Entity Relationship Stored Procedure Meta Language

Călin-Adrian Comes, Ioan Rus
Petru Maior University
Financial-Accounting Department,
Nicolae Iorga 1, 540088,
Târgu-Mureş, Mureş county, România

Nicolae Ghişoiu, Paul Vasile Breşfelean
Babeş-Bolyai University
IT Applied for Economics Department,
Mihail Kogălniceanu 1, 400084,
Cluj-Napoca, Cluj county, România

Abstract

ER-SPML (Entity Relationship-Stored Procedures Meta Language) is proposed to be independent by RDBMS team players in idea to support their own SQL dialects on Platform Specific Language, versus the conceptual level Platform Independent Language. The metalanguage proposes a new operation called Stored Procedures and the language behind him in idea to be platform independent in relation with SQL dialects and their Procedural Languages. The concept of stored procedures operation is abstract, helps in syntactic and semantic modeling, and is required in physical implementation. A stored procedures operation along with his language captures the syntactic and semantics of an RDBMS schema in his dynamical evolution, not in a statical way like ER diagram. An case study of database design with ER-SPML and description using the metalanguage and the diagrammatic technique is given.

Key-Words: Entity Relationship - Stored Procedures Language(ER-SPL), Entity, Attribute, Relationship, Relational Database Management System(RDBMS), Data Definition Language (DDL), Data Manipulation Language(DML), Data Control Language(DCL)

1 Introduction

In 1976, Peter Pin-Shan Chen with ER [5] describe a model that serves as the foundation of many systems analysis and design methodologies, repository systems, and CASE tools from commercial software vendors to free source world. A preliminary framework for Entity-Relationship model was define later in [6].

Any important article from journals and books from respectable publishing houses where the **modeling** word is a key word the ER Model or ER Model flavors are just in place. During the time because there is no standard for ER Model there are a lot confusions regarding the definition of the entity. Bernhard Thalheim in [23] cite more than twelve different approach for defining the entity notion with the following remark "The confusion is almost since most of the database and software engineering books do not define at all the notion of entity ".

OCL [15, 17] was formally developed as a business language with roots in Syntropy, second generation language object-oriented analysis and software design method developed at Object Designers Limited in the UK during the early 1990s, used to describe UML models.

In this paper we try to present the Entity Relationship-Stored Procedure Meta Language in idea to extend : Data Definition Language(DDL), Data Manipulation Language(DML), and Data Control Language (DCL) in Object Constraint Language(OCL) style to describe the Entity Relationship-Stored Procedure diagrammatical artifacts.

2 State of the art of ER Meta models

ER Model has been redefined because of his popularity with ER Meta models like EER (Extended Entity Relationship)[10] , CSL (Conceptual Schema Language) [4], HERM (Higher-Order Entity Relationship Model) [23], REMORA [20], and Barker [1].

2.1 Extended Entity Relationship (EER) Meta model

David W. Embley and Tok Wang Ling in [10] extend the ER Model with EER Meta-model with the following approaches: capturing the real world semantic, transform the EER Meta model in to a normalized EER Meta model and generate normalized relations.

Entity-Relationship (ER) models and extended Entity-Relationship (EER) models has limitations and problems:

- Entity-Relationship (ER) models and extended Entity-Relationship (EER) models require designers to distinguish between attributes and entities. This can cause downstream redesign when attributes and entities are mismatched.

- Before the transformation of the ER model to relations designer work with ER diagrams, but after the transformation they work with relation schemes. These relations may need normalization. Normalization is done through a combination of decomposition and synthesis techniques.

Their approach are to capture the real-world semantics in an improved EER model, transform the EER model to a normalized EER model, and generate normalized relations.

