

Sobre a Manutenção da Consistência de Representações Relacionais Otimizadas de Esquemas ER

Altigran S. da Silva^{1*} Alberto H.F. Laender¹ Marco A. Casanova²

¹Departamento de Ciência da Computação
Universidade Federal de Minas Gerais
Caixa Postal 702
30161-970 Belo Horizonte MG, Brasil
laender@dcc.ufmg.br

²Centro Científico Rio
IBM Brasil
Caixa Postal 4624
20071-001 Rio de Janeiro RJ, Brasil
casanova@vnet.ibm.com

Abstract

This paper discusses the problem of maintaining the semantic consistency of optimized relational representations of entity-relationship schemas produced by collapsing entity and relationship schemes according to common database design heuristics. An intra-relation type of integrity constraint, called null dependency, is proposed which enforces the correction of such relational representations according to the semantics of the entity-relationship model.

1 Introdução

No desenvolvimento de sistemas de informação, uma das tarefas cruciais é o projeto do banco de dados associado, o qual, em grande parte das aplicações atuais, utiliza a tecnologia relacional [11]. A estimativa é que essa tecnologia permaneça servindo de base para os sistemas gerenciadores de bancos de dados comerciais ainda por algum tempo, o que implica na necessidade de investimento em estudos de problemas ainda não totalmente resolvidos no que diz respeito ao projeto, implementação e manutenção de bancos de dados relacionais.

Os sistemas de gerência de bancos de dados relacionais, sendo hoje os mais utilizados, provêm mecanismos relativamente simples para suportar a representação lógica de objetos do mundo real. No entanto, a simplicidade desses mecanismos implica em dificuldades quando se tenta gerar esquemas a partir de abstrações criadas pela observação do mundo real, levando assim a falhas de representação e exigindo passos adicionais de verificação e refinamento no processo de modelagem.

Na tentativa de amenizar este problema, têm sido comumente usados modelos de dados de maior poder expressivo, chamados genericamente de modelos de dados conceituais ou semânticos [13, 18], os quais possuem recursos de representação capazes de capturar com mais precisão a riqueza semântica das descrições obtidas diretamente a partir da observação do mundo real. O processo de projeto de banco de dados baseado nesses modelos consiste em inicialmente gerar um esquema conceitual (fase de projeto conceitual) e, em seguida, mapear, ou seja converter de forma adequada, este esquema para um esquema relacional implementável em um SGBD relacional (fase de projeto lógico) [4, 12, 21].

Em particular, o modelo de Entidades e Relacionamentos (modelo ER) [9] e suas várias extensões [4, 6, 12, 15, 22], têm sido largamente utilizados para a construção de esquemas conceituais e são, por isso, comumente usados como base para métodos de projeto de bancos de dados. Especificamente para o projeto lógico, que chamaremos simplesmente de **projeto**, vários foram os métodos criados no intuito de realizar com mais eficiência e precisão o mapeamento do esquema conceitual ER para o esquema relacional de forma que sejam reduzidas ao máximo as perdas semânticas ocorridas no processo [2, 3, 4, 10, 14, 19, 22]. Esses métodos geralmente se baseiam em algum critério para definir as características do esquema gerado.

Quando se usa esquemas relacionais para implementar esquemas descritos segundo o modelo ER, uma maneira natural e direta de fazê-lo é utilizar um esquema de relação para implementar cada esquema de entidade ou relacionamento e garantir as restrições semânticas através de dependências de inclusão (DPI) [5]. Este tipo de mapeamento, chamado de representação relacional *um-para-um* [7, 8] é ineficiente no sentido que o número de DPIs a serem verificadas é potencialmente grande. Desta forma, visando reduzir o número de DPIs envolvidas, pode-se optar por representar em um mesmo esquema de relação vários esquemas de entidade ou relacionamento que compartilham alguma propriedade. Esta estratégia de otimização é chamada de **colapsamento**.

Em [7, 8] é proposto um método de projeto onde o principal critério adotado para a otimização do esquema relacional gerado é a redução do número de dependências de inclusão entre os esquemas de relação definidos. Para essa otimização são aplicadas duas heurísticas bastante conhecidas como critério para colapsamento de esquemas:

- Um esquema de relacionamento F pode ser colapsado em um esquema de entidade E se F é funcional em E (por exemplo, um relacionamento binário $n:1$);
- Um esquema de entidade F pode ser colapsado em um esquema de entidade E se F especializa E .

Nestes casos, E e F podem ser representados no mesmo esquema de relação.

Quando esquemas de entidade ou relacionamento são colapsados em um mesmo esquema de relação, a representação de suas instâncias fica dependente da ocorrência de instâncias da entidade que é a base do colapsamento. Isso cria restrições quanto à possibilidade dos atributos discriminadores das entidades ou relacionamentos assumirem valores nulos na relação que implementa a representação. Essas restrições se propagam ao longo das hierarquias do esquema, criando a necessidade de que se adote algum mecanismo de restrição de integridade para evitar o aparecimento de estados no banco de dados inconsistentes com a semântica do esquema ER mapeado.

Este trabalho discute o problema da manutenção da consistência semântica de representações relacionais de esquemas ER obtidas a partir do método descrito em [7, 8]. Propõe-se que esta consistência seja mantida através do uso de um tipo de restrição de integridade chamada *Dependência de Nulos (DN)* que é semelhante às restrições usadas no método de projeto descrito em [17] e na proposta de extensão do modelo relacional apresentada em [16].

O trabalho a seguir está organizado da seguinte forma. A Seção 2 apresenta a notação e a terminologia que serão utilizadas ao longo do trabalho. A Seção 3 apresenta um exemplo onde podem ocorrer as inconsistências acima descritas. As Seções 4 e 5 propõem um mecanismo de restrição de integridade adequado para garantir que não ocorram tais inconsistências e a Seção 6 aplica esta solução ao esquema anteriormente apresentado. Finalmente, a Seção 7 apresenta algumas conclusões e discute os resultados obtidos.

2 Notação e Terminologia

2.1 Esquemas Entidade-Relacionamento

Neste trabalho são utilizados os conceitos de esquema (schema) ER, esquemas (schemes) de entidade e de relacionamento, declaração de especialização e grafo de um esquema ER definidos em [7, 8]. Também são mantidas as mesmas restrições semânticas associadas aos relacionamentos e especializações.

