

TRIVIALIZAÇÃO DE DEPENDÊNCIAS DE INCLUSÃO EM ESQUEMAS CONCEITUAIS RELACIONAIS

Anelise P. Braga, Marco A. Casanova, Luiz Tucheran

Centro Científico Rio - IBM Brasil
Caixa Postal 4624
22.071, Rio de Janeiro, RJ

SUMÁRIO

Este trabalho apresenta um método para otimização de esquemas conceituais relacionais que elimina dependências de inclusão através de transformações estruturais, sem mudar a semântica do esquema. O método pode ser interpretado como um processo de normalização para dependências de inclusão e aplica-se tanto ao modelo relacional tradicional quanto ao modelo relacional NF².

1. INTRODUÇÃO

Quando um projetista de banco de dados especifica um esquema conceitual relacional, normalmente ele inclui um conjunto de restrições de integridade para capturar quando um estado do banco de dados reflete corretamente o mundo real. Uma classe particularmente importante de restrições são as chamadas dependências de inclusão [CFP]. No modelo relacional tradicional, todas as dependências de inclusão devem ser explicitamente declaradas. Porém, no modelo relacional NF² [JS], uma classe destas dependências pode ser implicitamente capturada através da estrutura de aninhamento de tabelas.

Este trabalho apresenta um método para otimização de esquemas relacionais, no modelo tradicional ou no modelo NF², que elimina dependências de inclusão através de transformações estruturais, sem mudar a semântica do esquema. O método pode ser interpretado como um algoritmo de normalização para dependências de inclusão que formaliza e estende certas transformações estruturais comumente aplicadas a esquemas relacionais representando diagramas entidade-relacionamento. Um tratamento mais detalhado desta representação encontra-se em [Br].

A organização do trabalho é a seguinte. A seção 2 define conceitos preliminares. A seção 3 descreve o método de otimização assumindo o modelo relacional tradicional. A seção 4 estende o método para o modelo relacional NF². Por fim, a seção 5 contém conclusões.

2. PRELIMINARES

Recordaremos nesta seção alguns conceitos e introduziremos uma notação básica para o modelo relacional, com o qual assumimos familiaridade.

Um *esquema de relação* é uma expressão da forma $R[A_1 \dots A_n]$ onde R é o *nome* e A_1, \dots, A_n é a lista de nomes de *atributos* do esquema, tirados de um dado conjunto de identificadores.

Frequentemente usaremos o nome do esquema em lugar do próprio esquema e o termo "atributo" em lugar de "nome do atributo".

Seja E um conjunto de esquemas de relação com nomes distintos.

Uma *definição de visão* sobre E é uma tripla $(V[A_1...A_n], Q, T)$ onde

- $V[A_1...A_n]$ é um esquema de relação cujo nome é distinto dos nomes dos esquemas em E ;
- Q é uma expressão relacional n -ária sobre E ;
- T é uma especificação de traduções corretas das operações sobre V em operações sobre os esquemas de E [FC].

Dizemos ainda que V é o *nome* da visão e que Q é a *expressão de definição* da visão. Quando a especificação de T for desnecessária, denotaremos uma definição de visão $(V[A_1...A_n], Q, T)$ por $V[A_1...A_n] = Q$.

Seja V um conjunto de visões sobre E . Suponha que os esquemas contidos em E e os definidos em V tenham nomes distintos. Um *estado* para E é uma função σ que associa a cada nome R de um esquema $R[A_1...A_n]$ em E , uma relação n -ária $\sigma(R)$ e a cada nome V de uma visão em V , a relação $\sigma(V)$ definida pelo valor da expressão de definição de V em σ . Denotaremos ainda o valor de uma expressão Q em σ por $\sigma(Q)$.

Utilizaremos no que se segue o símbolo λ para denotar indistintamente o valor nulo ou tuplas de valores nulos de comprimento arbitrário.

As classes de *restrições de integridade* consideradas neste trabalho serão a classe das dependências de nulidade, a classe das chaves e a classe das dependências de inclusão. Uma *dependência de nulidade* sobre E é uma expressão da forma " $P: X$ not null" onde P ou é o próprio nome R ou é uma restrição (no sentido usual da álgebra relacional) sobre um esquema em E de nome R e X é uma seqüência de atributos distintos de R . Dizemos ainda que cada atributo A em X *não admite valores nulos* em P . Um estado σ para E *satisfaz* " $P: X$ not null" se e somente se $t_A \neq \lambda$, para toda tupla t em $\sigma(P)$, para cada atributo A em X .