2.2 A Language for Defining Conceptual Schema (CSL)

In [4] B. Breutmann, R. Falkenberg and E. Mauer describe a high level data definition language, CSL(Conceptual Schema Language), for defining conceptual schemes. The language provides descriptive and procedural elements, so static aspects and dynamic behavior of data can be described. The proposal CSL provides: standard types, object types and association types:

- Standard operations are: $=, <, >, +, -, *$ and standard types are like INTEGER, REAL, CHAR, STRING, DIGIT

- Within an object type only those characteristics can be described which hold for all objects of a certain type

- The association type definition consists of the unique association type name, the participating object types together with their roles and simple occurrence frequencies and the identifier of the association type together with the candidate identifiers.

2.3 HERM (Higher-Order Entity Relationship Model) Meta model

B. Thalheim introduces the Higher-Order Entity Relationship Meta model in [23, 24], called HERM, providing an interesting conceptual data model, but it is strictly founded in theory.

The Higher-Order Entity Relationship Model is an extension of the Entity Relationship Model. Schemata in the Higher-Order Entity Relationship Model can be mapped automatically to relational database schemata. One key feature of the HERM is the nesting of attributes.

2.4 REMORA Meta model

In [20] the REMORA methodology and an expert design tool, that supports the design of information systems, are described.

The REMORA methodology provides a consistent set of models, languages, methods and software tools to design and implement large and semantically complex information systems. The conceptual models use, there is a more direct and efficient interaction between the designers and the users, this allows the definition of the information system conceptual schema.

2.5 Barker Meta model

The Barker model was introduced by R. Barker in [1] and modified slightly by the Oracle ^{TM}Corporation . The pedantic problem with the Barker model is that one needs to fully understand relational database theory to understand why the Barker model is done the way it is. We present the Barker model here because the way it unfolds is a bit different from the Chen model. The Chen model focuses on modeling data, whereas the Barker model adapts the data to the relational database concurrently with the design. Therefore, the ER design methodology for the Barker model will develop differently from the Chen model. Further, the Barker model does not have some of the conventions used in the Chen model. The Barker model does not directly use the concept of composite attributes, multi-valued attributes, or weak entities, but rather handles these concepts immediately in light of the relational model. Because the Barker model is so close to the relational model to begin with, the mapping rules are trivial the mapping takes place in the diagram itself. A Barker model uses soft boxes for entities (with the entity name in capital letters), and there is a line separating the entity name from the attributes (and the attribute names are in lowercase letters). A Barker model does not place the attributes in ovals (as the Chen model does), but rather lists the attributes below the entity name. All attributes in a Barker model are considered simple or atomic, as in relational databases.

The model does not have the concept of composite attributes. In Barker model, the primary key has a # in front of the name of the attribute.

A primary key has to be a mandatory attribute in a relational database, but again, all mandatory attributes here are not necessarily unique identifiers. In the Barker model, a relationship is represented by a line that joins two entities together. There is no diamond denoting the relationship (as we saw in the Chen model). The relationship phrase for each end of a relationship is placed near the appropriate end (entity) in lower case.

2.6 Data Schema Integration Meta model

In 1983 many problems of data integration are addressed but not all of them and the whole area of data integration is not at a mature stage.

In [2] C. Batini and M. Lenzerini describe a methodology for data schema integration consists of three steps: conflict analysis, merging and final enrichment and restructuring of the schema.

In the conflict analysis phase all incoherences should be detected and solved. The main tasks are naming conflicts analysis, resolution of synonymy and homonyms and the modeling compatibility analysis. In the merging phase schemata are merged into a unique draft integrated schema. The main tasks of the third phase are analysis of inter schema properties, analysis of redundant paths or schema restructuring. In this paper an effort is made to face all the relevant issues that can arise in the integration of several conceptual schemata and provide for all of them a design strategy

2.7. DATAID

DATAID-1 [7, 8] is a manual methodology dealing with the design of centralized databases. DATAID-1 is described by Valeria De Antonellis and Antonio Di Leva in 1985 and extends the basis of practical experience of several projects in banks and government offices. They also illustrate a case study of database design using the DATAID approach. The case study refers to a banking environment.

