

OPERAÇÕES SOBRE CONJUNTOS DE ENTIDADES E DE RELACIONAMENTOS  
ORGANIZADOS EM GRAFOS DE ESPECIALIZAÇÃO

Marco A. Casanova\*

Luiz Tucherma\*\*

**SUMÁRIO**

Este trabalho propõe inicialmente construtores especiais que formalizam e estendem as abstrações de generalização e subconjunto. Em seguida, analisa várias operações para a manutenção de conjuntos de objetos organizados de acordo com os construtores. Finalmente, define uma semântica precisa para os construtores e as operações, que destaca alguns aspectos importantes do modelo entidade-relacionamento.

**ABSTRACT**

Special constructs that formalize and extend the generalization and subset abstractions are first proposed. Then, several operations to maintain sets of objects organized according to the constructs are analysed. Finally, the constructs and operations are given a precise semantics that brings up some key aspects of the entity-relationship model.

\* Doutor em Matemática Aplicada (Harvard Univ., 1979); teoria de bancos de dados, sistemas de gerência de bancos de dados distribuídos, projeto conceitual de banco de dados e programação em lógica;  
Centro Científico Rio, IBM Brasil Ltda, Caixa Postal 4624, 20.001, Rio de Janeiro, RJ

\*\* Mestre em Ciência da Computação (PUC/RJ, 1983); sistemas de bancos de dados, sistemas de gerência de bancos de dados distribuídos e modelagem conceitual de banco de dados.  
Centro Científico Rio, IBM Brasil Ltda, Caixa Postal 4624, 20.001, Rio de Janeiro, RJ

## 1. INTRODUÇÃO

Entre as principais abstrações usadas na fase de projeto conceitual de um banco de dados encontram-se as hierarquias de generalização e de subconjunto. A maioria das definições encontradas na literatura para a hierarquia de generalização se baseiam na existência de uma condição de seleção (predicado) que divide, de acordo com o resultado desta seleção, uma classe em várias subclasses. Esta divisão pode particionar exaustivamente uma classe em todas as suas possíveis subclasses.

Para a hierarquia de subconjunto, a maioria das definições não dependem de nenhuma condição de seleção e estabelecem que a especialização de uma classe de entidade E não implica na partição de E. Isto significa que alguma entidade em E pode não estar em qualquer uma das suas subclasses. Assim, a única condição imposta por este tipo de hierarquia é que todas as entidades de uma subclasse de E devem também ocorrer em E. Alguns autores também diferenciam a hierarquia de subconjunto da hierarquia de generalização pela possibilidade da existência de sobreposição ou não dos subconjuntos, respectivamente.

Este trabalho propõe dois construtores que juntos capturam e estendem as abstrações de generalização e subconjunto e também sugere várias operações interessantes para a manutenção de conjuntos de entidades e de relacionamentos organizados de acordo com estas abstrações.

Diversas extensões ao Modelo Entidade-Relacionamento original [4] encontram-se em [2,3,5,6,9,10,11]. Em particular, propostas que capturam hierarquias de generalização e de subconjuntos encontram-se em [2,5,7,8,10,11].

Este trabalho está dividido da seguinte maneira. A seção 2 descreve a sintaxe e a semântica dos construtores. A seção 3 apresenta exemplos do uso dos construtores. A seção 4 introduz operações para a manipulação de conjuntos organizados de acordo com os construtores. Finalmente a seção 5 contém as conclusões.

## 2. CONSTRUTORES PARA CAPTURAR HIERARQUIAS DE GENERALIZAÇÃO E SUBCONJUNTO

### 2.1 Sintaxe da linguagem

#### 2.1.1 Declarações de Esquemas de Entidades e de Relacionamentos

A classe dos construtores nesta e nas seções seguintes são definidas no contexto de um dado esquema entidade-relacionamento ou, abreviadamente, um dado esquema-ER.

A definição de um conjunto de entidades segue o padrão usual, exceto que permitimos ao projetista do banco de dados definir que os valores dos atributos de cada entidade em um dado conjunto devem satisfazer certas restrições de integridade. Esta facilidade está em linha com a noção de especialização qualificada (veja seção 2.1.2). Uma *declaração de esquema de entidades* é então qualquer expressão da seguinte forma:

defina entidade E

[atributos  $A_1 D_1, \dots, A_n D_n$ ]  
[chave  $K_0$ ] ... [chave  $K_p$ ]  
[restricao  $Q_0$ ] ... [restricao  $Q_q$ ]

Dizemos que E é o *nome*,  $A_1, \dots, A_n$  é a lista de *nomes de atributos*,  $K_0$  é a *chave primária* e  $K_1, \dots, K_p$  são as *chaves alternativas* do esquema. Dizemos também que  $D_i$  é o tipo do domínio de  $A_i$ , para  $i=1, \dots, n$ , e que  $A_i$  é *multivalorado* ou *aceita valores nulos* se  $D_i$  também o for. Finalmente, dizemos que  $Q_0, \dots, Q_q$  são as *assertivas* definidas para E.

Exigimos que:

- E0. E deve ser um nome válido e distinto de todos os outros nomes de esquemas de entidades ou relacionamento do esquema-ER;
- E1.  $A_1, \dots, A_n$  devem ser nomes válidos e distintos de atributos;
- E2.  $D_1, \dots, D_n$  devem ser tipos de domínios válidos no esquema-ER;
- E3. para cada  $i$  em  $[0, p]$ ,  $K_i$  deve ser uma lista de nomes de atributos de E tal que
  - E3.0. todos os nomes de atributos em  $K_i$  são distintos;
  - E3.1. cada nome de atributo em  $K_i$  não é multivalorado e não admite valores nulos;
  - E3.2.  $K_i$  não está contido em  $K_j$ , para qualquer  $j$  em  $[0, p]$  com  $i \neq j$ ;
- E4.  $Q_0, \dots, Q_q$  devem ser qualificações sobre atributos do esquema (herdados ou não, veja seção 2.1.2).

O conjunto exato dos tipos de domínios válidos não necessitam ser especificados neste nível de discussão, mas assumimos que cada tipo de domínio indica se o atributo é multivalorado ou admite valores nulos. Da mesma forma, a sintaxe exata da qualificação pode ser deixada em aberto. Dizemos que uma entidade  $e$  *satisfaz* uma qualificação  $Q_i$  quando os valores dos atributos de  $e$  tornam  $Q_i$  verdadeiro.

