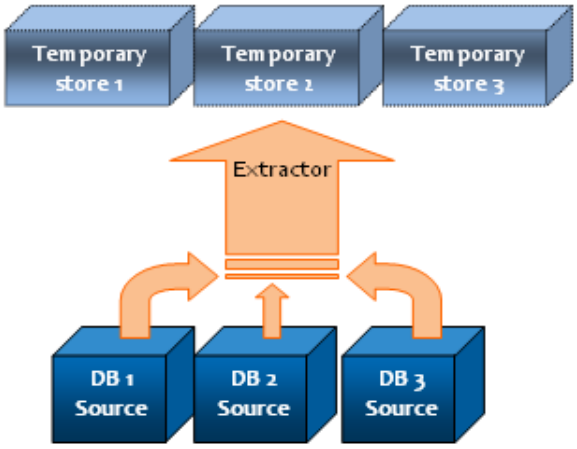

Figure 1. Extraction – part of an ETL process

Then it prepares data for transformation process into transformation function – ready data format. Transformation runs in DPA (Data processing area) where data are transformed and cleaned and after then are data exported by transformation function. Our research is focused on Near real time ETL improvement, using new compensation parts constructed in Matlab environment, showed on Figure 2. The difficulty of such a simple Transform function is to keep it simple. As though they can exert a vacuum of complexity, simple Transform functions attract additional functions and complexity to them. Resist this temptation at all costs. A simple Transform function is another beautiful and elegant design, and should be allowed to remain that way.[2]

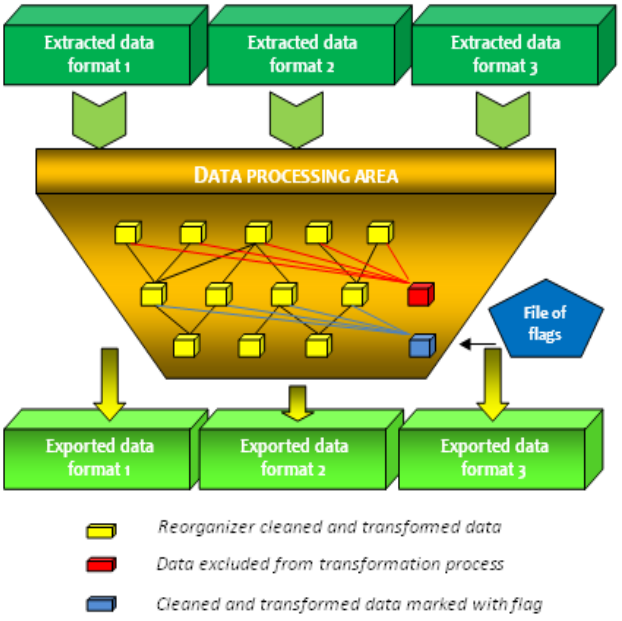

Figure 3. Transformation – part of an ETL process

Whole process in DPA runs automatically even there is no reason for excluding the data, like reference error or the system is overloaded due to high refreshment rate or high number of users. Each situation should be tested first. [4]

Loader then loads data into data warehouse fact and dimension tables. Whole process shows Figure 3.

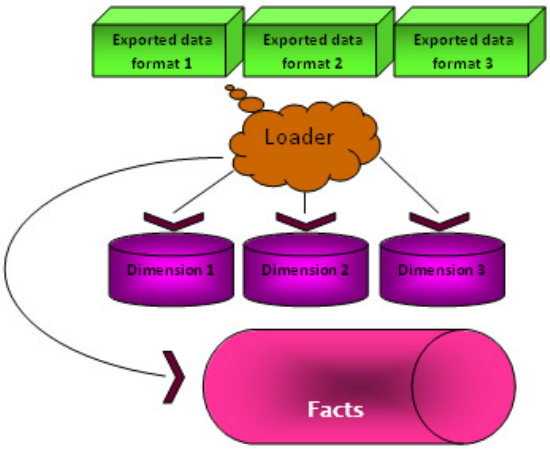

Figure 3. Loading – part of an ETL process

On this architecture is based traditional ETL. In case of near real time ETL there are built in compensation structures, which alleviates impact of high frequently refreshment. In a real world this cannot be performed, due to many possible reasons like high number of users, high rate refreshment or too expansive software and hardware parts, and this situation is solved by several technical and structural accessories. Practically it leads to compensated schema which contains complementary parts.

There is situated also a file of flags, containing file of conditions edited by administrator, that are used for additional inspection. Excluded and marked data are evaluated as well as reorganized, cleaned and transformed data. After then is calculated level of trust all outgoing data, which describes on how much valid the data are. It is showed on Equation (1).

Of course, for each data row is also available the pertinent time, so level of trust should be calculated for a certain time interval.

$$[\%] = 100 - \frac{\left(\text{Data excluded from transformation process} + \text{Cleaned and transformed data marked with flag}\right) \cdot 100}{\text{Reorganized cleaned and transformed data}} \quad (1)$$

This model of improved near real time ETL should be applied on the either ETL approach, and also should be applied on sequential, pipelining and partitioning execution of ETL process too described in [1].

