Algorithms and Architecture for Managing Evolving ETL Workflows

Judith Awiti(✉)

Université Libre de Bruxelles, Brussels, Belgium
judith.awiti@ulb.ac.be

Abstract. ETL processes are responsible for extracting, transforming and loading data from data sources into a data warehouse. Currently, managing ETL workflows has some challenges. First, each ETL tool has its own model for specifying ETL processes. This makes it is difficult to specify ETL processes that are beyond the capabilities of a chosen tool or switch between ETL tools without having to redesign the entire ETL workflow again. Second, a change in structure of a data source leads to ETL workflows that can no longer be executed and yields errors.

Therefore, we propose a logical model for ETL processes that makes it feasible to (semi-)automatically repair ETL workflows. Our first approach is to specify ETL processes using Relational Algebra extended with update operations. This way, ETL processes can be automatically translated into SQL queries to be executed into any relational database management system. Later, we will consider expressing ETL tasks by means of an Extensible Markup Language (XML) and other programming languages. We also propose the Extended Evolving-ETL (E3TL) framework in which we will develop algorithms for (semi-) automatic repair of ETL workflows upon data source schema changes.

Keywords: ETL · E3TL · Database · Data warehouse · Relational algebra

1 Introduction

There is no agreed-upon conceptual or logical model to specify ETL processes. Consequently, existing ETL tools use their own specific language to define ETL workflows. For this reason, several conceptual and logical models [3,6–9] have been proposed for ETL process design. Unfortunately, each model has some limitations discussed in the related work and hence does not provide suitable solutions to the problem. Moreover, it is our aim to have a model that makes ETL workflow reparation easy and feasible.

Not only that, most of the existing ETL tools tacitly assume that the structure of every DS is static, but such an assumption is incorrect. Data source (DS) schema changes are bound to happen, and they disrupt the smooth execution of

T. Welzer et al. (Eds.): ADBIS 2019, CCIS 1064, pp. 539–545, 2019.
https://doi.org/10.1007/978-3-030-30278-8_51

ETL workflows resulting in data loss. For example, Wikipedia had 171 schema versions from April 2003 to November 2007 [1]. A change in structure of a DS leads to ETL workflows that can no longer be executed and yields errors. Large organizations might have thousands of deployed ETL workflows which makes it nearly impossible for ETL designers to repair ETL workflows after every DS change.

Considering this, we study BPMN4ETL [2], a conceptual model for designing ETL processes based on the Business Process Modeling Notation (BPMN) a de-facto standard for specifying business processes. The model provides a set of primitives that cover usual ETL requirements. Since BPMN is already used for specifying business processes, adopting this methodology for ETL would be smooth for users already familiar with that language. Further, BPMN provides a conceptual and implementation-independent specification of such processes, which hides technical details and allows users and designers to focus on essential characteristics of such processes. ETL processes expressed in BPMN can be translated into executable specifications for ETL tools. Out of this model we develop a logical model based on relational algebra (RA). Being a well-studied formal language, RA provides a solid basis to specify ETL processes for relational databases, and its expressiveness allows providing a detailed view of the data flow of any ETL process. We prove the efficiency of using an extended RA to specify ETL processes at a logical level as compared to commercial tools with the TPC-DI benchmark [5]. Later, we may consider expressing ETL tasks by means of an Extensible Markup Language (XML) and other programming languages.

After that, we will develop algorithms for our proposed E3TL framework to (semi-) automatic reparation of ETL workflows. In these algorithms, we will apply rule-based reasoning (RBR) and case-based reasoning (CBR) and, a combination of both. Then, we will develop a prototype system for ETL workflow reparation based on the above-mentioned framework and verify the applicability of our proposed solution with the TPC-DI benchmark. The project has two main objectives:

- To propose a methodology for designing ETL processes that will facilitate a smooth transition from gathering user requirements to the actual implementation. This requires expressing ETL processes at conceptual, logical, and physical levels.
- To develop a framework to (semi-) automatically repair ETL workflows upon DS changes.

Currently, we have been working on the first objective of the project.