Uma *declaração de um esquema de relacionamentos* é qualquer expressão da forma:

defina relacionamento R  
sobre  $O_1$  [como  $N_1$ ] [total], ...,  $O_m$  [como  $N_m$ ] [total]  
[atributos  $A_1 D_1, \dots, A_n D_n$ ]  
[identificador  $I_0$ ] ... [identificador  $I_p$ ]  
[restricao  $Q_0$ ] ... [restricao  $Q_q$ ]

Dizemos que R é o *nome*,  $A_1, \dots, A_n$  é a lista de *nomes de atributos*,  $I_0$  é o *identificador primário* e  $I_1, \dots, I_p$  são os *identificadores alternativos* do esquema. Dizemos também que o esquema de relacionamentos R tem  $m$  *papéis*. O esquema no  $i^{\text{ésimo}}$  papel é  $O_i$  e o *nome* do  $i^{\text{ésimo}}$  papel é  $N_i$ , se especificado, ou o próprio  $O_i$ , caso contrário. Quando "total" é especificado para " $O_i$  [como  $N_i$ ]", dizemos que R é *total* no  $i^{\text{ésimo}}$  papel do esquema.

Os identificadores capturam a anotação usual que rotula com "1" ou "n" os arcos que deixam o losango que representa um conjunto de relacionamentos em um diagrama ER, mas eles são mais gerais do que esta convenção de anotação.

A partir de agora, quando nos referirmos a um esquema queremos dizer ou um esquema de entidades ou um esquema de relacionamentos.

Exigimos que:

- R0. R deve ser um nome válido e distinto de todos os outros nomes de esquemas do esquema-ER;
- R1, R2. (como E1 e E2);
- R3.  $O_1, \dots, O_m$  devem ser nomes de esquemas válidos, não necessariamente distintos, mas definidos no esquema-ER, com  $m \geq 2$ ;
- R4. os nomes dos papéis de R devem ser distintos;
- R5. para cada  $i$  em  $[0, p]$ ,  $I_i$  deve ser uma lista de nome de papéis de R tal que
  - R5.0. todos os nomes de papéis em  $I_i$  são distintos;
  - R5.1.  $I_i$  não está contido em  $I_j$ , para qualquer  $j$  em  $[0, p]$  com  $i \neq j$ ;
- R6. (como E4).

Note que, por R4, os nomes dos papéis são únicos. Portanto, podemos nos referir a um papel específico de R pelo seu nome ao invés de sua posição.

Como permitimos relacionamentos sobre relacionamentos, devemos também evitar definir um esquema de relacionamentos R sobre si mesmo pois, de outra forma, a semântica de R seria indefinida. Mais precisamente, exigimos ainda que:

- R7. o esquema-ER não pode ter uma seqüência de esquemas de relacionamentos  $O_0, O_1, \dots, O_{n-1}, O_0$  tal que  $O_i$  é sobre  $O_{i+1}$ , para  $i = 0, \dots, n-1$ , onde '+' é modulo  $n$ .

### 2.1.2 Declarações de Especialização

Permitimos ao projetista do banco de dados organizar os esquemas de entidades e de relacionamentos de um dado esquema-ER como um grafo de especialização com as seguintes características.

Em primeiro lugar, o grafo de especialização não necessita ser uma árvore, isto é, uma hierarquia estrita. Na verdade, nem exigimos que o grafo seja acíclico, embora grafos de especialização cíclicos devam ser evitados pois trazem certas redundâncias.

Um esquema de entidades E pode ser definido como uma especialização *qualificada* de outro esquema de entidade F, com qualificação C, o que implica que E denotará sempre o conjunto de entidades de F que satisfazem C. Por conseguinte, exigimos que um esquema de entidades E seja uma especialização qualificada de no máximo um outro esquema de entidades.

Um esquema de entidades E pode também ser definido como uma especialização *simples* de F, indicando que E deve sempre denotar um subconjunto do conjunto de entidades associada com F. Assim, um esquema de entidades pode ser uma especialização simples de mais de um esquema e, talvez ao mesmo tempo, ser uma especialização qualificada de um esquema de entidades.

Permitimos também definir um esquema de relacionamentos como uma especialização simples de um outro esquema de relacionamentos. Proibimos, para este caso,

especializações qualificadas porque elas iriam criar interrelacionamentos complexos entre os conjuntos de entidades e de relacionamentos, que seriam difíceis de garantir. Exigimos que, se  $R$  é um esquema de relacionamentos sobre  $P_1, \dots, P_n$  e  $S$  é um esquema de relacionamentos sobre  $Q_1, \dots, Q_n$  tal que  $S$  é uma especialização de  $R$ , então  $Q_i$  ou é igual a ou é uma especialização de  $P_i$ . Esta exigência garante que é semanticamente possível comparar  $S$  and  $R$ .

Introduzimos também o conceito de totalidade e exclusão mútua para especializações simples. Se declararmos que  $O$  é *totalmente especializado* em  $O_1, \dots, O_n$ , então qualquer objeto em  $O$  deve pertencer a  $O_i$ , para algum  $i$ . Ortogonalmente, se declararmos que  $O_1, \dots, O_n$  são *mutuamente exclusivos* então nenhum objeto em  $O$  pode pertencer a mais do que um  $O_i$ , para todo  $i$ . Estas são então restrições de integridade que relacionam  $O, O_1, \dots, O_n$ .

Permitimos especificar totalidade e exclusão mútua apenas para as especializações simples pelas seguintes razões. Primeiro, como especializações qualificadas definem conjuntos de objetos baseados nos valores dos atributos, o projetista do banco de dados pode, para este caso, expressar exclusão mútua através do uso de uma definição apropriada para as qualificações e expressar totalidade definindo uma assertiva para o esquema generalizado. Entretanto, não podemos aplicar a mesma estratégia para especializações simples uma vez que os conjuntos de objetos neste segundo caso são definidos "manualmente", ou seja, quando o usuário adiciona os objetos nos seus conjuntos.

