Enhancing Conceptual Modeling for Distributed Data Management: Introducing ER+

Gonçalo Carvalho[1][0000−0001−7095−5003], Bruno Cabral[1][0000−0001−9699−1133], Jorge Bernardino[2][0000−0001−9660−2011], and Vasco Pereira[1][0000−0002−4225−9075]

[1] University of Coimbra, Centre for Informatics and Systems of the University of Coimbra, Department of Informatics Engineering, Portugal
gcarvalho@dei.uc.pt, bcabral@dei.uc.pt, vasco@dei.uc.pt
[2] Polytechnic of Coimbra, Coimbra Institute of Engineering (ISEC), Coimbra, Portugal
jorge@isec.pt

Abstract. The Entity-Relationship (ER) diagram has long been a fundamental tool for Conceptual Modeling (CM) in database systems. However, its limitations in representing crucial aspects of Distributed Data Management (DDM), such as diverse data locations, data transformation, and transport, have prompted the development of ER+. ER+ is a modeling language that extends the ER model by incorporating these essential features.

In this paper, we highlight the advantages of ER+, expressly its ability to represent distinct data locations visually, incorporate data transformation operations, and accommodate data transport features. By enabling developers to define storage locations directly within the conceptual model, ER+ significantly streamlines the deployment process.

Keywords: Data transformation, Data transport, Conceptual Modeling Languages

1 Introduction

Data modeling plays a crucial role in representing information and its relationships. The ER model, introduced by Chen in 1976 [2], provides a unified view of structured data. However, existing ER notations cannot represent crucial concepts, such as data transport, data transformation and distribution, and information generation. This limitation hinders the modeling of distributed systems and the adaptation to new computation paradigms, such as edge and cloud computing.

The ER+ modeling language aims to solve these challenges. ER+ extends the original ER model and enables the representation of data dynamics in distributed and multilayer systems. However, ER+ focuses on conceptual representation and does not address physical implementation aspects, such as data consistency, security, scalability, and availability.

The remainder of this article is organized as follows: Section 2 summarizes the ER+ modeling language and its main features. And Section 3 concludes this work and presents future research directions.

2 The ER+ modeling language

ER+ was proposed in a previous work [1]. ER+ expands the ER model with new features for DDM and is based on the Crow's Foot notation. ER+ allows:

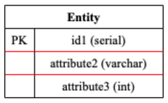
– *Integration of multiple data locations* in the same data model. A location may be a different database, a physical machine, or containers (deployed in the same server or distributed).
– *Functional relationships*, which can represent data transformation and data transport amongst locations (for example, entities or databases), including line functions and group functions.

Table 1 provides a summary of each ER+ concept and corresponding symbol.

Functional relationships are the basis of our proposal and the main concepts are **data transformation**, from a set of data S it produces another set of data S' as a result of a function f (S' can be totally different from S, it can have a different number of tuples and/or attributes); and **data transport**, moving a set of data from A to B, after applying the transformation functions. There are two types:

– *Data transport with group definitions and data transformation:* it is possible to set group definitions and aggregate functions. In group definitions, the user sets the granularity of data. The aggregate functions, such as average, count, maximum, minimum, standard deviation, sum, and variance will aggregate the attributes' values. In addition, the attribute(s) used in the group definition will be the Primary Key(s) in the destination entity.
– *Data transport without group definitions and with data transformation:* in these cases, there is not a group definition. There are two possibilities for data transformation:
  - combine lines, such as concatenate, subtract, and multiply attribute values. For example, a line function (*multiplication*) between the two attributes, in a *top-down* order (despite not having importance in this particular operation), but with a line identifier (the attribute with a dashed line). Also, a solid line with a vertical line near the triangle will represent a new attribute after the transformation.
  - set a group function for all values of an attribute. For example, a group function *count*, and this will store in the new entity the *count* of all the entity's attributes, creating a single instance. Because no group definition is set, and the destination entity needs a Primary Key, this will be set automatically, and the attribute will be set as *"id"*.

Table 1. ER+ concepts and symbols

Name	Summary	Symbol
Locations definition	Sets a boundary between the data source and the data destination after transport. It may represent distinct physical locations, nodes, servers, containers, or a different database in the same server. The number of possible boundaries is only limited by the designer's needs.	▪ ▪ ▪ ▪ ▪
Entity notation add-on	A line will separate all attributes to unambiguously identify the attribute as the source or target of the functional relationship by which contains its start or end. In the case of a *date* type attribute, it may include the time granularity for data.	 Entity; PK id1 (serial); attribute2 (varchar); attribute3 (int)
Functional relationships	Group definitions (allow applying aggregation functions to subgroups of tuples in a relation).	○
	Aggregate functions (uses a set of tuples as an argument and generates a value for each aggregate set). Inside the symbol (half circle), the designer sets the type of function.	◗
	Line functions (it is a function that is applied to the linked attributes. The order of the operation is applied by a *top-down* approach, which means that the items on top will be on the left of the operation). Inside the symbol, the designer sets the type of the line function.	▷
	Inputs for the data operations (used only in the aggregate functions or line functions).	——
	Inputs for the data operations (used in the aggregate or line functions, but also become an attribute in the destination entity).	——+
	Inputs for the group definitions or line identifiers (attributes that will set subgroups of data).	- - - - - - - - -
	Data transport (moving a set of data from A to B, after applying the transformation functions. This solid line with an arrow must cross the boundary set by the *Locations definition*).	←

All the data transformations introduced in our study used the logic and rules of relational algebra, which can be translated into SQL. Proof of equivalence between SQL and relational algebra can be found in previous works by Guagliardo et al. [3]. Data transport model features were not provided with a relational algebra equivalent, as they only imply data extraction and loading, with no associated transformations.

3 Conclusions

Data modeling is crucial in representing information and its relationships. The traditional ER notations have limitations in capturing important concepts for distributed systems. The ER+ modeling language addresses these challenges by extending the ER model and enabling the representation of data dynamics in distributed and multilayer systems.

This article highlights the potential of ER+. By visually representing different data locations and incorporating data transformation and transport operations, ER+ streamlines the implementation process for DDM systems developers. However, it's important to note that ER+ primarily focuses on the conceptual level and does not directly address the physical implementation aspects such as data consistency, security, scalability, and availability. We will integrate the physical elements of the implementation in our future work.

Acknowledgments

This work was partially supported by the Portuguese Foundation for Science and Technology (FCT) under Ph.D. grant 2022.12090.BD, and within the scope of the project CISUC - UID/CEC/00326/2020 and by the European Social Fund, through the Regional Operational Program Centro 2020.

References

1. Carvalho, G., Bernardino, J., Pereira, V., Cabral, B.: Er+: A conceptual model for distributed multilayer systems. IEEE Access **11**, 62744–62757 (2023). https://doi.org/10.1109/ACCESS.2023.3287796
2. Chen, P.: The Entity-Relationship Model—toward a Unified View of Data. ACM Transactions on Database Systems (TODS) **1**(1), 9–36 (1976)
3. Guagliardo, Paolo and Libkin, Leonid: A formal semantics of SQL queries, its validation, and applications. Proceedings of the VLDB **11**(1), 27–39 (2017). https://doi.org/10.14778/3136610.3136613