2 Related Work

The conceptual model proposed in [8] analyzes the structure and data of the DSs and their mapping to a target data warehouse (DW). Attributes are treated as *first-class citizens* in the inter-attribute mappings. In [9], a logical model was developed independent of this conceptual model. The authors modelled an ETL

workflow as a graph which they named the *architecture graph*. All entities in an ETL scenario including data stores, activities and their constituent parts are modelled as nodes of the graph. These models are complex and not easy to understand or design because they do not utilize a standard modelling language. Apart from that, the logical model cannot be implemented directly in a DW environment unless via a custom-built tool. Another conceptual model of ETL processes developed by [7], uses the Unified Modeling Language (UML) to design ETL processes. Each ETL process is represented by a stereotyped class which is in turn represented by an icon. Because UML is a widely accepted standard for object-oriented analysis and design, this approach minimizes the efforts of developers in learning new diagrams for It was refined in [3] with the use of UML activity diagrams instead of class diagrams and icons. Recently, RA has been introduced as a logical model for ETL processes [6]. RA provides a set of operators that manipulates relations to ensure that there is no ambiguity. Even though loops and certain complex functions cannot be specified directly with RA, it has enough operations to model common ETL processes when extended with update operations. It can also be directly translated into SQL to be executed in any relational database management system (RDBMS).

Although little research has been done in the field of evolving ETL workflows, a recent one in [10] suggest a solution to the problem by using case-based reasoning. The authors developed a framework for solving the problem of evolving ETLs by applying user-defined rules and case-based reasoning. The user-defined rules approach is independent of the case-based reasoning approach. A previous research [4] also provided a framework in which the ETL workflow is manually annotated with rules that defines its behavior in response to data source changes. Since both approaches do not provide comprehensive solutions, the problem of evolving ETL workflows still needs substantial research.

TPC-DI [5], the first industry benchmark for Data Integration (also known as ETL) provides a suitable case study for this research. The source data model used was based on the relational data of an Online Transaction Processing (OLTP) database of a fictitious retail brokerage firm and some other external sources. The target data model has a snowstorm schema. The source system is made up of an Online Transaction Protocol Database (OLTP DB), a Customer Relation System (CRM) and a Human Resource (HR) System. Some data in the source system are static and only required to be loaded during the historical load. Examples of such static data of the dataset are date/time, industry segments, trade types and tax rates. Other non-static data such as the customer data are loaded in two additional incremental loads. The source file comprises a variety of data formats including XML, CSV and data dumps.

3 Our Approach

3.1 Modelling ETL Processes

We begin our design of ETL processes with the BPMN4ETL [2] conceptual model. This model customized from BPMN improves communication and

validation between various stakeholders in a DW project. It serves as a standard means of documentation. It being technology-independent provides several paths to implementing ETLs including RA (which can be directly translated into SQLs), XML (which can be extended as in interchange format), and commercial tool specifications. In our approach, we develop a logical model with RA extended with update operations. RA provides a set of operators that manipulates relations to ensure that there is no ambiguity. This unambiguity will make it easier for us to repair the logical model upon DS schema changes. RA can be directly translated into SQL queries which is faster than commercial tool implementations. Commercial tools mostly add another layer for manipulations of ETL processes before finally SQL queries are generated to be executed on RDBMSs. We intend to implement ETL processes directly in the RDBMS environment from our logical model.

We explain our modelling approach with Fig. 1, the ETL process that loads the DimBroker dimension table of the TPC-DI. Two input data tasks inserts records from an HR.csv file (a DS file) and DimDate (a dimension table in the TPCDI DW) into the ETL flow. An aggregate task filters only records with EmployeeJobCode value of 314. Records from DimDate are aggregated to obtain the record with the minimum value in the DateValue column. Records of both paths are joined by a right outer join. Note that the condition of the join is just to ensure that each record at S3 is joined to the date value. A surrogate key column with the value of the row number is added the the ETL flow and finally, an insert data task inserts the records into DimBroker.

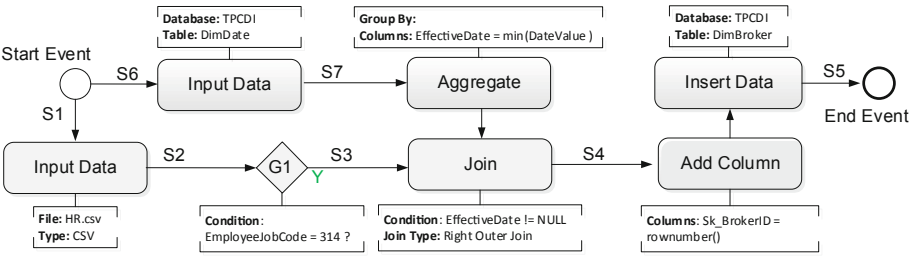

Fig. 1. Load of DimBroker dimension table

Now, we translate the BPMN model in Fig. 1 into our extended RA logical model in Fig. 2. Equations 1 and 2 stores records from HR.csv and DimDate into TempHR and TempDate temporary tables. Equation 3 selects records with EmployeeJobCode = '314' into Temp1. An aggregate operation retrieves the record with the minimum value in the DateValue column into Temp2 (Eq. 4). Temp1 and Temp2 are joined by a right outer join and the results are stored in Temp3 (Eq. 5). A surrogate key column (SK_BrokerID) is added in Eq. 6 and the records are inserted into DimBroker (Eq. 7).