Finalmente, permitimos especificar no esquema-ER mais de uma declaração de especialização para o mesmo esquema  $O$ , mas não permitimos especificar mais de uma declaração indicando que um esquema  $O$  é uma especialização de um esquema  $P$ .

Mais precisamente, introduzimos uma *declaração de especialização* como qualquer expressão da forma:

especialize  $O$  [totalmente] [exclusivamente]  
em  $O_1$  [onde  $C_1$ ], ...,  $O_m$  [onde  $C_m$ ]

Dizemos que  $O$  é o esquema *especializado* na declaração e que  $O_i$  é uma *especialização simples* de  $O$ , se a cláusula onde  $C_i$  não for especificada, ou que  $O_i$  é uma *especialização qualificada* de  $O$  em caso contrário. Dizemos também que  $O$  é uma *generalização* de  $O_1, \dots, O_m$ , que é total se "totalmente" for especificado. Da mesma forma, se "exclusivamente" for especificado, dizemos que  $O_1, \dots, O_m$  são especializações *mutuamente exclusivas* de  $O$ .

Antes de prosseguirmos, introduziremos algumas definições auxiliares. Seja  $S$  um esquema-ER. O *grafo de especialização* de  $S$ ,  $G(S) = (V, A)$ , é um grafo dirigido tal que  $V$  é o conjunto de nomes de esquemas de  $S$  e um par  $(O, P)$  está em  $A$  se e somente se  $O$  é uma especialização simples ou qualificada de  $P$  em  $S$ .

Dados  $F, G \in V$ , dizemos que  $G$  é uma *especialização transitiva* de  $F$  e que  $F$  é uma *generalização transitiva* de  $G$  se e somente se existe um caminho de  $G$  para  $F$  em  $G(S)$ ;

dizemos que  $G$  é uma *especialização qualificada transitiva* de  $G$  se e somente se existe um caminho de  $G$  para  $F$  em  $G(S)$  tal que, para todos os arcos  $(H,I)$  do caminho,  $H$  é uma especialização qualificada de  $I$ .

Finalmente, dados  $E,F,G \in \mathcal{V}$ , dizemos que  $E$  está *entre*  $G$  e  $F$  se e somente se  $G$  é uma especialização transitiva de  $E$  e  $E$  é uma especialização transitiva de  $F$ . Dado um conjunto de nós  $\mathbf{N}$ , denotaremos a união dos conjuntos de generalizações transitivas dos nós em  $\mathbf{N}$  por  $gen(\mathbf{N})$  e a união dos conjuntos de nós entre  $G$  e cada nó em  $\mathbf{N}$  por  $entre(G,\mathbf{N})$ .

Por simplicidade, introduziremos  $T$  como um nó artificial, chamado *topo*, tal que  $T$  é uma generalização transitiva de todos os nós e nenhum nó é uma generalização de  $T$ . Assim, temos que  $gen(\{T\}) = \emptyset$  e  $entre(E,\{T\}) = gen(E)$ , para todo nó  $E$ .

Dizemos que um atributo  $A$  (ou uma chave  $K$ ) é *nativo* em  $O$  se e somente se  $A$  é o nome de um atributo (ou  $K$  é uma chave) definida no esquema cujo nome é  $O$ . Dizemos que um atributo  $A$  (ou uma chave  $K$ ) é *herdado por*  $O$  de  $P$  se e somente se  $A$  é o nome de um atributo de  $P$  (ou  $K$  é a chave de  $P$ ) e  $O$  é uma especialização transitiva de  $P$ . Note que devemos especificar o esquema originário de  $A$  (ou  $K$ ) uma vez que  $O$  pode ser uma especialização transitiva de esquemas cujos atributos podem ter o mesmo nome (ou cujas chaves podem ser iguais). Assim, no contexto do esquema  $O$  e sempre que necessário, usaremos a notação  $P.A$  (ou  $P.K$ ) para nos referirmos ao atributo (ou chave) herdado por  $O$  de  $P$ .

Para as declarações de especialização exigimos que:

- S0.  $O, O_1, \dots, O_m$  sejam nomes distintos de esquemas definidos no esquema-ER;
- S1. para cada  $i$  em  $[1,m]$ ,  $C_i$ , se especificado, deve ser uma qualificação sobre atributos de  $O$  envolvendo pelo menos um atributo nativo de  $O$ ;
- S2. para cada  $i$  em  $[1,m]$ , não pode existir outra declaração de especialização indicando que  $O_i$  é uma especialização de  $O$ ;
- S3. para cada  $i$  em  $[1,m]$ , se  $O_i$  é uma especialização qualificada de  $O$  então  $O_i$  não pode ser uma especialização qualificada de qualquer outro esquema de entidades;
- S4. se 'totalmente' ou 'exclusivamente' são especificados então  $O_1, \dots, O_m$  devem ser especializações simples;
- S5. se  $O$  é o nome de um esquema de relacionamentos então nenhuma qualificação pode ser especificada;
- S6. se  $O$  é o nome de um esquema de relacionamentos sobre  $P_1, \dots, P_n$  então, para cada  $i$  em  $[1,m]$ ,  $O_i$  deve ser o nome de um esquema de relacionamentos sobre esquemas  $Q_1, \dots, Q_n$  tal que  $Q_j$  ou é igual a ou é uma especialização transitiva de  $P_j$ , para cada  $j$  em  $[1,n]$ .

Para o Requisito S1, consideramos que  $C_i$  deve envolver pelo menos um atributo nativo de  $O$  uma vez que, em caso contrário,  $O_i$  poderia ser redefinido como uma especialização qualificada de alguma generalização transitiva de  $O$ , isto é, o esquema onde todos os atributos em  $C_i$  são definidos. Em vista de S4, é necessário especificar duas declarações separadas de especialização para o esquema  $O$  quando as especializações simples de  $O$  modelam uma classificação total ou exclusiva de  $O$ , mas  $O$  possui ainda uma ou mais especializações qualificadas.

### 2.1.3 Declarações de Exclusão Mútua

Esta seção introduz uma forma bem geral para se definir que vários conjuntos de objetos são mutuamente exclusivos, generalizando assim a palavra-chave exclusivamente da declaração de especialização.