DATAID consists of the following methodological phases: requirement collection and analysis, local views design, local views integration, logical design, physical design. These phases are followed in the development of the case study of database design.

3 UML Meta model

The Unified Modeling Language(UML) [16, 18] is a non-proprietary object modeling and specification language used in software engineering. UML includes a standardized graphical notation that may be used to create an abstract model of a system: the UML model. While UML was designed to specify, visualize, construct, and document software-intensive systems is not restricted to modeling software.

UML has its strengths at higher, more architectural levels and has been used for modeling hardware (engineering systems) and is commonly used for business process modeling, systems engineering modeling, and representing organizational structure.

4 Quality measures and Transformation of conceptual schemes – ANNAPURNA

In the 1980'ties conceptual schemes are recognized and accepted as an important tool for the design and evolution of integrated databases and knowledge-based systems, but the question of quality of conceptual schemes is largely ignored by researchers. In relational database theory quality is defined by the presence or absence of certain normal forms; the definition of quality was very restrictive because a conceptual schema is either good or bad. Quality measures are also ignored, but transformation of conceptual schemes are explored systematically.

Christoph F. Eick represents in his paper [9] the back end of a conceptual schema design methodology, called **ANNAPURNA**. This methodology aims to automate conceptual schema design focusing on the transformation and evaluation of conceptual schemes based on quality measures and transformations that has a theoretical foundation. A general framework for the specification of conceptual schema transformations is proposed and algorithms for evaluation and transformations are provided.

5 ER-SPML our Approach

5.1 Entities and attributes

At this level we consider entities and relationships.

Definition 1(ENTITY)

The set of entities is a finite set of names

$$ENTITY \subseteq N \tag{1}$$

In [5] Chen argue that "An entity is a thing which can be distinctly identified."

Our approach is that each entity e_i belongs to a Entity Set E_i, $e_i \in E_i$, induces *entity type's* $t_{e_i} \in T_{E_i}$. T_{E_i} is a set of domain type names. All entities have a distinct name; in particular, an entity name may not be used again to define another entity with a different type.

$$\forall e_1, e_2 \in T_E : (e_1 : t_{e_1} \rightarrow E_i, e_2 : t_{e_2} \rightarrow E_i) \Rightarrow e_1 = e_2$$

Entities with the same name may, however, appear in different ER Models that are not related by generalization.

Definition 2(ATRIBUTES)

An attribute a_i, could be formally defined as a function which maps from an entity, E_i or a relationship, R_i into a value set, V_i or a cartesian product $\prod_{1 \leq j \leq n} V_{i_j}$ of value sets:

$$a : E_i \; or R_i \;\rightarrow V_i \; or \; V_{i_1} \times V_{i_2} \times V_{i_3} \times \ldots \times V_{i_n} \tag{2}$$

5.2 Relationships

A relationship is an association among entities.

Definition 3(RELATIONSHIP)

A relationship R_i, is a mathematical relation among n entities each taken from an entity set:

$$\{[e_1, e_2, \cdots, e_n] | e_1 \in E_1, e_2 \in E_2, \cdots, e_n \in E_n\} \quad (3)$$

and each tuple of entities, $[e_1, e_2, \cdots, e_n]$, is a relationship. Note that the E_i in the above definition may not be distinct.