Um esquema de entidade possui um nome, um conjunto de atributos com seus respectivos domínios e, opcionalmente, uma chave primária e uma ou mais chaves alternativas. As chaves são compostas por subconjuntos de atributos. Pelo menos um dos atributos de um esquema de entidade deve ser um *discriminador*, ou seja, deve sempre possuir valor diferente de nulo, o que serve para indicar a ocorrência de uma entidade (instância). Os atributos da chave primária são sempre discriminadores.

Um esquema de entidade pode ser *especializado* por um ou mais esquemas de entidade e também pode especializar um ou mais esquemas de entidade, sendo que se um esquema de entidade F especializa um esquema de entidade E, então toda instância de F é também uma instância de E.

A chave primária de um esquema de entidade, caso não seja declarada na sua definição é tomada como sendo formada por todos os seus atributos ou então é herdada de um dos esquemas por ele especializados caso seja uma especialização.

Um esquema de relacionamento possui um nome e pode opcionalmente possuir uma lista de atributos. Suas instâncias são elementos do produto cartesiano dos conjuntos de instâncias de esquemas de entidades ditos *participantes* do relacionamento. A cada participante é atribuído um papel que deve ser único para cada relacionamento, mesmo que o esquema de entidade apareça como participante do esquema de relacionamento mais de uma vez. Se R é um esquema de relacionamento e E é um de seus participantes com papel N, dizemos que R é *funcional* em E se cada instância de E pode participar de apenas uma instância de R. Dizemos que R é *total* em E se todas as instâncias de E têm participação obrigatória nas instâncias de R.

Definimos um grafo de um esquema ER como sendo um multigrafo rotulado dirigido $g = (V, A, l)$, tal que V é o conjunto de nomes de todos os esquemas de entidade e relacionamento do esquema ER e A é um conjunto de arcos parcialmente rotulados pelos rótulos em l . Um arco (O, P) pertence a A se e somente se: *i*) O é uma especialização de P no esquema ER, neste caso $l((O, P))$ não está definido *ii*) O é o nome de um esquema de relacionamento no esquema ER tal que P participa com papel N , neste caso $l((O, P)) = N$ ou *iii*) P é um esquema de relacionamento no esquema ER tal que O participa com papel N e P é total em N , neste caso $l((O, P)) = N$.

Como forma de simplificar a discussão, serão utilizadas as funções abaixo definidas. Sejam S e R esquemas de entidade ou relacionamento de um esquema ER. Assim, temos:

- $A(S)$: É o conjunto de atributos de S devidamente qualificados. Os elementos de $A(S)$ são triplas da forma $\langle Q, N, D \rangle$ onde Q é um qualificador do atributo, N o seu nome e D o seu domínio conforme definido no esquema ER. Um atributo A pertence a $A(S)$ se, e somente se, um dos casos abaixo ocorre:

– A é declarado como sendo um atributo de S, neste caso $Q=S$ e $N=A$;

- A pertence à chave primária de um esquema de entidade especializado pelo esquema de entidade S através do caminho $S, S_k, S_{k-1}, \dots, S_1, S_0$ ($k \geq 0$) do grafo do esquema ER. Neste caso $Q=S.S_k.S_{k-1} \dots S_1.S_0$ e $N=A$ em $A(S)$;
- A pertence à chave primária de um esquema de entidade participante do esquema de relacionamento S. Seja S_k esta entidade, A ocorrerá com $Q=S.S_k.S_{k-1} \dots S_1.S_0$ e $N=A$ em $A(S)$ ($k \geq 0$), onde $S_k, S_{k-1}, \dots, S_1, S_0$ ($k \geq 0$) é um caminho no grafo do esquema ER;
- $K(S)$: Se S é um esquema de entidade, $K(S)$ é o conjunto de atributos que compõem a chave primária de S. Se S é um esquema de relacionamento, $K(S)$ é o conjunto dos atributos que compõem a chave primária do seu identificador primário, se houver algum identificador, ou senão, a união dos conjuntos de atributos que compõem as chaves primárias de todos os seus participantes. Os elementos de $K(S)$ são triplas $\langle Q, N, D \rangle$ definidos conforme especificado acima;
- $U(S)$: Se S é um esquema de entidade, $U(S)$ é o conjunto de chaves alternativas de S. Se S é um esquema de relacionamento, $U(S)$ é união das chaves primárias dos seus identificadores secundários, caso haja algum. Os elementos de $U(S)$ são triplas $\langle Q, N, D \rangle$ definidos conforme especificado acima;
- $N(S)$: Conjunto dos atributos discriminadores do esquema de entidade ou relacionamento S. Os atributos discriminadores são aqueles que determinam a existência de uma instância da entidade ou relacionamento. Neste trabalho iremos considerar $N(S)$ como sendo a união de $K(S)$ com o subconjunto de $A(S)$ cujos elementos não aceitam valores nulos conforme definição no esquema ER. Supõe-se que este subconjunto nunca é vazio. Se S é um esquema de relacionamentos, todos os atributos das chaves primárias dos esquemas de entidades participantes fazem parte de $N(S)$.
- $\varphi(S, R)$ é o papel de S no esquema de relacionamento R.

Os elementos de $A(S)$, $K(S)$, $U(S)$ e $N(S)$ serão referenciados a seguir na forma $Q.N$ por questão de simplicidade.

2.2 Esquemas Relacionais

Também para esquemas relacionais, serão usados os conceitos definidos em [7, 8]. Assim, um esquema de relação possui um nome e, opcionalmente, uma lista de atributos (cada um com seu respectivo domínio), uma chave primária e uma ou mais chaves alternativas. Pode-se especificar que um certo domínio não admite valores nulos. Como restrição semântica entre as relações de um esquema relacional serão adotadas as *dependências de inclusão* [5]. Será permitido o uso de qualificadores nas referências de atributos de esquemas de relação.

Para nos referirmos a uma instância de uma relação R usaremos $r(R)$.

2.3 Florestas de Colapsamento

Seja $g = (V, A, l)$ um grafo de um esquema ER. Um arco (F, E) em A é *colapsável* sse F é o nome de um esquema que especializa o esquema E ou F é o nome de um esquema de relacionamento que é funcional em um papel N e $l((F, E)) = N$. Um nodo F é colapsável

sse existe um arco colapsável partindo de F em g . Uma floresta de colapsamento é definida, conforme [7, 8], como sendo um sub-grafo do grafo de colapsamento g de um esquema ER tal que os seus vértices são os mesmos do grafo de colapsamento g e um vértice F é filho de E sse o arco (F, E) em g é colapsável. Uma floresta é completa sse nenhuma de suas raízes é um nodo colapsável. Uma floresta de colapsamento de um esquema ER indica uma possível forma de colapsamento para os seus esquemas de entidade e relacionamento.