Em geral, o projetista do banco de dados pode indicar que dois esquemas  $O$  e  $P$  devem sempre denotar conjuntos disjuntos de objetos, desde que as seguintes condições sejam satisfeitas. Primeiro,  $O$  e  $P$  devem ser comparáveis, o que formalizamos exigindo que  $O$  e  $P$  especializem transitivamente um terceiro esquema  $Q$ . Segundo, exigimos que  $O$  e  $P$  não sejam especializações qualificadas de outros esquemas  $O'$  e  $P'$ , já que a exclusão mútua neste caso criaria iterações complexas com as qualificações.

Assim, uma *declaração de exclusão mútua* é qualquer expressão da forma

$$\text{exclua } (O_{11}, \dots, O_{1n_1}) \mid \dots \mid (O_{k1}, \dots, O_{kn_k})$$

Para cada  $p, r \in [1, k]$ , com  $p \neq r$ , para cada  $q \in [1, n_p]$  e cada  $s \in [1, n_r]$ , dizemos que  $O_{pq}$  and  $O_{rs}$  são *mutuamente exclusivos*.

Exigimos que:

- M0.  $O_{11}, \dots, O_{1n_1}, \dots, O_{k1}, \dots, O_{kn_k}$  devem ser nomes de esquemas definidos no esquema-ER que não sejam especializações qualificadas de qualquer esquema;
- M1. existe um esquema  $Q$  tal que  $O_{11}, \dots, O_{1n_1}, \dots, O_{k1}, \dots, O_{kn_k}$  são especializações transitivas de  $Q$ ;
- M2.  $O_{pq}$  e  $O_{rs}$  devem ser distintos, para cada  $p, r \in [1, k]$ , com  $p \neq r$ , para cada  $q \in [1, n_p]$  e cada  $s \in [1, n_r]$ .

### 2.2 Semântica

Esta seção se preocupará com a noção de estado e estado consistente de um esquema-ER. Resumidamente, um estado de um esquema-ER  $S$  mapeia cada esquema de entidades de  $S$  em um conjunto de entidades, obtidas de um conjunto comum  $E$ , cada esquema de relacionamentos em um subconjunto do produto cartesiano do conjunto de entidades envolvidas, e cada atributo de um esquema em uma função que mapeia cada objeto em um valor obtido do domínio apropriado. É muito importante salientar que, ao contrário do modelo relacional, entidades e relacionamentos tem uma existência independente dos valores dos atributos, que são necessários entretanto para localizá-las no banco de dados [1].

Definiremos a *aridade* de um esquema  $O$  da seguinte forma:

- se  $O$  é um esquema de entidades, então a aridade de  $O$  é 1;
- se  $O$  é um esquema de relacionamentos sobre  $O_1, \dots, O_m$  então a aridade de  $O$  é a soma das aridades de  $O_1, \dots, O_m$ .

Fatoramos a semântica dos tipos de domínios introduzindo o conceito de *função domínio*  $D$  como qualquer função que mapeia cada tipo de domínio válido  $D$  em um conjunto  $D(D)$ .

Um estado para um esquema-ER  $S$  é uma tripla  $\alpha = (D, E, I)$ , onde:

- 1)  $D$  é a função domínio;
- 2)  $E$  é um conjunto não-vazio, chamado de *domínio entidade*;
- 3)  $I$  é a função que satisfaz as seguintes condições:
  - a) o domínio de  $I$  é o conjunto dos nomes de todos os esquemas definidos em  $S$ , união com o conjunto de todos os pares  $(O, A)$  onde  $O$  é o nome de um esquema e  $A$  é o nome de um atributo de  $O$ ;
  - b) se  $E$  é o nome de um esquema de entidades de  $S$ , então  $I(E)$ , o *conjunto de entidades* de  $E$  em  $\alpha$ , é um subconjunto finito de  $E$ ;
  - c) se  $R$  é o nome de um esquema de relacionamentos de  $S$  com aridade  $n$ , então  $I(R)$ , o *conjunto de relacionamentos* de  $R$  em  $\alpha$ , é um subconjunto finito de  $E^n$ ;
  - d) se  $A$  é o nome de um atributo de um esquema  $O$  de  $S$  e se  $D$  é o tipo de domínio de  $A$ , então  $I((O, A))$ , a *função de atributo* para o atributo  $A$  de  $O$  em  $\alpha$ , é uma função de  $I(O)$  em  $D(D)$ .

Lembre que, pelos requisitos impostos sobre esquemas-ER, dois esquemas em  $S$  não tem o mesmo nome assim como dois atributos de um esquema não tem o mesmo nome. Logo, o domínio da função  $I$  terá exatamente um elemento para cada objeto do esquema-ER  $S$ .

Se  $R$  é o nome de um esquema de relacionamentos  $n$ -ário e  $N$  é o nome do  $i$ ésimo papel de  $R$ , indique por  $\pi(i, I(R))$  a  $i$ ésima projeção de um conjunto de relacionamentos  $I(R)$  e, da mesma forma, indique por  $\pi(i, r)$  e  $\pi(N, r)$  o  $i$ ésimo elemento de uma tupla  $r$  em  $I(R)$ .

Um estado  $\alpha = (D, E, I)$  de  $S$  é *consistente* se e somente se

- 1) para qualquer esquema de entidades  $E$  de  $S$ , para qualquer chave primária ou alternativa  $K$  de  $E$ ,

$$\forall e \forall e' (e \in I(E) \wedge e' \in I(E) \wedge \bigwedge_{A \in K} (I((E, A))(e) = I((E, A))(e')) \Rightarrow e = e')$$

- 2) se  $R$  é o nome de um esquema de relacionamentos de  $S$  sobre  $O_1, \dots, O_m$ , então  $I(R)$  é um subconjunto de  $I(O_1) \times \dots \times I(O_m)$ ;
- 3) para qualquer esquema de relacionamentos  $R$  de  $S$ , se  $K$  é um identificador de  $R$  então

$$\forall r \forall r' (r \in I(R) \wedge r' \in I(R) \wedge \bigwedge_{N \in K} (\pi(N, r) = \pi(N, r')) \Rightarrow r = r')$$