5.3 Primary Key and Foreign Key

Definition 4(CANDIDATE KEY)

An attribute or set of attributes that uniquely identifies individual occurrences of an entity type.

$$V(a_1^i, \dots, a_k^i) \neq V(a_1^j, \dots, a_k^j), \; \forall \, i \neq j \quad (4)$$

where $(a_1^i, \dots, a_k^i)$ is candidate key, a subset of entity $e_i = (a_1^i, \dots, a_n^i) \; with \, k \; \leq n$

Definition 5(PRIMARY KEY)

A unique identifier for a row in a table in relational database; A selected candidate key of an entity.

$$V(a_1^i, \dots, a_k^i, \dots, a_n^i) \neq V(a_1^j, \dots, a_k^j, \dots, a_n^j) \quad (5)$$

$\forall \, i \neq j, \, with \, k \; \leq n$

Definition 6(FOREIGN KEY)

An attribute that is a primary key of another relation (table). A foreign key is how relationships are implemented in relational databases.

$$e_i = (a_1^i, \dots, a_k^i, \dots a_n^i) \quad (6)$$

where $(a_1^i, \dots, a_k^i)$ is a primary key, a subset of entity $e_i = (a_1^i, \dots, a_n^i) \; with \; k \; \leq \; n$ and $e_j = (b_1^j, \dots, b_l^j, \dots b_m^j, a_1^i, \dots, a_k^i)$ is the corespondent entity from relationship and $(a_1^i, \dots, a_k^i)$ represents the foreign key of e_j, $\forall \, i \neq j$.

5.4 Stored Procedures

Stored Procedures:(**procedures**, **functions** and **triggers**) are SQL subroutine statements in a RDBMS, for use by all application including the control statements that allow repetition(LOOP) and conditional execution(IF and CASE statements)

A stored procedure is an ordinary program that can be called by an application with an SQL CALL statement. The stored procedure can be called locally or remotely. A remote stored procedure provides the most advantages:

- It reduces traffic across the communication line;
- It splits the application logic and encourages an even distribution of the computational workload;
- It provides an easy way to call a remote program

Triggers are user-written programs that are associated with database tables. You can define a trigger for update, delete, and insert operations. Whenever the operation takes place, regardless of the interface that is changing the data, the trigger program is automatically activated by RDBMS-es and executes its logic. In this way, you can implement complex rules at the database level with total independence from the application environment. We can use triggers for a variety of purposes in your database design. Functions and user-defined programs that enrich the functionality of the RDBMS by adding new functions to the set of built-in functions. Database functions are scalar functions, which means functions that receive some parameters,perform some operations, and return a unique value, such as converting farenhait to celsius degrees or calculating the net present value given the final amount, monthly payment, number of payments and interest rate. Functions return a table for a given set of parameters instead of a single scalar value, such as the top k performing salesperson or the projected currency exchange rates between an initial and final date for a given pair of currencies.

5.5. From Entity Relationship Model to Relational Model

Entity Relationship Model(conceptual level) could be translate in Relational Model(physical level) in a natural mod using the following transformation [13]:

- any entity became a table;
- any attribute from entity became a field in the repectiv table;
- any relationship became a special table or an primary key in one of two tables and referencing in the corespondent table.

6 Case study

Let us look at a typical example where stored procedure can be effective with ER-SPML. A company named ACME Inc runs its business on a server located at the headquarters and on the client systems in one named branch where Human Resource(HR) Department is located is running the local application. An payroll clerk is working with HR application on client side, which has to update minimally two tables on the server side:

- the employee table is named EMPLOYEE
- the department table is named DEPARTMENT

We consider that the relationship between DEPARTMENT and EMPLOYEE is 1:m ; in one department are working many employees. A company policy raised the employees salary after five years with 10%; this will be on the server side.

A standard syntax for procedure in RDBMS flavors [3, 11, 12, 14, 19, 21, 22] is:

```
CREATE

 [PROCEDURE]|
 [FUNCTION] procedure_name[list_of_param]

[RETURN]|[RETURNS]

[IS]|[IN]|[AS]

 [local_declaration]

[OS :Windows | NetWare | UNIX]

BEGIN

 [procedure_body]

[EXCEPTION]|[ON EXCEPTION RESUME]

 [exception_body]

END procedure_name;
```

In Figure 1 we present the ER Diagram from P.P. Chen point-of-view, conceptual level with rectangle for entity name, oval for attributes and lines connected to the entity, for the Human Resource(HR); this diagram is easy to transform in Figure 3 at the physical level for HR Database Diagram.