Uma *chave* sobre E é uma expressão da forma $R:K$ onde R é o nome de um esquema de relação em E e K é uma seqüência de atributos distintos de R que não admitem valores nulos em R . Um estado σ para E *satisfaz* $R:K$ se e somente se, para todo par de tuplas t, u em $\sigma(R)$, se $t_K = u_K$ então $t = u$.

Uma *dependência de inclusão* sobre E é uma expressão da forma $P_1[X_1] \subseteq P_2[X_2]$ onde, para $i = 1, 2$, P_i ou é o próprio nome R_i ou é uma restrição sobre um esquema em E de nome R_i , X_2 é uma chave de R_2 e X_1 é uma lista de atributos distintos de R_1 de mesmo comprimento que X_2 . Dizemos ainda que $P_1[X_1] \subseteq P_2[X_2]$ é *de* R_1 *para* R_2 e que é *simples* em R_i se e somente se P_i for o próprio nome R_i . Um estado σ para E *satisfaz* $P_1[X_1] \subseteq P_2[X_2]$ se e somente se $\sigma(P_1[X_1]) \subseteq \sigma(P_2[X_2])$.

Note que esta definição de dependência de inclusão difere da definição usual [CFP], sendo no entanto adequada para o desenvolvimento da seção 3. Em particular, a definição original não exige que X_2 seja chave de R_2 mas, por outro lado, não permite que P_i seja uma restrição.

Uma dependência de inclusão $P_1[X_1] \subseteq P_2[X_2]$ de R_1 para R_2 pode ser adjetivada ainda com uma *opção de remoção* para R_2 tirada do conjunto:

- (1) remoção de R_2 bloqueia imediatamente
- (2) remoção de R_2 bloqueia
- (3) remoção de R_2 propaga imediatamente
- (4) remoção de R_2 propaga

onde as opções (3) e (4) só são permitidas se a dependência for simples em R_1 , e uma *opção de inserção* para R_1 tirada do conjunto:

- (5) inserção em R_1 bloqueia imediatamente
- (6) inserção em R_1 bloqueia
- (7) inserção em R_1 propaga imediatamente
- (8) inserção em R_1 propaga

onde as opções (7) e (8) só são permitidas se a dependência for simples em R_2 e todos os atributos de R_2 , excluindo-se os pertencentes a X_2 , admitirem valores nulos.

As restrições impostas às opções (3), (4), (7) e (8) apenas evitam ambigüidades no processo de propagação. Porém, para contornar tais restrições seria necessário definir opções mais elaboradas e modificar o processo de otimização, descrito na seção 3, para acomodar as novas opções.

Esta adjetivação altera o comportamento das operações da forma usual [Da,CFT]. Ilustraremos este ponto considerando dois casos de adjetivação para uma dependência de inclusão da forma $P_1[X_1] \subseteq P_2[X_2]$, onde P_1 e P_2 não são nomes de relação.

Para a opção (1), o teste de bloqueio associado rejeita a remoção de uma tupla t de R_2 em um estado σ se t satisfizer P_2 e $t_{X_2} \in \sigma(P_1[X_1])$. Note que, como X_2 é a chave de R_2 , não existe nenhuma outra tupla $t' \in \sigma(R_2)$ tal que $t'_{X_2} = t_{X_2}$. Para a opção (2), o teste de bloqueio associado aborta a transação se o estado no momento do comprometimento ("commit") dos dados for tal que $\sigma(P_1[X_1]) \not\subseteq \sigma(P_2[X_2])$. As opções (1) e (2) não precisam levar em consideração atualizações pois não permitimos tais operações sobre chaves, logo sobre X_2 .

Como último exemplo, à opção (7) está associado um gatilho que, para cada inserção de uma tupla t em R_1 em um estado σ , se t satisfizer P_1 e $t_{X_1} \notin \sigma(R_2[X_2])$, insere uma tupla u em R_2 tal que $u_{X_2} = t_{X_1}$ e $u_A = \lambda$, para todo atributo A de R_2 que não pertence a X_2 . Note que P_2 neste caso é o próprio nome R_2 e que A admite valores nulos, por suposição. Ação semelhante ocorre quando uma atualização modificar os valores dos atributos em X_1 de uma tupla t de tal forma que t passe a satisfizer P_1 e $t_{X_1} \notin \sigma(R_2[X_2])$.