- 4) para qualquer esquema de relacionamentos  $R$  de  $S$ , se  $O$  é o esquema no  $i$ ésimo papel e se  $R$  é total no  $i$ ésimo papel, então  $I(O) \subseteq \pi(i, I(R))$ ;
- 5) para quaisquer dois esquemas  $O$  e  $P$  de  $S$ , se  $P$  é uma especialização simples de  $O$  então  $I(P) \subseteq I(O)$ ;
- 6) para quaisquer dois esquemas de entidades  $E$  e  $F$  de  $S$ , se  $F$  é uma especialização qualificada de  $E$ , com qualificação  $C$ , então  $I(F) = \{e \in I(E) \mid e \text{ satisfaz } C\}$ ;

- 7) para quaisquer esquemas  $O, O_1, \dots, O_m$ , se  $O$  é uma generalização total de  $O_1, \dots, O_m$ , então  $I(O) \subseteq I(O_1) \cup \dots \cup I(O_m)$ ;
- 8) para quaisquer dois esquemas  $O$  e  $P$ , se  $O$  e  $P$  são mutuamente exclusivos então  $I(O) \cap I(P) = \emptyset$ .
- 9) para cada esquema  $O$ ,  $I(O)$  deve satisfazer a todas as assertivas definidas para  $O$ .

Como consequência da definição de estado consistente, temos que, se  $O$  especializa transitivamente  $O'$ , então o conjunto de objetos associados a  $O$  é sempre um subconjunto do conjunto de objetos associados a  $O'$ . Portanto, qualquer função de atributo associada com um atributo de  $O'$  é definida sobre todos os objetos do conjunto associado a  $O$ . Esta propriedade nos permite estender um estado consistente  $\alpha = (D, E, I)$  de um esquema-ER  $S$  para dar semântica aos atributos herdados, da seguinte forma:

- 1) o domínio de  $I$  é estendido para incluir todos os pares da forma  $(O, (O', A))$  onde  $A$  é herdado por  $O$  de  $O'$ ;
- 2)  $I((O, (O', A)))$  é a função  $I((O', A))$  restrita ao conjunto  $I(O)$ .

### 3. UM EXEMPLO INFORMAL

Ilustraremos nesta seção o uso dos construtores apresentados na seção 2, combinando especializações simples e qualificadas de forma a capturar conceitos bastante complexos do mundo real.

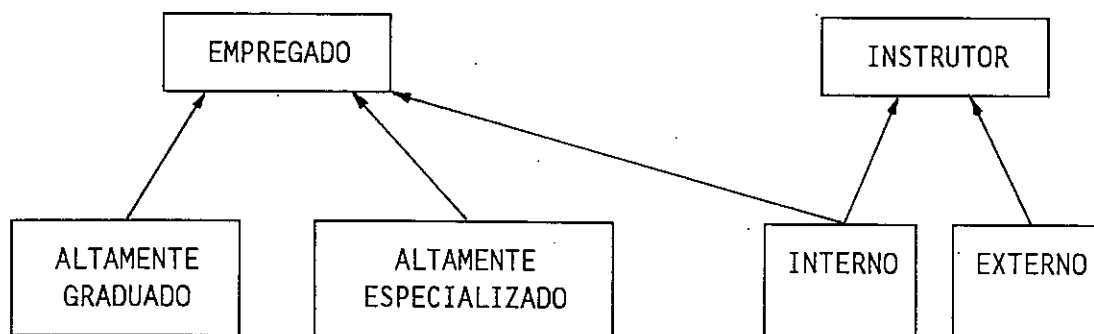


Figura 1

Classificamos EMPREGADO em duas especializações qualificadas, ALTAMENTE\_GRADUADO e ALTAMENTE\_ESPECIALIZADO, e em uma especialização simples, INTERNO. Além disso, categorizamos INSTRUTOR em duas especializações qualificadas, INTERNO e EXTERNO, que são subconjuntos disjuntos. Assim, INTERNO representa um empregado que ensina algum curso. As declarações de especialização apropriadas são descritas a seguir e o diagrama ER correspondente apresentado na Figura 1.

especialize INSTRUTOR

em EXTERNO onde TIPO = 'EXTERNO',  
INTERNO onde TIPO = 'INTERNO';

especialize EMPREGADO

em INTERNO,  
ALTAMENTE\_GRADUADO onde EDUCACAO = 'PHD' ou  
EDUCACAO = 'MSC',  
ALTAMENTE\_ESPECIALIZADO onde ESPECIALIZACAO = 'TECNICA' e  
EXPERIENCIA >= 10;

Observe que podemos ter instâncias de EMPREGADO que não são instâncias das duas especializações qualificadas, mas que são instâncias de INTERNO. Além disso, podemos também ter instâncias de EMPREGADO que não são classificados em qualquer das três subclasses.

#### 4. OPERAÇÕES EM GRAFOS DE ESPECIALIZAÇÃO

Propomos nesta seção um conjunto de operações para a manutenção de conjuntos de objetos organizados como um grafo de especialização. Por uma questão de simplicidade, consideraremos somente grafos de especialização de conjuntos de entidades e manteremos a discussão em um nível informal e exploratório.

Assumimos nesta subseção que a inclusão de uma nova entidade em um conjunto sempre propaga para as generalizações transitivas e para as especializações transitivas qualificadas do conjunto, dependendo das qualificações, e que a remoção de uma entidade de um conjunto sempre propaga para as especializações transitivas e para as generalizações transitivas qualificadas do conjunto, dependendo também das qualificações. Esta premissa pode ser relaxada através da introdução de outros comandos indicando como a propagação deve se dar. Além disso, definimos as operações de forma a preservarem: (1) as restrições semânticas relativas as declarações de especialização e exclusão mútua (itens 1, 5, 6, 7, 8 e 9 da definição de estado consistente) e (2) o conceito de que cada entidade existe independentemente.

Consideremos primeiro uma *operação de remoção* da forma

remove de E [onde Q]

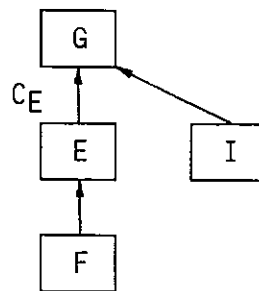
onde E é o nome de um esquema de entidade e Q é uma seleção, cuja sintaxe exata não será especificada, que define, em cada estado do esquema-ER, um subconjunto de um conjunto de entidades associada com E. A semântica desta operação pode ser resumidamente descrita da seguinte maneira:

- 1) construa o conjunto **D** de todas as entidades no conjunto correntemente associado com E que satisfaçam a Q (se Q é omitido, tome **D** como o conjunto de todas as entidades no conjunto correntemente associado com E);
- 2) execute  $remove^*(E, D)$ .