In Figure 2 we extend the ER Diagram with ER SP appending a new diagram shape for $salary\_raised()$ operation for the conceptual level; this diagram is easy to transform in Figure 4 at the physical level in HR Database Diagram with procedure.

The ER-SPML is represent with the following code:

```
context salary_raised(Emp_ID : NUMBER,
                      raise : NUMBER)
Emp_ID NUMBER,raise NUMBER;
if(YEAR(SYSDATE()-YEAR(HireDate)) >5
raise=salary+salary*0.1
end if;
```

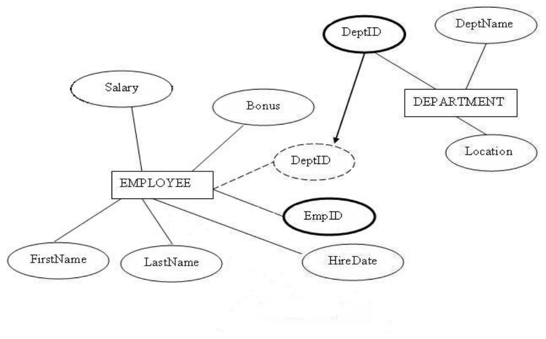

Figure 1. ER Diagram for HR without ER SP

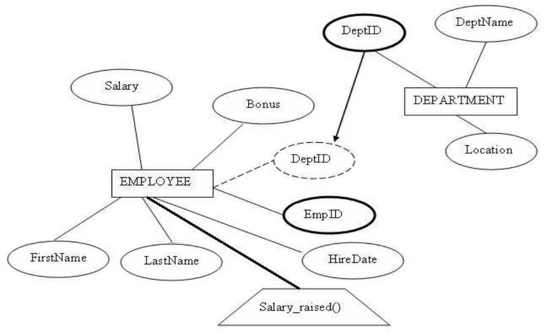

Figure 2. ER Diagram for HR with ER SP

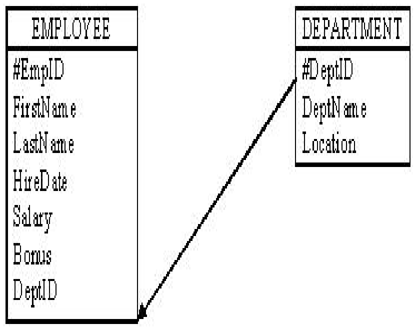

Figure 3. HR Database diagram without procedure

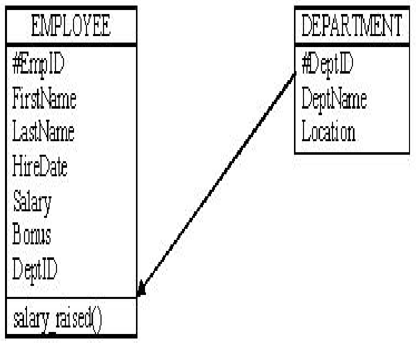

Figure 4. HR Database diagram with procedure

7. Conclusion

ER-SPML proposed a operation called Stored Procedures for the diagrammatical representation(trapezoidal) and the language behind him in idea to be platform independent in relation with SQL dialects and their Procedural Languages. The concept of stored procedures operation is abstract, helps in syntactic and semantic modeling, and is required in physical implementation. A stored procedures operation along with his language captures the syntactic and semantics of an RDBMS schema in his dynamical evolution, not in statical way like ER model.

Acknowledgments: The author would like to acknowledge the financial support by "Petru Maior" University of Târgu-Mureş, through the Grant no. 1783/2006).

References

[1] R. Barker. *CASE*METHOD Entity Relationship Modeling*. Workingham, England, 1990.

[2] C. Batini and M. Lenzerini. A methodology for data schema integration in the entity relationship model. *ER*, pages 413–420, 1983.