Utilizaremos a seguinte notação. Sejam f uma floresta de colapsamento e S um esquema de entidade ou relacionamento. Temos que :

- n_S é o nodo de f que representa S ;
- a_S^* é a árvore de f que contém n_S ;
- a_S é a sub-árvore de a_S^* cuja raiz é n_S ;
- $\delta(n_S)$ é o conjunto de descendentes de n_S ;
- $\gamma(n_S)$ é o conjunto de filhos de n_S ;
- $\varepsilon(n_S) = S$.

2.4 Representações Relacionais

As representações relacionais aqui utilizadas são produzidas de acordo com o método de projeto descrito em [7, 8]. Os conjuntos de atributos oriundos de esquemas de entidade ou relacionamento serão fornecidos pelas funções $A(S)$, $K(S)$, $U(S)$ e $N(S)$. Portanto, será considerado que estas funções, quando usadas no contexto de esquemas relacionais, retornam atributos de esquemas ER convertidos para atributos de (esquemas de) relações.

Outra hipótese importante assumida, garante que se $n_R \in \gamma(n_S)$ em uma árvore de colapsamento, as instâncias de S e R correspondentes serão representadas sempre na mesma tupla da representação relacional. Será usado o termo *representação relacional ER-dirigida* para caracterizar essas relações.

3 Exemplo de Mapeamento Otimizado

Exemplo 1 Sejam os esquemas de entidade E e F e o esquema de relacionamento R , como definidos abaixo.

- | | |
|--|--|
| <ul style="list-style-type: none"> • define entity E
 attributes K_E integer not null
 A_E integer
 key K_E | <ul style="list-style-type: none"> • define entity F
 attributes K_F integer not null
 A_F integer
 key K_F |
| <ul style="list-style-type: none"> • define relationship R over E, F total
 identifier F | |
| <ul style="list-style-type: none"> • specialize E into F | |

A representação deste esquema ER é o esquema de relação :

$$E^*(E.K_E, E.A_E, F.K_F, F.E.K_E, F.A_F, R.E.K_E, R.F.K_F, R.A_R)$$

construído a partir da árvore de colapsamento da Figura 1(c).

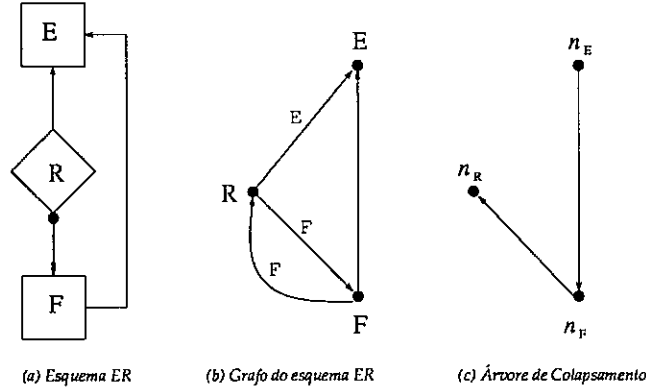


Figura 1: Componentes do Exemplo 1

No esquema de relação E^* , $(E.K_E, E.A_E)$ é o sub-conjunto de atributos que representa instâncias de E, $(F.K_F, F.E.K_E, F.A_F)$ é o sub-conjunto de atributos que representa instâncias de F e $(R.E.K_E, R.F.K_F, R.A_R)$ é o sub-conjunto de atributos que representa instâncias de R. Assim, podemos observar, para toda tupla t de $r(E^*)$, o seguinte:

- Usando a noção de atributos discriminadores, toda tupla t de $r(E^*)$ que representa uma instância de E deve satisfazer $t(E.K_E) \neq \lambda$. O mesmo vale para tuplas representando instâncias de F que devem satisfazer $t(F.K_F) \neq \lambda \wedge t(F.E.K_E) \neq \lambda$ e para tuplas representando instâncias de R que devem satisfazer $t(R.E.K_E) \neq \lambda \wedge t(R.F.K_F) \neq \lambda$.
- Para que o mapeamento esteja consistente, é necessário garantir que

$$(t(F.K_F) \neq \lambda \wedge t(F.E.K_E) \neq \lambda) \vee (t(F.K_F) = \lambda \wedge t(F.E.K_E) = \lambda)$$

onde a primeira conjunção é verdadeira apenas se há uma instância de F representada, o contrário ocorrendo para a segunda conjunção. O mesmo pode ser dito para $R.E.K_E$ e $R.F.K_F$ com respeito a uma instância de R.

- O atributo $E.A_E$ só pode possuir valor não nulo em tuplas de E^* que representem instâncias de E. O mesmo vale para $F.A_F$ e para $R.A_R$ no que diz respeito a instâncias de F e R respectivamente.
- Se supusermos que é usada uma representação relacional ER-dirigida, ou seja, toda instância de F é representada na mesma tupla de E^* que a instância de R da qual participa (note que F é identificador de R), então só pode haver uma instância de R representada na tupla t de $r(E^*)$ se houver uma instância de F representada, ou seja, se $t(F.K_F) \neq \lambda \wedge t(F.E.K_E) \neq \lambda$.
- R é um relacionamento total em F, portanto toda instância de F deve estar associada a alguma instância de R. Assim temos que não pode haver representação de uma instância de F em uma tupla onde não ocorra também uma instância de R.

Podemos, portanto, escrever as seguintes sentenças que expressam as restrições existentes para toda tupla t da relação quanto a atributos de E^* assumirem valores nulos :

$$F1. \neg t(E.K_E = \lambda)$$

$$F2. \neg t(E.K_E = \lambda) \vee t(E.A_E = \lambda)$$

$$F3. \neg t(E.K_E = \lambda) \vee (t(F.K_F = \lambda) \wedge t(F.E.K_E = \lambda))$$

$$F4. (\neg t(F.K_F = \lambda) \wedge \neg t(F.E.K_E = \lambda)) \vee t(F.A_F = \lambda)$$

$$F5. (\neg t(F.K_F = \lambda) \wedge \neg t(F.E.K_E = \lambda)) \vee (t(R.E.K_E = \lambda) \wedge t(R.F.K_F = \lambda))$$

$$F6. (\neg t(R.E.K_E = \lambda) \wedge \neg t(R.F.K_F = \lambda)) \vee t(R.A_R = \lambda)$$

$$F7. (\neg t(R.E.K_E = \lambda) \wedge \neg t(R.F.K_F = \lambda)) \vee (t(F.K_F = \lambda) \wedge t(F.E.K_E = \lambda))$$

$$F8. (\neg t(F.K_F = \lambda) \wedge \neg t(F.E.K_E = \lambda)) \vee (t(F.K_F = \lambda) \wedge t(F.E.K_E = \lambda))$$

$$F9. (\neg t(R.E.K_E = \lambda) \wedge \neg t(R.F.K_F = \lambda)) \vee (t(R.E.K_E = \lambda) \wedge t(R.F.K_F = \lambda))$$