A operação  $remove^*(E, D)$  não é acessível ao usuário, isto é, ela somente auxilia na definição da semântica da operação original remove. Ela é definida recursivamente da seguinte forma:

- 1) se a remoção de todas as entidades em **D** do conjunto correntemente associado com **E** não violar nenhuma restrição de totalidade envolvendo **E**,  
então remova todas as entidades em **D** do conjunto correntemente associado com **E**;  
senão rejeite a operação;
- 2) para cada **F** tal que **F** especializa **E**, execute **remove\*(F,D)**;
- 3) para cada **G** tal que **E** é uma especialização qualificada de **G** com qualificação  $C_G$  faça:
  - a) seja  $D_G$  o conjunto de todos os  $e \in D$  tal que  $e$  satisfaça  $C_G$ ;
  - b) execute **remove\*(G,D)**;

Por exemplo, considere o seguinte grafo de especialização:



A remoção de um conjunto de entidades de **E** propagará primeiro para baixo em direção a **F**, depois para cima em direção a **G**, via a qualificação  $C_E$ , e finalmente para baixo na direção de **I**.

Consideraremos agora uma primeira forma de inclusão, que chamamos de *classificação*, que toma uma entidade  $e$  do conjunto associado com os esquemas  $H_1, \dots, H_n$  e inclui  $e$  em uma especialização transitiva comum **E** de  $H_1, \dots, H_n$ , instanciando os valores dos novos atributos de  $e$ . Se **E** é uma especialização de algum esquema **I** que não está nem entre  $H_i$  e **E** e nem é uma generalização transitiva de  $H_i$ , para algum  $i \in [1, n]$ , então  $e$  já deve estar no conjunto associado com **I**. Se **G** está entre **E** e  $H_i$ , para algum  $i \in [1, n]$ , então  $e$  é de fato incluído em **G** como efeito colateral, se já não estiver presente. Note que isto é possível uma vez que todos os atributos de **G** são na verdade atributos (herdados) de **E** e, portanto, eles devem ser especificados na classificação.

O formato geral da *operação de classificação* é:

classifique de  $H_1$  onde  $Q_1$ , ..., de  $H_n$  onde  $Q_n$   
em **E** fazendo  $X = \bar{x}$

onde, para cada  $i \in [1, n]$ ,  $H_i$  é um esquema de entidades,  $Q_i$  é uma seleção que define um subconjunto do conjunto de entidades correntemente associado com  $H_i$ , **E** é uma especialização transitiva de  $H_1, \dots, H_n$ ,  $X$  é uma lista de todos os atributos nativos de **E** ou herdados dos esquemas entre **E** e  $H_1, \dots, H_n$  e  $\bar{x}$  é a lista de valores para os atributos em  $X$ .

A semântica para esta operação é:

- 1) para cada  $i \in [1, n]$ , seja  $I_i$  o conjunto de todas as entidades, no conjunto correntemente associado com  $H_i$ , que satisfazem  $Q_i$ ;
- 2) seja  $I$  a interseção de  $I_1, \dots, I_n$ ;  
se  $I$  contém mais de uma entidade,  
então rejeite a operação;  
senão seja  $e$  a única entidade em  $I$ ;
- 3) execute  $\text{classifique}^*(e, \mathbf{H}, E, X, \bar{x}, \lambda, \lambda)$ , onde  $\mathbf{H} = \{H_1, \dots, H_n\}$ .

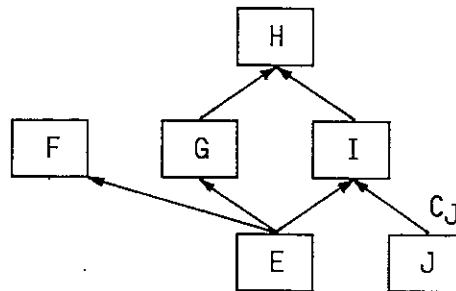
Considere agora a operação (interna)  $\text{classifique}^*(e, \mathbf{H}, E, X, \bar{x}, GP, SP)$ , onde  $\mathbf{H}$ ,  $E$ ,  $X$ ,  $\bar{x}$  são como para a operação de classificação,  $e$  é uma entidade nos conjuntos associados com os esquemas em  $\mathbf{H} \cup \text{gen}(\mathbf{H})$  e os dois últimos argumentos são usados somente para evitar trabalho redundante nas chamadas recursivas. A semântica de  $\text{classifique}^*$  é a seguinte ( $\lambda$  denota um valor indefinido):

- 1) para cada  $G$  tal que  $E$  é uma especialização de  $G$  e  $G \neq GP$  faça
  - a) se  $G \notin \mathbf{H} \cup \text{entre}(E, \mathbf{H}) \cup \text{gen}(\mathbf{H})$  mas  $e$  não está no conjunto correntemente associado com  $G$ ,  
então rejeite a operação;
  - b) se  $G \in \text{entre}(E, \mathbf{H})$  então  
se  $e$  existe no conjunto correntemente associado com  $G$ ,  
então  
se os valores dos atributos nativos de  $G$  para  $e$  não concordam com aqueles em  $\bar{x}$ ,  
então rejeite a operação;  
senão não faça nada;
  - senão  
execute  $\text{classifique}^*(e, \mathbf{G}, G, Y, \bar{y}, \lambda, E)$ , onde  $\mathbf{G}$  é a interseção de  $\mathbf{H}$  e  $\text{gen}(G)$ ,  $Y$  é a restrição de  $X$  para os atributos de  $G$  e  $\bar{y}$  é a restrição de  $\bar{x}$  para os atributos em  $Y$ ;
- 2) para a entidade  $e$ , faça os valores dos atributos nativos e dos atributos herdados dos esquemas entre  $E$  e  $\mathbf{H}$  serem dados por  $\bar{x}$  e faça os valores de todos os outros atributos de  $e$  serem aqueles que  $e$  já possui (o passo anterior garante isso);
  - a) se qualquer um dos seguintes testes falha, rejeite a operação:
    - i) teste se  $e$  satisfaz todas as assertivas de  $E$ ;
    - ii) se  $E$  é uma qualificação especializada de algum esquema de entidade  $G$  com qualificação  $C_G$ , teste se  $e$  satisfaz  $C_G$ ;
    - iii) teste se cada entrada em  $\bar{x}$  é do tipo apropriado (inclusive se o tipo admite valores nulos);
    - iv) teste se os valores dos atributos de  $e$  introduzirão qualquer valor de chave duplicado para as chaves nativas de  $E$ ;
    - v) teste se a inclusão de  $e$  em  $E$  violará qualquer condição imposta por uma declaração de exclusão mútua ou pela declaração de especialização com as palavras chaves totalidade ou exclusão mútua envolvendo  $E$ ;
  - b) inclua  $e$  no conjunto correntemente associado com  $E$ , com os valores dos atributos nativos de  $E$  dados pela lista  $\bar{x}$ .
- 3) para cada  $F$  tal que  $F$  é uma especialização qualificada de  $E$  com qualificação  $C_F$  e  $F \neq SP$  faça