[3] H. Bedoya, F. Cruz, D. Lema, and S. Singkorapoom. Stored procedures, triggers, and user-defined functions on db2 universal database. *IBM: International Technical Support Organization*, pages 81–92, October 2006.

[4] B. Breutmann, R. Mauer, and E. Falkenberg. *CSL: A language for defining conceptual schema*. Elsevier, North Holland 1, 1979.

[5] P. P. Chen. The entity-relationship model: Toward a unified view of data. *ACM Trans. Database Systems, ACM Press, New York, NY, USA.*, pages 9–36, 1976.

[6] P. P. Chen. A PRELIMINARY FRAMEWORK FOR ENTITY-RELATIONSHIP MODELS. *Entity-Relationship Approach to Information Modeling and Analysis, Elsevier Science Publishers, North-Holland*, pages 9–36, 1983.

[7] V. De Antonellis and A. Di Leva. A case study of database design using the dataid approach. *Information Systems*, 10:339–359, 1985.

[8] V. De Antonellis and A. Di Leva. Dataid-1: Adatabase design methodology. *Information Systems*, 10(2):181–195, 1985.

[9] C. F. Eick. A methodology for the design and transformation of conceptual schemes. *Proceedings of the 17th International Conference on Very Large Data Bases, Barcelona*, September 1991.

[10] D. W. Embley and T. W. Ling. Synergistic database design with an extended entity-relationship model. *Proceedings of the Eight International Conference on Entity-Relationship Approach to Database Design and Querying, North-Holland*, pages 111–128, 1990.

[11] iAnywhare a Sybase ©Company. Adaptive Server ©. *AnywhereSQL Reference, Part number: DC38129-01-0902-0*, pages 362–367, October 2004.

[12] iAnywhare a Sybase ©Company. Adaptive Server ©. *Fundamentals, Part number: B035-1141-115A*, pages 48–54, May 2006.

[13] F. E. Ipate and M. Popescu. *Dezvoltarea aplicaţiilor de baze de date în Oracle8 şi Forms6*. BIC ALL, Bucureşti, 2000.

[14] MySQL 5.1 Reference Manual. Document generated on: 2006-10-29. *revision: 3792, Copyright MySQL AB*, pages 1169–1174, October 1997-2006.

[15] Object Management Group. Object Constraint Language. *(OCL 2.0 Specification), Version 2.0*, pages 21–30, June 2005.

[16] Object Management Group. Unified Modeling Language. *(UML) specification: Superstructure, revised final adopted specification, Version 2.0*, pages 1–16, August 2005.

[17] Object Management Group. Object Constraint Language. *(OMG Available Specification), Version 2.0*, pages 5–10, May 2006.

[18] Object Management Group. Unified Modeling Language. *(UML) Infrastructure, Version 2.0*, pages 10–15, March 2006.

[19] PostgreSQL 7.2 Programmers Guide. The postgresql global development group. *Copyright © 1996-2001 by The PostgreSQL Global Development Group*, pages 329–334, 1996-2001.

[20] C. Rolland. An information system methodology supported by an expert design tool. *Elsevier Science Publishers*, 1998.

[21] Teradata ©a division of NCR ™. SQL Reference . *Stored Procedures and Embedded SQL, Release V2R6.1, B035-1148-115A*, pages 88–110, November 2005.

[22] Teradata ©a division of NCR ™. SQL Reference . *Fundamentals, Release V2R6.1, B035-1148-115A*, pages 68–74, May 2006.

[23] B. Thalheim. HERM: Putting theory into practice. *IFIP-Workshop on Database Intellectualization, Control Systems and Machines, IDS-92*, pages 85–93, May 1992.

[24] B. Thalheim. The Strength of ER Modeling. *Proceedings – 20 Years of ER Modeling, LNCS 1565*, pages 227–242, 1998.