4 Dependências de Nulos

Definição 1 *Sejam $A = \{A_1, A_2, \dots, A_n\}$ e $B = \{B_1, B_2, \dots, B_m\}$ dois subconjuntos não necessariamente disjuntos do conjunto de atributos de um esquema de relação R . A notação*

$$R : A_1, A_2, \dots, A_n \rightsquigarrow B_1, B_2, \dots, B_m$$

*será usada para indicar que os atributos do conjunto B só podem assumir valores nulos se os atributos do conjunto A tiverem valor nulo. Dizemos que existe uma **dependência de nulos** de B para A .*

As LDD de diversos modelos de dados permitem descrever que um certo atributo jamais poderá assumir o valor nulo. No contexto de dependências de nulos isto pode ser expresso escrevendo $A_k \rightsquigarrow \square$ para um certo atributo A_k .

Definição 2 *Seja $A = \{A_1, A_2, \dots, A_n\}$ um subconjunto de atributos de um esquema de relação R . A notação*

$$R : [A_1, A_2, \dots, A_n]$$

*será usada para indicar que os atributos do conjunto A são ditos **dependentes de nulos entre si (DNE)** ou seja que os atributos do conjunto A devem ter todos valor nulo ou todos valor não nulo.*

As duas notações definidas acima serão usadas para expressar restrições de integridade que serão genericamente denominadas de **dependências de nulos (DN)**.

4.1 Verificação de Dependências de Nulos

Uma vez definido um conjunto de dependências de nulos sobre um esquema relação, é necessário garantir que estas dependências sejam respeitadas em qualquer estado consistente do banco de dados. O conceito de verificação de um conjunto de dependências de nulos fornece a noção de quando uma dependência de nulos foi violada.

Daqui em diante usaremos A no lugar de $t(A)$, para nos referirmos ao valor do atributo A em toda tupla t de uma relação, quando fizermos a comparação deste valor com λ .

Definição 3 *Seja uma dependência de nulos*

$$N = R : A_1, \dots, A_n \rightsquigarrow B_1, \dots, B_m \quad (1 \leq n, m)$$

A fórmula

$$(\neg(A_1 = \lambda) \wedge \dots \wedge \neg(A_n = \lambda)) \vee ((B_1 = \lambda) \wedge \dots \wedge (B_m = \lambda)),$$

chamada Predicado de Verificação de N , é expressa pela função $\mathcal{F}(N)$, admitindo-se que $\neg(\square = \lambda)$ é uma tautologia.

Definição 4 *Seja uma dependência de nulos*

$$N = R : [A_1, \dots, A_n] \quad (1 \leq n)$$

A fórmula

$$((A_1 = \lambda) \wedge \dots \wedge (A_n = \lambda)) \vee (\neg(A_1 = \lambda) \wedge \dots \wedge \neg(A_n = \lambda)),$$

chamada Predicado de Verificação de N , é expressa pela função $\mathcal{F}(N)$.

Definição 5 *Seja D um conjunto de dependências de nulos $D = \{N_1, N_2, \dots, N_k\}$ sobre o esquema de relação R . Dizemos que D se verifica para um certo estado de $r(R)$ se, e somente se,*

$$\mathcal{F}(D) = \mathcal{F}(N_1) \wedge \mathcal{F}(N_2) \wedge \dots \wedge \mathcal{F}(N_k)$$

se verifica para uma função de interpretação cujo domínio é o conjunto de fórmulas atômicas $A = \lambda$, onde A é um atributo de R . Qualquer estado não transitório onde $\mathcal{F}(D)$ não se verifique é dito ser um estado inconsistente sob o ponto de vista do conjunto D . $\mathcal{F}(D)$ será chamado de Predicado de Verificação de D .

O uso de predicados de verificação é importante pois possibilita a implementação de dependências de nulos em SGBDs que possuam algum sistema de regras ou asserções.

Definição 6 *Dois conjuntos de DNs D_1 e D_2 são equivalentes se, e somente se, $\mathcal{F}(D_1) \equiv \mathcal{F}(D_2)$.*

4.2 Geração do Conjunto de Dependências de Nulos

Seja n_S uma raiz em uma sub-árvore de uma floresta de colapsamento, sendo S o esquema de entidade ou relacionamento representado por n_S . \mathcal{C}_S é o conjunto de dependências de nulos impostas por n_S e deve incluir:

- Dependências indicando que os atributos discriminadores de um esquema de entidade ou relacionamento são DNE na representação;
- Dependências da forma $A \rightsquigarrow \square$, sendo A o conjunto dos atributos discriminadores do esquema representado pela raiz da árvore de colapsamento na qual a relação é baseada;
- Dependências garantindo que a atribuição de nulos aos atributos não discriminadores depende dos discriminadores;
- Dependências garantindo que a atribuição de nulos aos atributos discriminadores depende dos discriminadores no nó “pai” de um certo nó, exceto para a raiz;
- Dependências garantindo que a atribuição de nulos aos atributos discriminadores em um esquema de entidade depende da nulificação dos atributos discriminadores em um esquema de relacionamento do qual ele é participante total e cujo nó seja “filho” do nó que o representa.

Desta forma, o conjunto de dependências de nulos \mathcal{C}_S deve ser constituído como segue:

$$\mathcal{C}_S : \bigcup_{R \in \gamma(S)} \mathcal{C}_R \cup \{N(S) \rightsquigarrow \square \mid n_S \text{ é uma raiz em } f\} \\ \cup \{N(S)\} \\ \cup \{N(S) \rightsquigarrow (A(S) - N(S))\} \\ \cup \{N(S) \rightsquigarrow N(R) \mid n_R \in \gamma(n_S)\} \\ \cup \{N(\varepsilon(p)) \rightsquigarrow N(S) \mid p \in \gamma(n_S), (S, \varepsilon(p)) \in A_g \text{ e } l_g(S, \varepsilon(p)) = \varphi(S, \varepsilon(p))\}$$

onde $g = (V_g, A_g, l_g)$ é o grafo do esquema ER em questão.

Quando n_S é uma raiz da floresta de colapsamento, a relação S^* , que serve de base para a representação ER-dirigida das entidades e relacionamentos representados na árvore cuja raiz é n_S , tem como conjunto de dependências de nulos o próprio \mathcal{C}_S .