Um *esquema conceitual relacional* é um par $S = (E, I)$ onde E é um conjunto de esquemas de relação com nomes distintos e I é um conjunto de restrições de integridade sobre E contendo exatamente uma chave para cada esquema em E .

O método de trivialização trabalha com esquemas conceituais com mais estrutura, definidos da seguinte forma. Um *esquema conceitual estendido* é um par $SS = (SI, SV)$ onde

- $SI = (EB, DV, IA)$, a *componente interna* de SS , é tal que

- EB é um conjunto de esquemas de relação com nomes distintos, chamados de *esquemas básicos* de SS ;
- DV é um conjunto de definição de visões sobre EB com nomes distintos entre si;
- IA é um conjunto de restrições de integridade sobre os esquemas em EB , chamadas de *restrições ativas* de SS , contendo exatamente uma chave para cada esquema em EB .
- $SV=(E,I)$, a *componente visível* de SS , é um esquema conceitual relacional tal que $E \subseteq EB \cup e(DV)$, onde $e(DV)$ é o conjunto dos esquemas definidos em visões em DV ;

Um esquema conceitual estendido $SS=((EB,DV,IA),(E,I))$ está *correto* se e somente se:

- todo estado consistente de (EB,IA) induz, via DV , um estado consistente de (E,I) ;
- as assertivas e gatilhos associados às dependências de inclusão em IA corretamente implementam, novamente via DV , as assertivas e gatilhos associados às dependências em I .

3. O MÉTODO DE TRIVIALIZAÇÃO PARA O MODELO RELACIONAL TRADICIONAL

Esta seção introduz um método de otimização para esquemas conceituais relacionais, no modelo tradicional, que minimiza o número de dependências de inclusão. O método, chamado de *trivialização*, generaliza e rigoriza uma otimização, habitualmente feita na representação de um diagrama entidade-relacionamento por um esquema conceitual relacional, que colapsa em uma mesma tabela duas tabelas possuindo a mesma chave.

Seja SS um esquema estendido. Uma dependência de inclusão d em SS é *trivializável* se e somente se d for da forma (a) $S[L] \subseteq R[K]$ ou da forma (b) $S[L] \subseteq P[K]$ onde,

- 1) S é um esquema básico de SS com chave L e que possui um atributo N que não ocorre em L e que não admite valores nulos;
- 2) R é um esquema básico de SS com chave K e P é uma expressão relacional sobre R , no caso (b) apenas;
- 3) a adjetivação da dependência d indica que remoções de R bloqueiam ou propagam imediatamente para S e que inserções em S bloqueiam ou propagam imediatamente para R ;
- 4) não há uma dependência de inclusão ativa em SS de T para S tal que as inserções em T propaguem, imediatamente ou não, ou uma dependência de inclusão ativa em SS de S para T tal que as remoções de T propaguem, imediatamente ou não.

As condições (1) e (2) permitem, via as chaves K e L , colapsar os esquemas S e R em um único esquema G . Cada tupla t de G corresponde sempre a uma tupla de R e, se $t_N \neq \lambda$, também uma tupla de S . A condição (3) clarifica quais são as opções de inserção e remoção compatíveis com a criação de G . Finalmente, a condição (4) reflete as restrições sobre as opções de remoção e inserção impostas na seção 2. Sem ela, a criação de G também poderia causar problemas no passo (7) do processo de trivialização, descrito mais adiante.

O algoritmo de trivialização, como o nome indica, trivializa sucessivamente dependências de inclusão de um esquema relacional $S=(E,I)$, dado como entrada, produzindo como saída um esquema estendido $SS_n=((EB_n,DV_n,IA_n),(E,I))$ que preserva na sua componente visível o esquema original. Ou seja, a otimização é transparente aos usuários pois estes continuam a ver o esquema original $S=(E,I)$.