Transformation process is reorganizing data according to DWH needs, transforming measuring units, performing reference assignment, modifying key values, sorting products in sections and so on. File of flags contains exceptions for transformation process, like certain company (customer) assignment, some product exceptions, which are stored in "Exported data format for errors" (eDFE). Loading process then pushes exported data into DWH as well as exported

"error" data, which are marked with flag in the fact table (Profit).

If any data are needed for analyzing, level of trust is calculated for specified data selection.

This way we can determine the level of trust for analyzing data, while the (near) real time ETL process is maintained.

ETL PROCESS IN MATLAB ENVIRONMENT

Whole DPA process runs via routines constructed in Matlab environment, which the transformation makes more transparent and simple. Matlab contains many functions, which can be used to convert data into specified format. Example of the transformation process is showed below.

```
% Data Sources Extraction
% Reading arguments and storeing to temporary store
AA = xlsread('*.xls', 'field margins eg. A2:A1194);
BB = xlsread('*.xls', 'field margins eg.C2:C1194);
CC = xlsread('*.xls', 'field margins eg.A2:A203);
DD= xlsread('*.xls', 'field margins eg.E2:E203);
EE = xlsread('*.xls', 'field margins eg.C2:C988);
FF = xlsread('*.xls', 'field margins eg. B2:B988);
%Cleaning data and calculating field sizes
cycle1 = size(EE);
cycle1(1);
%Transformation process area
%Assignment data and searching for foreign key values
for i=1:cycle1(1)
   j=1;
   while EE(i)  ~= AA(j) & j<cycle2(1)
      j=j+1;
   end;
   O(i,1)=BB(j);
   j=1;
   while FF(i)  ~= CC(j) & j<cycle3(1)
      j=j+1;
   end;
   P(i,1)=DD(j);
end;
%Storeing foreign key values to Exported data format
xlswrite(*.xls', 'field margins eg. O, BB, P);
%Loading  process
% Reading arguments for loading process
%Assignment dimensions to facts
Field1 =    xlsread('*.xls', 'field margins eg. A2:A1681');
Field2 =    xlsread('*.xls', 'field margins eg. G2:G1681');
%Calculating field sizes
cycle1 = size(Field1);
%Storeing to data warehouse
xlswrite('*.xls', Field2,'sheet name','begnning of the field storeig eg. G2');
```

CONCLUSIONS

Managing ETL process during different phases must be clear and explicit. Although data from data sources can be in different formats and quality conditions, ETL must significantly assign each input to any output data.

Improved ETL process together with appropriate development environment and toolkit helps manage extraction, transformation and finally delivering data to data warehouse.

REFERENCES

[1.] S. KOZIELSKI, R. WREMBEL, "New Trends in Data Warehousing and Data Analysis". Springer; 2009. ISBN 978-0-387-87430-2

[2.] L. REEVES, "A Manager's Guide to Data Warehousing", Published by Wiley Publishing, Inc.; 2009. ISBN: 978-0-470-17638-2

[3.] F. SIVERS, "Building and Maintaining a Data Warehouse", CRC Press; 2008. ISBN 978-1-4200-6462-9

[4.] P. TANUSKA, O. MORAVCIK, P. VAZAN, "The base testing activities proposal". In: DAAAM International Vienna 2009: No. 1 Annals of DAAAM for 2009 & Processings of the 20th International DAAAM Symposium. 25-28 November 2009, Vienna, Austria. ISSN 978-3-901509-70-4

[5.] M. KEBISEK, P. SCHREIBER, I. HALENAR, "Knowledge Discovery in Databases and its application in manufacturing". In International workshop Innovation Information Technologies – Theory and Practice; 2010 September 06-10, Dresden, Germany, pp. 204-207. ISBN 978-3-941405-10-3

[6.] A. TRNKA, "RFM Analysis as a Part of DMAIC Phases". In: 2011 International Conference on Database and Data Mining (ICDDM 2011) : 25-27, March, 2011 : Proceedings / editors: Steve Thatcher and Liu Guiping. - Sanya : IEEE, 2011. - 1 CD-ROM. - ISBN 978-1-4244-9610-5. - S. 278-281.

[7.] M. STRÉMY, A. ELIÁŠ: Virtual laboratory communication. In: Annals of DAAAM and Proceedings of DAAAM Symposium. - ISSN 1726-9679. - Vol. 20, No. 1 Annals of DAAAM for 2009 & Proceedings of the 20th international DAAAM symposium "Intelligent manufacturing & automation: Focus on theory, practice and education" 25 - 28th November 2009, Vienna, Austria. - Vienna: DAAAM International Vienna, 2009. - ISBN 978-3-901509-70-4, s. 0139-0140

ACTA TECHNICA CORVINIENSIS – BULLETIN of ENGINEERING

ISSN: 2067-3809 [CD-Rom, online]