5 Grafo de Dependências de Nulos

As dependências de nulos referentes aos atributos de uma representação relacional derivada de uma árvore de colapsamento podem ser descritas através de um grafo que chamaremos de **grafo de dependências nulos (GDN)**. A construção desse grafo pode ser feita inicialmente com base no grafo do esquema ER e na árvore de colapsamento. É possível também extrair o conjunto de dependências de nulos a partir de um GDN.

Definição 7 *Sejam R uma representação relacional ER-dirigida, $h = (N_h \cup M_h \cup \{\Lambda\}, A_h)$ um grafo dirigido, sendo N_h e M_h conjuntos disjuntos de vértices e A_h um conjunto de arcos, e α a função sobre $N_h \cup M_h$ que retorna uma lista de atributos formada pelos elementos de um subconjunto do conjunto de atributos de R . O grafo h é definido como um **grafo de dependências de nulos (GDN)** para R se, e somente se, as seguintes condições se verificam:*

- 1) Se $V \in N_h$, temos que os elementos de $\alpha(V)$ são DNE;
- 2) Para qualquer arco $(Q, P) \in A_h$, temos $\alpha(Q) \rightsquigarrow \alpha(P)$;
- 3) Se $(\Lambda, Q) \in A_h$ então $\alpha(Q) \rightsquigarrow \square$.

5.1 Construção do Grafo de Dependências de Nulos

A construção de um GDN pode ser feita inicialmente partindo-se de uma floresta de colapsamento e usando-se o grafo do esquema ER do qual ela foi retirada.

Seja f uma floresta de colapsamento e $g = (A_g, V_g, l_g)$ o grafo do esquema ER do qual f foi retirada. Um GDN $h = (N_h \cup M_h \cup \{\Lambda\}, A_h)$ pode ser construído para a representação relacional derivada de f como segue:

- $N_h = \{N_S \mid \alpha(N_S) = N(S), \text{ para todo } S \text{ tal que } n_S \in f\}$
- $M_h = \{M_S \mid \alpha(M_S) = A(S) - N(S), \text{ para todo } S \text{ tal que } n_S \in f\}$
- $A_h = \{(N_S, M_S) \mid n_S \in f\} \cup \{(\Lambda, N_S) \mid n_S \text{ é uma raiz em } f\} \cup \{(N_S, N_R) \mid (R, S) \in A_g \text{ e } \varphi(R, S), \varphi(S, R) \notin l_g\} \cup \{(N_P, N_S) \mid n_P \in \gamma(n_S) \text{ em } f, (S, P) \in A_g \text{ e } l_g(S, P) = \varphi(S, P)\}$

onde P, R e S são esquemas de entidade ou relacionamento no esquema ER em questão.

Definição 8 *Ao grafo de dependência de nulos construído como descrito acima chamaremos grafo de dependências de nulos natural.*

Desta forma, o GDN natural de um esquema ER contém :

- Dois vértices, N_S e M_S para cada nodo n_S da floresta de colapsamento e um arco (N_S, M_S) entre estes vértices, indicando que a atribuição de nulos aos atributos não discriminadores ($\alpha(M_S)$) depende dos discriminadores ($\alpha(N_S)$);
- Um arco (Λ, N_S) onde N_S é o vértice correspondente à raiz n_S da floresta de colapsamento;
- Um arco (N_R, N_S) para cada arco entre S e R não rotulado existente no grafo g do esquema, indicando que, se S especializa R, a atribuição de nulos aos atributos discriminadores de S ($\alpha(N_S)$) depende da atribuição de nulos aos atributos discriminadores de R ($\alpha(N_R)$);
- Um arco (N_P, N_S) para cada arco (S,P) rotulado existente em g tal que n_P e n_S são adjacentes em f , indicando que a atribuição de nulos aos atributos discriminadores de um esquema de relacionamento depende da atribuição de nulos dos atributos discriminadores do esquema de entidade com o qual este relacionamento for colapsado segundo f . Se o esquema de entidade for participante total deste relacionamento haverá um arco também no sentido contrário.

Além disso, para todos os atributos associados pela função α aos elementos de N_h teremos entre eles dependências de nulos, ou seja, eles formam conjuntos DNE.

Deve-se notar que a atribuição de nulos em hierarquias de especialização para uma dada floresta de colapsamento independe de como são formadas as árvores, uma vez que só há uma instância envolvida (a do esquema de entidade representado na raiz da árvore).

5.2 Grafo de Dependências Nulos e Conjunto de Dependências de Nulos

Uma vez construído um GDN h , é simples gerar um conjunto de dependências de nulos que garantirá à representação relacional a correção semântica. A geração pode ser feita da seguinte forma:

- Gere uma dependência $\alpha(Q) \rightsquigarrow \square$, para o arco $(\Lambda, Q) \in A_h$.
- Gere uma dependência $\alpha(P) \rightsquigarrow \alpha(Q)$, para cada arco $(P, Q) \in A_h$.
- Gere uma dependência $[\alpha(P)]$ para cada vértice $P \in N_h$, somente se $|\alpha(P)| > 1$.

Definição 9 A função $\mathcal{G}(h)$ gera um conjunto de dependências de nulos a partir do grafo dependências de nulos h .

O conjunto de DNs gerado a partir de um GDN construído conforme descrito acima é igual ao conjunto gerado pelo procedimento descrito na Seção 4.2.

Definição 10 Dois grafos de dependências de nulos h_1 e h_2 são equivalentes se, e somente se, $\mathcal{G}(h_1) \equiv \mathcal{G}(h_2)$.

5.3 Simplificações no Grafo de Dependências de Nulos

Os conjuntos de dependências de nulos gerados com base nos métodos descritos nas Seções 4.2 e 5.1 podem ser simplificados de tal forma que a sua implementação diminua o número de verificações de atributos, evitando também verificações redundantes. Para realizar essas simplificações, são feitas transformações no GDN natural produzido conforme descrito na Seção 5.1, sendo o conjunto de dependências de nulos gerado a partir do grafo resultante conforme a Seção 5.2. Os Teoremas apresentados a seguir garantem que as transformações efetuadas produzem GDNs equivalentes. Suas demonstrações podem ser verificadas em [20].