Algoritmo de Trivialização

faça $SS_0 := ((E, \emptyset, I), (E, I))$

faça $i := 0$;

enquanto houver uma dependência de inclusão trivializável em SS_i faça

selecione não-deterministicamente uma dependência de inclusão trivializável d_i de SS_i ;

trivialize d_i , como descrito abaixo, produzindo um novo esquema SS_{i+1} ;

faça $i := i + 1$;

O algoritmo de trivialização produz um esquema conceitual estendido final que é correto. Este resultado segue por indução sobre o número de iterações do algoritmo observando que, a cada passo, a dependência de inclusão trivializada torna-se consequência das restantes, por força da definição das novas visões.

O processo de trivialização de uma dependência d_i é refinado da seguinte forma. Suponha que d_i seja da forma $S[L] \subseteq R[K]$ ou da forma $S[L] \subseteq P[K]$, onde P é uma expressão relacional sobre R e $R[K, X]$ e $S[L, Y]$ são esquemas básicos de SS_i . Suponha que N seja um atributo em Y que não admite valores nulos.

O novo esquema estendido SS_{i+1} é obtido de SS_i através das seguintes transformações:

- 1) Remova d_i do conjunto das restrições ativas;
- 2) Acrescente $G[K, X, Y']$ ao conjunto dos esquemas básicos, onde Y' é uma renomeação dos atributos em Y para evitar conflitos com os atributos de K e X (denotaremos por A' o atributo de Y' correspondente a um atributo A de Y). Defina K como a chave de G .
- 3) Remova $R[K, X]$ do conjunto dos esquemas básicos e transforme-o em visão acrescentando ao conjunto de definições de visões a tripla $(R[K, X], G[K, X], T)$, onde T define a tradução das operações sobre R para G através da seguinte tabela:

Operação	Tradução	Obs.
insert into R(t)	insert into G(K=t _K , X=t _X , Y'=λ)	
delete from R where Q	delete from G where Q	(1)
delete from R where Q	delete from G where Q and N'=λ	(2,3)
update R set X=t where Q	update G set X=t where Q	

Obs.:

- (1) tradução quando remoções de R propagam imediatamente
 - (2) tradução quando remoções de R bloqueiam imediatamente
 - (3) $N' = \lambda$ indica que a tupla de G não codifica uma tupla de S .
- 4) Remova $S[L, Y]$ do conjunto dos esquemas básicos e transforme-o em uma visão acrescentando ao conjunto de visões a tripla $(S[L, Y], G[N' \neq \lambda][K, Y'], U)$ onde U é definido como:

Operação	Tradução	Obs.
insert into S(L=u _L , Y=u _Y)	insert into G(K=u _L , Y'=u _Y , X=λ) if insert fails then	(1)

	update G set $Y' = u_Y$	
	where $K = u_L$ and $N' = \lambda$	
insert into S($L = u_L, Y = u_Y$)	update G set $Y' = u_Y$	(2)
	where $K = u_L$ and $N' = \lambda$	
insert into S($L = u_L, Y = u_Y$)	update G set $Y' = u_Y$	(3)
	where $K = u_L$ and $N' = \lambda$ and $Z \neq \lambda$	
delete from S where Q	update G set $Y' = \lambda$ where Q'	(4)
update S set $Y = u_Y$ where Q	update G set $Y' = u_Y$	(5)
	where Q' and $N' \neq \lambda$	

Obs.:

- (1) tradução quando inserções em S propagam imediatamente
- (2) tradução quando inserções em S bloqueiam imediatamente e a dependência é simples em R;
- (3) tradução quando inserções em S bloqueiam imediatamente e a dependência é da forma $S[L] \subseteq R[Z \neq \lambda][K]$;
- (4) Q' é uma renomeação de Q para acompanhar a renomeação de Y' a partir de Y
- (5) o termo $N' \neq \lambda$ evita que a atualização se transforme em uma inserção
- 5) Em cada expressão de definição de visão, exceto as de R e S, substitua R e S pelas suas expressões de definição, simplificando o resultado se possível.
- 6) Em cada tradução de operação de visão, exceto aquelas associadas a R e S, aplique as traduções especificadas em T e U a cada operação sobre R e S.
- 7) Em cada restrição ativa, substitua R e S pelas suas expressões de definição, simplificando o resultado.
- 8) Descarte as definições de visões cujos esquemas não pertencem à parte visível.

Note que as dependências sobre R e S, que agora são visões sobre G, passam a ser consequência de dependências sobre G. De fato, definindo K como chave de G no passo (2), garantimos que K e L são chaves de R e S e, através da transformação genérica definida no passo (7), mapeamos automaticamente todas as dependências de nulidade e de inclusão envolvendo R e S para dependências equivalentes sobre G.