Currently, we are working on BXF, an Extensible Markup Language (XML) interchange format for BPMN4ETL which is to express and interchange

BPMN4ETL model information across tools that are compliant with BPMN 2.0. This XML interchange format does not only describe the control flow of ETL processes but also the data flow.

3.2 ETL Workflow Reparation

RBR is a type of reasoning that uses (if-then-else) rule statements whereas CBR means understanding and solving a new problem by remembering how a similar old problem was solved. A problem and its solution could be described as a case. Case-based management systems learn by acquiring knowledge from their knowledge base and they also augment the knowledge base with newly developed knowledge. The overall system architecture may look as depicted in Fig. 3.

An ETL parser takes an ETL workflow in the form of RA or SQLs and parses the parts of each command of the workflow. The ETL manager assesses the impact of the DS change on each command of the ETL workflow. For DS changes that can be handled by applying rules stored in the rule base, the ETL Rewriter rewrites the commands in the ETL workflow by applying recommendations from the ETL manager. It iterates trough the commands and makes changes to each part. It also stores the history of versions of the ETL workflow. Therefore, it can rewrite the workflow into an old version if the new version becomes problematic. For situations that the ETL manager does not find any suitable rules, the ETL Rewriter waits for the user's input to rewrite ETL commands. The Rule Base contains distinct rules based on conditions. In the beginning of this framework implementation, there may be few rules in the rule base but with time, the rules are increased by the translation of cases into rules. Translation of cases into rules are done by the translator component of the framework. There are different ways in which rules can be inferred from cases. One of them is to make the translator check the case base anytime there is a new case, if it is possible to infer rules from the cases in the case base. Another is to wait for a specific amount of time or a specific number of new cases to be added to the case base before rules are inferred from cases. The choice of method of rule inference will depend on which one provides a more efficient and accurate framework.

For situations where no rules were found, or several solutions are applicable with the rules the user decides how the problem should be handled. The ETL rewriter rewrites the ETL commands in the ETL workflow by applying the user's

$$\text{TempHR} \leftarrow \mathcal{I}_{\text{EmployeeID,ManagerID,FirstName,}\ldots}(\text{HR.csv}) \quad (1)$$

$$\text{TempDate} \leftarrow \mathcal{I}_{\text{Sk\_dateid,Datevalue,Datedesc,}\ldots}(\text{DimDate}) \quad (2)$$

$$\text{Temp1} \leftarrow \sigma_{\text{EmployeeJobCode = '314'}}(\text{TempHR}) \quad (3)$$

$$\text{Temp2} \leftarrow \mathcal{A}_{\text{EffectiveDate=min(DateValue)}}(\text{TempDate}) \quad (4)$$

$$\text{Temp3} \leftarrow \text{Temp1} \bowtie_{\text{EffectiveDate} \neq \text{NULL}} (\text{Temp2}) \quad (5)$$

$$\text{Temp4} \leftarrow \mathcal{E}_{\text{SK\_BrokerID = rownumber()}}(\text{Temp3}) \quad (6)$$

$$\text{DimBroker} \leftarrow \text{DimBroker} \cup (\pi_{\text{SK\_BrokerID, EmployeeID,ManagerID,FirstName,}\ldots}(\text{Temp4})) \quad (7)$$

Fig. 2. RA expressions to model the ETL for DimBroker

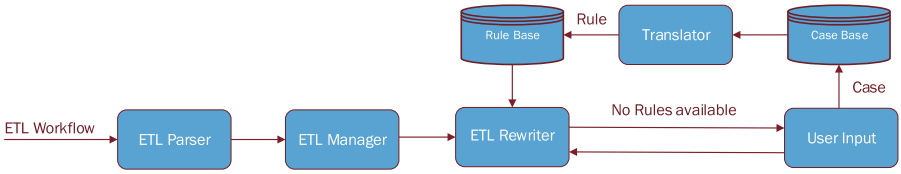

Fig. 3. Framework for ETL workflow reparation (E3TL)

decision. The DS change and its solution are stored in a knowledge base called the case base. The translator component applies algorithms to develop distinct rules from cases. These rules are stored in the rule base and are used to handle similar DS changes in the future.