5.3.1 Eliminação de Redundâncias nos Conjuntos α

Teorema 1 Sejam R um esquema ER e R uma representação relacional ER-dirigida deste esquema. Sejam também $h = (N_h \cup M_h \cup \{\Lambda\}, A_h)$ e $h' = (N_{h'} \cup M_h \cup \{\Lambda\}, A_{h'})$ GDNs de R para R , tais que:

- $N_h = N \cup \{P, Q\}$;
- $A_h = A \cup \{(P, Q)\}$;
- $\alpha(P) = P_1, A, P_2$ e $\alpha(Q) = Q_1, B, Q_2$
- h é um GDN natural;
- $N_{h'} = N \cup \{P, Q'\}$;
- $A_{h'} = (A - \{(R, Q), (Q, S) \mid (R, Q), (Q, S) \in N_h\}) \cup \{(P, Q')\} \cup \{(R, Q'), (Q', S) \mid (R, Q), (Q, S) \in N_h\}$
- $\alpha(Q') = Q_1, Q_2$;

onde P_1, P_2, Q_1, Q_2 são listas de atributos e A, B são atributos, $R, S, P, Q \in N_h \cup \{\Lambda\}$ e $R, S, P, Q' \in N_{h'} \cup \{\Lambda\}$. Se, para toda tupla t de $r(R)$, tivermos $t(A) = t(B)$ então $h \equiv h'$

Este teorema garante que se há um arco de um vértice P para um vértice Q em um GDN natural, os atributos de $\alpha(Q)$ que são redundantes com os de $\alpha(P)$ podem ser eliminados sem afetar as dependências de nulos. Isto é particularmente útil quando $\alpha(Q)$ possui atributos que foram “herdados” de uma chave primária e, neste caso, este atributo ocorre em $\alpha(P)$ sem a qualificação devida ao nível de Q na hierarquia do esquema.

Corolário 1 *Sejam R um esquema ER e R uma representação relacional ER-dirigida deste esquema. Sejam também $h = (N_h \cup M_h \cup \{\Lambda\}, A_h)$ e $h' = (N_{h'} \cup M_h \cup \{\Lambda\}, A_{h'})$ GDNs de R para R , tais que:*

- $N_h = N \cup \{N_E, N_F\}$;
- $A_h = A \cup \{(N_E, N_F)\}$;
- $\alpha(N_E) = E_1, E.A, E_2$ e $\alpha(N_F) = F_1, F.E.A, F_2$;
- h é um GDN natural;
- $N_{h'} = N \cup \{N_E, N_{F'}\}$;
- $A_{h'} = (A - \{(R, N_F), (N_F, S) \mid (R, N_F), (N_F, S) \in N_h\}) \cup \{(N_E, N_{F'})\} \cup \{(R, N_{F'}), (N_{F'}, S) \mid (R, N_F), (N_F, S) \in N_h\}$;
- $\alpha(N_{F'}) = F_1, F_2$;

onde E_1, E_2, F_1, F_2 são listas de atributos e $F.E.A$ é o atributo $E.A$ herdado de E por F ou incluído como atributo discriminador de F por ser E o seu identificador e $R, S, N_E, N_F \in N_h \cup \{\Lambda\}$ e $R, S, N_E, N_{F'} \in N_{h'} \cup \{\Lambda\}$. Os GDNs h e h' são equivalentes.

5.3.2 Eliminação de Ciclos Unitários

Teorema 2 *Sejam R um esquema ER e R uma representação relacional ER-dirigida deste esquema. Sejam também $h = (N_h \cup M_h \cup \{\Lambda\}, A_h)$ e $h' = (N_{h'} \cup M_h \cup \{\Lambda\}, A_{h'})$ GDNs de R para R , tais que:*

- $N_h = N \cup \{P, Q\}$, $A_h = A \cup \{(P, Q), (Q, P)\}$, $\alpha(P) = L_P$ e $\alpha(Q) = L_Q$, sendo h um GDN natural;
- $N_{h'} = N \cup \{PQ\}$ e $\alpha(PQ) = L_P, L_Q$;
- $A_{h'} = (A - \{(R, Q), (Q, S), (T, P), (P, U) \mid (R, Q), (Q, S), (T, P), (P, U) \in N_h\}) \cup \{(R, PQ), (PQ, S), (T, PQ), (PQ, U) \mid (R, Q), (Q, S), (T, P), (P, U) \in N_h\}$

onde L_P e L_Q são listas de atributos, $R, S, T, U \in (N_h \cap N_{h'}) \cup M_h \cup \{\Lambda\}$. Os GDNs h e h' são equivalentes.

De acordo com este teorema, é possível “unificar” em um GDN vértices P e Q tais que existem os arcos (P, Q) e (Q, P) , sem que as dependências de nulos sejam alteradas.

5.3.3 Eliminação do Primeiro Nível do Grafo de Dependências Nulos

Nas representações relacionais discutidas até agora, uma tupla de uma relação só existe se pelo menos os atributos discriminadores dos esquemas de entidade e relacionamento representado na raiz da árvore de colapsamento forem diferentes de nulo. Essa exigência é facilmente atendida se usarmos a declaração *not null* comum a várias LDD relacionais. Assim, se S é uma entidade representada na raiz de uma árvore colapsamento, podemos considerar o predicado de verificação $\mathcal{F}(\alpha(N_S) \rightsquigarrow \square)$ como sendo uma tautologia. Se essa consideração for feita, o Teorema 3 garante mais uma possibilidade de simplificação.

Teorema 3 *Sejam R um esquema ER e R uma representação relacional ER-dirigida deste esquema. Sejam também $h = (N_h \cup M_h \cup \{\Lambda\}, A_h)$ e $h' = (N_{h'} \cup M_h, A_{h'})$ GDNs de R para R tais que:*

- $N_{h'} = (N_h - \{\Lambda\}) - \{N_S \mid (\Lambda, N_S) \in A_h\}$;
- $A_{h'} = (A_h - \{(\Lambda, N_S) \mid (\Lambda, N_S) \in A_h\})$
 $\quad - \{(N_S, N_R) \mid (\Lambda, N_S), (N_S, N_R) \in A_h\}$

Os GDNs h e h' são equivalentes.

6 Exemplo de Utilização de Dependências de Nulos

Exemplo 2 Para o esquema de relação E^* do Exemplo 1 podemos definir as seguintes dependências de nulos:

- DN1.* $E^* : E.K_E \rightsquigarrow \square$
- DN2.* $E^* : E.K_E \rightsquigarrow E.A_E$
- DN3.* $E^* : E.K_E \rightsquigarrow F.K_F, F.E.K_E$
- DN4.* $E^* : F.K_F, F.E.K_E \rightsquigarrow F.A_F$
- DN5.* $E^* : F.K_F, F.E.K_E \rightsquigarrow R.E.K_E, R.F.K_F$
- DN6.* $E^* : R.E.K_E, R.F.K_F \rightsquigarrow R.A_R$
- DN7.* $E^* : R.E.K_E, R.F.K_F \rightsquigarrow F.K_F, F.E.K_E$
- DN8.* $E^* : [F.K_F, F.E.K_E]$
- DN9.* $E^* : [R.E.K_E, R.F.K_F]$

Os predicados de verificação das dependências de nulos definidas sobre o esquema de relação E^* são as fórmulas de F1 até F9 definidas no Exemplo 1.