se  $e$  satisfaz  $C_F$   
então

execute  $\text{classifique}^*(e, H, F, Z, \bar{z}, E, \lambda)$ , onde  $Z$  é uma lista com os atributos nativos de  $F$  concatenados com  $X$  e  $\bar{z}$  concorda com  $\bar{x}$  nos atributos de  $X$  e  $\bar{z}$  é igual a nulo para os atributos nativos de  $F$ .

Considere, por exemplo, o seguinte grafo de especialização:



Suponha que  $e$  está no conjunto correntemente associado com  $H$ , mas  $e$  não está em  $G$ ,  $I$  ou  $E$ . Então, para classificar  $e$  em  $E$ , o usuário deve fornecer valores para todos os atributos nativos de  $E$  e para todos os atributos herdados de  $G$  e  $I$ . A classificação terá sucesso se, entre outros fatores,  $e$  já pertencer ao conjunto correntemente associado com  $F$ . Note que  $e$  será inserido em  $G$  e  $I$ . Além disso, se  $e$  satisfizer  $C_J$ ,  $e$  também será inserido em  $J$ ; os valores dos atributos nativos de  $J$  serão instanciados com nulos, se o tipo do domínio permitir, caso contrário a inclusão de  $e$  em  $J$  será rejeitada e toda a classificação falhará.

Considere agora uma *operação de inclusão* da forma:

inclua em  $E$  com  $X=\bar{x}$

onde  $E$  é o nome de um esquema de entidades,  $X$  é a lista de todos os atributos, herdados ou não de  $E$  e  $\bar{x}$  é a lista de valores para os atributos em  $X$ . A semântica que daremos para este comando força a inclusão de uma nova entidade em  $E$ , em todas as generalizações transitivas de  $E$  e em todas as especializações transitivas qualificadas de  $E$ , se necessário. Note, entretanto, que  $\bar{x}$  não fornece valores para os atributos nativos das especializações transitivas qualificadas de  $E$ , que são então instanciadas com nulos, se possível. Observe também, que o usuário que executou esta operação deve saber exatamente como computar os atributos herdados de  $E$  e deve estar consciente de todas as propagações.

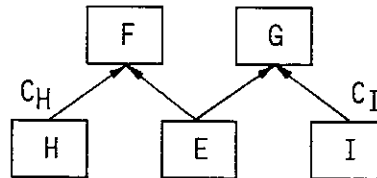
Mais precisamente, a semântica da operação, no contexto de um dado grafo de especialização, é:

- 1) crie uma nova entidade  $e$ ;
- 2) execute  $\text{classifique}^*(e, \{T\}, E, X, \bar{x}, \lambda, \lambda)$ ;

Enfatizamos que o primeiro passo cria uma nova entidade, isto é, uma entidade que não existe no estado corrente, uma ação que está totalmente de acordo com o conceito semântico de que as entidades tem uma existência independente. O segundo passo

inclui então a nova entidade em E e em todas as generalizações transitivas de E pois, pela definição do "topo" T,  $entre(E, \{T\}) = gen(E)$ .

Considere, por exemplo, o seguinte grafo de especialização:



Suponha que todos os atributos nativos de H e I admitem valores nulos. Então, para incluir uma nova entidade em E, o usuário especifica os valores para os atributos nativos de E e para os atributos de E herdados de F e G. A execução da operação criará primeiro uma nova entidade digamos,  $e$ , e a incluirá em E. Depois,  $e$  será incluída em F e G. Se  $e$  satisfaz  $C_H$ , então  $e$  também será incluída em H com todos os atributos nativos instanciados com nulos e, se  $e$  satisfizer  $C_I$ , então  $e$  será da mesma maneira incluída em I.

Vamos considerar agora, resumidamente, uma outra forma interessante de inclusão. Como motivação, suponha que E especialize F e G e que existam entidades  $f$  e  $g$  nos conjuntos correntemente associados com F e G, respectivamente. Suponha que (no mundo real) o usuário descubra que  $f$  e  $g$  são a mesma entidade e que ele deseje classificá-la em E. Propomos então uma operação que identifica  $f$  e  $g$  e inclui a entidade identificada em E. Note que não podemos, neste caso, usar a operação classifique porque (no banco de dados)  $f$  e  $g$  são entidades distintas.

A sintaxe geral da *operação de identificação* é:

identifica de  $H_1$  onde  $Q_1$ , ..., de  $H_n$  onde  $Q_n$   
[em E fazendo  $X = \bar{x}$ ]

onde, para cada  $i \in [1, n]$ , com  $n > 1$ ,  $H_i$  é um esquema de entidades,  $Q_i$  é uma seleção que define um subconjunto de um conjunto de entidades correntemente associada com  $H_i$ , e, se especificado, E é uma especialização transitiva de  $H_1, \dots, H_n$ , X é a lista de atributos nativos de E ou herdados dos esquemas entre  $H_1, \dots, H_n$  e E e  $\bar{x}$  é a lista de valores para os atributos em X.