3.3 Results

In this section we discuss the results of the experimental evaluation of our approach in modelling ETL processes. We implemented TPC-DI first, with our approach from BPMN4ETL into extended RA (using Postgres PLSQL) and second, with BPMN4ETL translated directly into the Pentaho Data Integration (PDI) tool. Both tests were run over an Intel i7 computer, with a RAM of 16 GB, running the Windows 10 Enterprise OS, using PostgresSQL as the data warehouse storage. The experiments showed that the SQL implementation runs orders of magnitude faster than that of PDI for all scale factors (SF). For example, the total execution time for the historical load of SF 3 in PLSQL was 12:50 min whereas that of PDI was 11 h and 23:52 min. Part of the reason for this poor performance of PDI is because it handles loops with the Copy Rows to Result step, that stores the rows in memory first and retrieves them one row at a time.

4 Conclusion

ETL process development is one of the complex and costly part of any data warehouse (DW) project. Currently, managing ETL workflows has some challenges. First, each ETL tool has its own model for specifying ETL processes making it is difficult to specify ETL processes that are beyond the capabilities of a chosen tool or switch between ETL tools without having to redesign the entire ETL workflow again. Second, a change in structure of a data source (DS) leads to ETL workflows that can no longer be executed and yields errors. This project provides a means of managing ETL processes in two ways. First, their modelling and second, their reparation upon DS schema changes. In the former, we model ETL processes with BPMN4ETL, an extended BPMN model for ETL at the conceptual level and with relational algebra (RA) extended with update operations at a logical level. With the later, we propose the E3TL framework in which we will develop algorithms for (semi-) automatic repair of ETL workflows

upon DS schema changes. In order to assess the performance of our RA approach, we experimentally evaluated it on the TPC-DI benchmark. The results showed that our RA implementation is orders of magnitude faster than that of the PDI tool, for all scale factors. In future, we will evaluate the performance of our approach with other ETL tools such as Microsoft SSIS and Talend, and handle DW changes. We will also extend the E3TL framework to handle NoSQL environments.

Acknowledgements. The work of Judith Awiti is supported by the European Commission through the Erasmus Mundus Joint Doctorate project *Information Technologies for Business Intelligence-Doctoral College* (IT4BI-DC).

References

1. Curino, C., Moon, H.J., Tanca, L., Zaniolo, C.: Schema evolution in Wikipedia: toward a web information system benchmark. In: Proceedings of the 10th International Conference on Enterprise Information Systems, ICEIS 2008. pp. 323–332. Citeseer, Barcelona, Spain (2008)
2. El Akkaoui, Z., Zimányi, E.: Defining ETL worfklows using BPMN and BPEL. In: Proceedings of the ACM 12th International Workshop on Data Warehousing and OLAP, DOLAP 2009. pp. 41–48. ACM, Hong Kong, China (2009)
3. Muñoz, L., Mazón, J.-N., Pardillo, J., Trujillo, J.: Modelling ETL processes of data warehouses with uml activity diagrams. In: Meersman, R., Tari, Z., Herrero, P. (eds.) OTM 2008. LNCS, vol. 5333, pp. 44–53. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-88875-8_21
4. Papastefanatos, G., Vassiliadis, P., Simitsis, A., Sellis, T., Vassiliou, Y.: Rule-based management of schema changes at ETL sources. In: Grundspenkis, J., Kirikova, M., Manolopoulos, Y., Novickis, L. (eds.) ADBIS 2009. LNCS, vol. 5968, pp. 55–62. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-12082-4_8
5. Poess, M., Rabl, T., Jacobsen, H., Caufield, B.: TPC-DI: The first industry benchmark for data integration. Proc. of the VLDB Endowment **7**(13), 1367–1378 (2014)
6. Santos, V., Belo, O.: Using relational algebra on the specification of real world ETL processes. In: Proceedings of the 13th IEEE International Conference on Dependable, Autonomic and Secure Computing, DASC 2015. pp. 861–866. IEEE, Liverpool, UK (2015)
7. Trujillo, J., Luján-Mora, S.: A UML based approach for modeling ETL processes in data warehouses. In: Song, I.-Y., Liddle, S.W., Ling, T.-W., Scheuermann, P. (eds.) ER 2003. LNCS, vol. 2813, pp. 307–320. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-39648-2_25
8. Vassiliadis, P., Simitsis, A., Skiadopoulos, S.: Conceptual modeling for ETL processes. In: Proceedings of the 5th ACM International workshop on Data Warehousing and OLAP, DOLAP 2002. pp. 14–21. ACM, McLean, Virginia, USA (2002)
9. Vassiliadis, P., Simitsis, A., Skiadopoulos, S.: Modeling ETL activities as graphs. In: Proceedings of the 4th International Workshop on Design and Management of Data Warehouses, DMDW 2002. pp. 52–61. CEUR-WS.org, Toronto, Canada (2002)
10. Wojciechowski, A.: ETL workflow reparation by means of case-based reasoning. Inf. Syst. Front. **20**(1), 21–43 (2018)