Para que este conjunto de DNs se verifique, ou seja, para que $r(E^*)$ seja consistente sob o ponto de vista do conjunto de DNs, é necessário que o predicado formado pela conjunção destas fórmulas se verifique para qualquer estado do banco de dados, ou seja, em qualquer tupla t de $r(E^*)$.

Um possível GDN para a representação relacional E^* do Exemplo 1 é mostrado na Figura 2(a), onde:

$$\begin{aligned}
\alpha(N_E) &= E.K_E \\
\alpha(M_E) &= E.A_E \\
\alpha(N_F) &= F.K_F, F.E.K_E \\
\alpha(M_F) &= E.A_F \\
\alpha(N_R) &= R.E.K_E, R.F.K_F \\
\alpha(M_R) &= E.A_R
\end{aligned}$$

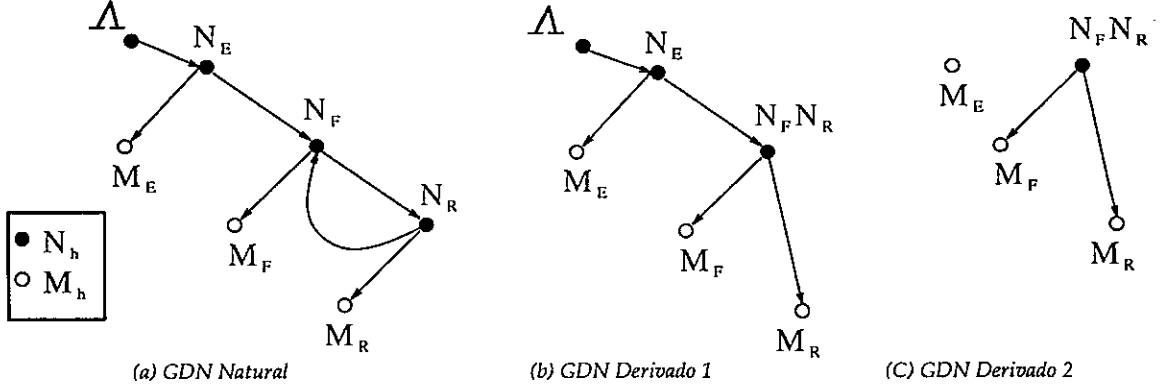


Figura 2: Grafos de dependências de nulos do exemplo 2

De acordo com este grafo, podemos verificar então que:

- $\neg(E.K_E = \lambda)$, de acordo com o arco (Λ, N_E) ;
- $\neg(E.K_E = \lambda) \vee (E.A_E = \lambda)$, de acordo com o arco (N_E, M_E) ;
- $\neg(E.K_E = \lambda) \vee ((F.K_F = \lambda) \wedge (F.E.K_E = \lambda))$, de acordo com arco (N_E, N_F) ;
- $(\neg(F.K_F = \lambda) \wedge \neg(F.E.K_E = \lambda)) \vee (F.A_F = \lambda)$, de acordo com arco (N_F, M_F) ;
- $(\neg(F.K_F = \lambda) \wedge \neg(F.E.K_E = \lambda)) \vee (R.E.K_E = \lambda \wedge R.F.K_F = \lambda)$, de acordo com arco (N_F, N_R) ;
- $(\neg(R.E.K_E = \lambda) \wedge \neg(R.F.K_F = \lambda)) \vee ((F.K_F = \lambda) \wedge (F.E.K_E = \lambda))$, de acordo com o arco (N_R, N_F) ;
- $(\neg(R.E.K_E = \lambda) \wedge \neg(R.F.K_F = \lambda)) \vee (R.A_R = \lambda)$, de acordo com o arco (N_R, M_R) ;
- $(\neg(F.K_E = \lambda) \vee \neg(F.K_E = \lambda))$, pois $N_E \in N_h$, o que, na realidade, é uma tautologia;
- $(\neg(F.K_F = \lambda) \wedge \neg(F.E.K_E = \lambda)) \vee (F.K_F = \lambda \wedge F.E.K_E = \lambda)$, pois $N_F \in N_h$;
- $(\neg(R.E.K_E = \lambda) \wedge \neg(R.F.K_F = \lambda)) \vee (R.E.K_E = \lambda \wedge R.F.K_F = \lambda)$, pois $N_R \in N_h$.

Seguindo os passos descritos na Seção 5.1 para o grafo da Figura 2(a), obteremos o conjunto de dependências de nulos descrito acima. De acordo com a Seção 5.3.1, os conjuntos α se tornam:

$$\begin{aligned}
\alpha(N_E) &= E.K_E \\
\alpha(N_F) &= F.K_F \\
\alpha(N_R) &= R.E.K_E
\end{aligned}$$

Assim, as dependências de nulos para o esquema de relação E^* podem ser reescritas da seguinte forma:

$$DN1. E^* : E.K_E \rightsquigarrow \square$$

$$DN2. E^* : E.K_E \rightsquigarrow E.A_E$$

$$DN3. E^* : E.K_E \rightsquigarrow F.K_F$$

$$DN4. E^* : F.K_F \rightsquigarrow F.A_F$$

$$DN5. E^* : F.K_F \rightsquigarrow R.E.K_E$$

$$DN6. E^* : R.E.K_E \rightsquigarrow R.A_R$$

$$DN7. E^* : R.E.K_E \rightsquigarrow F.K_F$$

$$DN8. E^* : \{F.K_F\}$$

$$DN9. E^* : \{R.E.K_E\}$$

sendo que as duas últimas dependências produzem tautologias como predicados de verificação.