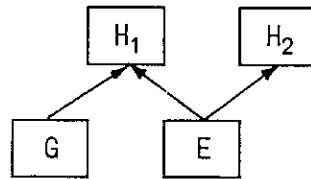
A semântica desta operação é a seguinte:

- 1) para cada  $i \in [1, n]$ , seja  $I_i$  o conjunto de todas as entidades, no conjunto correntemente associado com  $H_i$ , que satisfazem  $Q_i$ . Se  $I_i$  não contém exatamente um único elemento, então rejeite a operação;
- 2) para cada  $i \in [1, n]$ , crie uma nova entidade  $e$  e substitua, no estado corrente, a única entidade em  $I_i$  por  $e$ , ajustando as funções de atributo e os conjuntos de relacionamentos;
- 3) execute  $classifique^*(e, \{H_1, \dots, H_n\}, E, X, \bar{x}, \lambda, \lambda)$

Nota: o passo (3) somente é executado se E foi especificado.

O leitor deve neste ponto analisar cuidadosamente a diferença entre as operações de classificação, inclusão e identificação.

Podemos definir uma operação inversa para a operação de identificação como uma forma seletiva de remoção com redefinição de entidade. Por exemplo, considere o seguinte grafo de especialização:



Suponha que uma entidade  $e$  está nos conjuntos correntemente associados com  $H_1$ ,  $H_2$  e  $E$ . Separar  $e$  de  $E$  com relação a  $H_1$  e  $H_2$  significa remover  $e$  de  $E$  e substituir  $e$  por uma nova entidade, digamos,  $e_i$  em  $H_i$  e em todas as especializações transitivas e nas generalizações de  $H_i$ , para  $i=1,2$ . Se  $H_1$  e  $H_2$  tem uma generalização transitiva em comum, digamos  $H$ , então a substituição de  $e$  não será feita de modo a evitar a criação de duas entidades em  $H$  com o mesmo valor de chave.

A sintaxe para a operação de separação é:

separe de  $E$  onde  $Q$  com-relacao-a  $H_1, \dots, H_n$

onde,  $E$  é um esquema de entidades,  $Q$  é uma seleção que define um subconjunto do conjunto de entidades correntemente associada a  $E$  e  $H_1, \dots, H_n$  são generalizações transitivas de  $E$ .

A semântica deste comando é:

- 1) seja  $D$  o conjunto de entidades, no conjunto correntemente associado a  $E$ , que satisfaz  $Q$ ;
- 2) para cada generalização transitiva  $F$  de  $E$  tal que  $F$  é uma especialização de algum  $H_i$ , para  $i \in [1, n]$  faça  
execute  $remove^*(F, D)$ ;
- 3) seja  $H = \{H_1, \dots, H_n\}$ ;
  - a) particione  $H$  em  $P_1, \dots, P_k$  tal que todos os esquemas em  $P_i$  tem uma generalização transitiva em comum e nenhum esquema em  $P_i$  tem uma generalização transitiva em comum com um esquema em  $P_j$ , para  $i, j \in [1, k]$  com  $i \neq j$ ;
  - b) para cada  $i \in [1, k]$ , para cada entidade  $e$  em  $D$  e no conjunto correntemente associado com algum esquema em  $P_i$ , faça  
crie uma nova entidade  $e'$  e substitua  $e$  por  $e'$ , ajustando as funções de atributo e os conjuntos de relacionamentos.

Por fim, observamos que existem outras operações úteis sobre conjuntos de entidades organizados em grafos de especialização. Podemos citar como exemplos, operações para mover ou copiar uma entidade de um conjunto para outro.

## 5. CONCLUSÕES

Os construtores apresentados na seção 2 formalizam e estendem as abstrações de generalização e subconjunto dentro do contexto do modelo entidade-relacionamento, enquanto as operações da seção 4 oferecem alternativas interessantes para o desenvolvimento de uma linguagem de manipulação de dados que leva em consideração tais abstrações.

Qualquer implementador de um sistema de gerência de banco de dados baseado no modelo entidade-relacionamento deve, em um primeiro estágio, analisar cuidadosamente a semântica dos construtores e das operações para avaliar a sua complexidade. Em um segundo estágio, o implementador deverá, talvez, restringir os construtores e as operações de forma a reduzir o custo de implementação para um nível desejado.

## BIBLIOGRAFIA

- [1] A. Borgida, "Features of languages for the development of Information Systems at the Conceptual level", IEEE Software 2(1), January 1985, 63-73.
- [2] U. Bussolati, S. Ceri, V. De Antonellis and B. Zonta, Views Conceptual Design, in S. Ceri (ed.) Methodology and Tools for Data Base Design, North-Holland, 1983.
- [3] S. Ceri (ed.), Methodology and tools for data base design, North-Holland, 1983.
- [4] P. P. Chen - The entity-relationship model: toward a unified view of data - ACM TODS, 1, 1 (1976) 9-36.
- [5] A. Dogac, P.P. Chen, Entity-Relationship Model in the ANSI/SPARC Framework, in Entity-Relationship Approach to System Analysis and Design, P.P. Chen (ed.), North-Holland (1981), 361-378.
- [6] R. Elmasri, J. Weeldreyer, A. Heuner, The Category Concept: An extension to the entity-relationship model, in Data and Knowledge Engineering 1, North-Holland, 1985, 75-116.
- [7] H.F. Korth and A. Silberschatz, *Database System Concepts*, McGraw-Hill Book Co. (1986).
- [8] W. Kozaczynski, L. Lillien, An Extended Entity-Relationship (E<sup>2</sup>R) Database Specification and its Automatic Verification and Translation into the Logical Relational Design, Proc. of the Sixth Int. Conf. on Entity-Relationship Approach, New York (Nov. 1987), 497-513.
- [9] M. Lenzerini, G. Santucci, Cardinality constraints in the E-R model, Entity-Relationship Approach to Software Engineering, North-Holland, 1983.
- [10] P. Scheuermann, G. Schiffner, H. Weber, Abstraction capabilities and invariant properties modelling within the Entity-Relationship approach, P.P. Chen (ed.) Entity-Relationship approach to System Analysis and Design, North-Holland, 1980.
- [11] T.J. Teorey, D. Yang, J.P. Fry, A Logical Design Methodology for Relational Databases Using the Extended Entity-Relationship Model, ACM Computing Survey, 18, 2 (June 1986), 197-222.