Como o GDN da Figura 2(a) pode ser transformado no GDN da Figura 2(b), de acordo com a Secção 5.3.2, as dependências de nulos sobre E^* podem então ser reescritas da seguinte forma:

$$DN1. E^* : E.K_E \rightsquigarrow \square$$

$$DN2. E^* : E.K_E \rightsquigarrow E.A_E$$

$$DN3. E^* : E.K_E \rightsquigarrow F.K_F, R.E.K_E$$

$$DN4. E^* : F.K_F, R.E.K_E \rightsquigarrow F.A_F$$

$$DN5. E^* : F.K_F, R.E.K_E \rightsquigarrow R.A_R$$

$$DN6. E^* : \{F.K_F, R.E.K_E\}$$

De acordo com a Secção 5.3.3, o grafo de nulos da Figura 2(b) pode ser transformado no grafo da Figura 2(c), de modo que as dependências de nulos sobre E^* podem ser reescritas da seguinte forma:

$$DN1. E^* : F.K_F, R.E.K_E \rightsquigarrow F.A_F$$

$$DN2. E^* : F.K_F, R.E.K_E \rightsquigarrow R.A_R$$

$$DN3. E^* : \{F.K_F, R.E.K_E\}$$

sendo este o conjunto mínimo de dependências de nulos necessário para garantir a consistência da representação relacional, pois é obtido a partir do GDN da Figura 2(c) que é equivalente ao GDN da Figura 2(a).

7 Conclusões

Apresentamos neste trabalho uma discussão sobre o problema de manutenção da correção semântica em esquemas relacionais que servem para a representação otimizada de esquemas ER obtidos pelo método de projeto descrito em [7, 8]. Introduzimos nestes esquemas, um tipo de restrição de integridade intra-relação chamado dependência de nulos, destinado a reforçar esta correção semântica de tal forma que permaneça a característica básica do método que é a de reduzir o número de DPIs a serem validadas durante uma operação de atualização no banco de dados. Mesmo que sejam necessárias verificações adicionais, elas certamente terão um custo menor uma vez que são “internas” a cada relação. Além disso, foram apresentadas formas de simplificação que permitem reduzir o número de DNs necessárias sem o comprometimento da correção semântica.

Um aspecto importante referente às DNs é que elas podem ser implementadas em qualquer SGBD cuja LDD permita especificar regras ou asserções (por exemplo [1]), mesmo que somente possam ser associados atributos dentro de uma mesma relação, utilizando-se da semântica fornecida pelos predicados de verificação (Seção 4.1).

O assunto deste trabalho se insere no contexto do estudo iniciado em [7, 8]. No atual estágio deste estudo, o método de reprojeto descrito nestes artigos vem sendo detalhado e formalizado de tal forma que possa ser automatizado como já o foi o método de projeto.

Referências

- [1] ANSI X3H2-90-264 / ISO IEC JTC1 SC21 WG3, “(ISO working draft) Database Language SQL2”, Melton, J. (ed.) (Jul. 1990).
- [2] Albano, A., de Antonellis, V. and di Leva, A., *Computer-Aided Database Design: The DATAID Project*, North-Holland (1985).
- [3] Azar, N. and Pichat, E., “Translation of an extended entity-relationship model into the universal relation with inclusion formalism”, Proc. 5th Int’l. Conf. on the Entity-Relationship Approach, Dijon, France (Nov. 1986).
- [4] Batini, C., Ceri, S. and Navathe, S., *Conceptual Database Design: An Entity-Relationship Approach*, Benjamin Cummings (1992).
- [5] Casanova, M.A., Tucheran, L., Furtado, A.L. and Pacheco, A., “Optimization of relational schemas containing inclusion dependencies”, Proc. 15th Int’l. Conf. on Very Large Data Bases, Amsterdam, Holland (Sept. 1989).
- [6] Casanova, M.A., Tucheran, L., Gualandi, P.M., Pacheco, A. and Cavalcanti, M.R., A data definition language for an extended entity-relationship model, Technical Report CCR072, Rio Scientific Center, IBM Brazil (July 1989).
- [7] Casanova, M.A., Tucheran, L. and Laender, A.H.F., “Algorithms for designing and maintaining optimized relational representations of entity-relationship schemas”, Proc. 9th Int’l. Conf. on Entity-Relationship Approach, Lausanne, Switzerland (Oct. 1990).

- [8] Casanova, M.A., Tucherman, L. and Laender, A.H.F., "On the design and maintenance of optimized relational representations of entity-relationship schemas" *Data and Knowledge Engineering* 11:1 (1993).
- [9] Chen, P.P., "The entity-relationship model: toward a unified view of data", *ACM Transactions on Database Systems* 1:1 (1976).
- [10] Ceri, S. (editor), *Methodology and Tools for Database Design*, North-Holland (1983).
- [11] Codd, E.F. "A relational model for large shared data banks", *Communications of the ACM* 6:13 (1970).
- [12] Elmasri, R. and Navathe, S., *Fundamentals of Database Systems*, Benjamin Cummings (1989).
- [13] Hull, R. and King, R., "Semantic database modeling: survey, applications and research issues", *ACM Computing Surveys* 3:19 (1987).
- [14] Laender, A.H.F. and Flynn, D.J., "A semantic comparison of the modelling capabilities of the ER and NIAM models", *Proc 12th Int'l Conf. on Entity-Relationship Approach*, Arlington, Texas (Dec. 1993).
- [15] Laender, A.H.F. and Fernandes, A.C., "MER+ : Uma extensão do modelo de entidades e relacionamentos para projeto conceitual de bancos de dados", *Revista Brasileira de Computação* 5:1 (1989).
- [16] Levene, M. and Loizou, G., "Semantics for null extended nested relations" *ACM Transactions on Database Systems*, 18:3 (Sept. 1993).
- [17] Markowitz, V.M. and Shoshani, A., "Representing extended entity-relationship structures in relational databases: a modular approach", *ACM Transactions on Database Systems*, 17:3 (Sept. 1992).
- [18] Navathe, S.B., "Evolution of data modeling for databases", *Communications of the ACM*, 35:9 (September 1992).
- [19] Ridolfi, L., Carvalho, A.P. and Casanova, M.A., "Uma ferramenta para projeto conceitual de banco de dados baseada no modelo entidade relacionamento, Relatório Técnico CCR130, Centro Científico do Rio, IBM Brasil (Nov. 1991).
- [20] Silva, A.S., "Uma Contribuição para o Problema de Manutenção de Representações Relacionais Otimizadas de Esquemas de Entidade e Relacionamento, Dissertação de Mestrado, Departamento de Ciência da Computação — Universidade Federal de Minas Gerais (em preparação).
- [21] Teorey, T.J., *Database Modeling and Design: The Entity-Relationship Approach*, Morgan Kaufmann (1990).
- [22] Teorey, T.J., Yang, D. and Fry, J.P., "A logical design methodology for relational databases using the extended entity-relationship model", *ACM Computing Survey* 18:2 (June 1986).