Este processo pode ser otimizado se, para $i=0, \dots, n-1$, existir uma dependência d_i da forma $R_i[K_i] \subseteq R_{i+1}[K_{i+1}]$, onde a soma é módulo n, tal que:

- 1) $R_i[K_i, X_i]$ é um esquema básico com chave K_i tal que todo atributo em X_i admite valores nulos;
- 2) a adjetivação de d_i indica que remoções de R_{i+1} propagam imediatamente para R_i e inserções em R_i propagam imediatamente para R_{i+1} .

Por simplicidade, assumiremos no que se segue que $K = K_i$ e que X_i e X_j não têm atributos em comum, para todo $i, j \in [0, n-1]$, com $i \neq j$. Denotaremos ainda a lista de atributos $X_1 \dots X_{i-1} X_{i+1} \dots X_{n-1}$ por \bar{X}_i .

Estas dependências podem ser simultaneamente trivializadas da seguinte forma.

- 1) Remova cada dependência d_i do conjunto das restrições ativas.

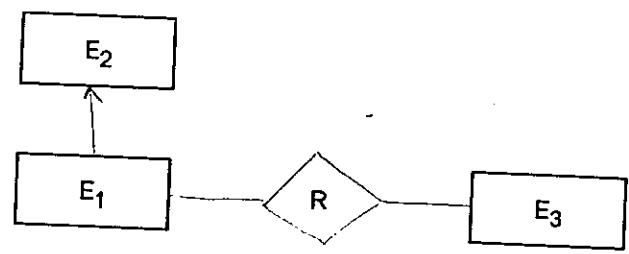
- 2) Acrescente $G[K, X_0, \dots, X_{n-1}]$ ao conjunto dos esquemas básicos, tendo K como chave.
- 3) Remova cada esquema $R_i[K, X_i]$ do conjunto dos esquemas básicos e transforme-o em uma visão acrescentando ao conjunto de definição de visões a tripla $(R_i[K, X_i], G[K, X_i], T_i)$, onde T_i é definido por:

Operação	Tradução	Obs.
insert into $R_i(t)$	insert into $G(K=t_{K_i}, X_i=t_{X_i}, \bar{X}_i=\lambda)$ if insert fails then	(1)
delete from R_i where Q	update G set $X_i=t_{X_i}$ where $K=t_{K_i}$	(2)
update R_i set $X_i=t_{X_i}$ where Q	delete from G where Q update G set $X_i=t_{X_i}$ where Q	

Obs.:

- (1) inserções em R_i propagam imediatamente para R_{i+1} , por suposição
 - (2) remoções de R_{i+1} propagam imediatamente para R_i , por suposição
- 4, 5, 6 e 7) Semelhantes aos passos 5, 6, 7 e 8 do processo de trivialização original.

Segue-se um exemplo do processamento do algoritmo de trivialização, que omite vários detalhes por simplicidade. Considere o diagrama entidade-relacionamento abaixo, onde E_1 é uma especialização de E_2 :



O mapeamento imediato deste diagrama para o modelo relacional tradicional, com uma população mínima de atributos, gerará o esquema conceitual relacional S_0 definido como:

Esquema Conceitual : S_0

Esquemas	Chave	Dep. Nulidade
$E_1[K_2, A_1]$	$E_1:K_2$	(n^0_1) $E_1: K_2, A_1$ not null
$E_2[K_2]$	$E_2:K_2$	(n^0_2) $E_2: K_2$ not null
$E_3[K_3]$	$E_3:K_3$	(n^0_3) $E_3: K_3$ not null
$R[K_2, K_3]$	$R:K_3$	(n^0_4) $R: K_2, K_3$ not null

Dependências de Inclusão

- (d^0_1) $E_1[K_2] \subseteq E_2[K_2]$: remoção de E_2 propaga imediatamente, inserção em E_1 bloqueia imediatamente

- (d^0_2) $R[K_2] \subseteq E_1[K_2]$: remoção de E_1 bloqueia imediatamente, inserção em R bloqueia imediatamente
- (d^0_3) $R[K_3] \subseteq E_3[K_3]$: remoção de E_3 propaga imediatamente, inserção em R bloqueia imediatamente

Resumidamente, o processamento do algoritmo produzirá a seguinte sucessão de esquemas estendidos:

- Esquema Estendido :** SS_1
- dep. trivializada :* d^0_3
- novo esquema :* $E_3R[K_2K_3]$, chave : $E_3R:K_3$
- novas visões :* $E_3[K_3] = E_3R[K_3]$
 $R[K_2K_3] = E_3R[K_2 \neq \lambda][K_2K_3]$
- dep. modificadas :*
- (n^1_3) $E_3R: K_3$ not null
- (n^1_4) $E_3R[K_2 \neq \lambda]: K_2, K_3$ not null (implicada por (n^1_3))
- (d^1_2) $E_3R[K_2 \neq \lambda][K_2] \subseteq E_1[K_2]$: remoção de E_1 bloqueia imediatamente, inserção em E_3R bloqueia imediatamente

- Esquema Estendido :** SS_2
- dep. trivializada :* d^0_1
- novo esquema :* $E_2E_1[K_2A_1]$, chave : $E_2E_1:K_2$
- novas visões :* $E_1[K_2A_1] = E_2E_1[A_1 \neq \lambda][K_2A_1]$
 $E_2[K_2] = E_2E_1[K_2]$
- dep. modificadas :*
- (n^2_1) $E_2E_1[A_1 \neq \lambda]: K_2, A_1$ not null (implicada por (n^2_2))
- (n^2_2) $E_2E_1: K_2$ not null
- (d^2_3) $E_3R[K_2 \neq \lambda][K_2] \subseteq E_2E_1[A_1 \neq \lambda][K_2]$: remoção de E_2E_1 bloqueia imediatamente, inserção em E_3R bloqueia imediatamente

Como d^2_3 não é trivializável, o algoritmo pára retornando o esquema estendido SS_2 , que contém (eliminando as dependências de nulidade redundantes):

Componente Interna:

- Esquemas Básicos :* $E_3R[K_2K_3]$, chave : $E_3R:K_3$
 $E_2E_1[K_2A_1]$, chave : $E_2E_1:K_2$
- Visões Definidas :* $E_1[K_2A_1] = E_2E_1[A_1 \neq \lambda][K_2A_1]$

$$E_2[K_2] = E_2E_1[K_2]$$

$$E_3[K_3] = E_3R[K_3]$$

$$R[K_2K_3] = E_3R[K_2 \neq \lambda][K_2K_3]$$

Dep. de Nulidade :

E_3R : K_3 not null

E_2E_1 : K_2 not null

Dep. de Inclusão :

$E_3R[K_2 \neq \lambda][K_2] \subseteq E_2E_1[A_1 \neq \lambda][K_2]$:
remoção de E_2E_1 bloqueia imediatamente,
inserção em E_3R bloqueia imediatamente

Componente Visível :

S_0 , o esquema conceitual original

Concluiremos esta seção reinterpretando o método de trivialização como um processo de normalização. Esta reinterpretação justifica-se pois tanto o método de trivialização quanto os processos tradicionais de normalização simplificam o tratamento de dependências por transformações estruturais.

Dizemos que um esquema conceitual relacional $S=(E,I)$ está em *forma normal para dependências de inclusão (FN/DI)* se e somente se I não possuir nenhuma dependência de inclusão trivializável.

O algoritmo de trivialização pode então ser reinterpretado como um processo para transformar um dado esquema $S=(E,I)$ em um esquema em FN/DI. De fato, suponha que o algoritmo produza como saída para S o esquema estendido $SS=((EB,DV,IA),(E,I))$. Podemos interpretar o esquema conceitual $SN=(EB,IA)$ como a normalização de S pois, pela própria construção de SS , o esquema SN está em FN/DI. Além disso, SN representa S no sentido de que S pode ser corretamente reconstruído a partir de SN utilizando-se as definições em DV , já que SS é um esquema estendido correto.

4. EXTENSÃO DO MÉTODO DE TRIVIALIZAÇÃO AO MODELO RELACIONAL NF2

Esta seção discute como estender o método de trivialização para o modelo NF². A discussão é bastante resumida e baseada em um exemplo. Em particular, sem definir explicitamente, usaremos a contra-partida para o modelo NF² dos conceitos introduzidos na seção 2.

Usaremos o símbolo μ para indicar a operação de desaninhamento de uma relação NF² [JS], subscrito com a parte a ser desaninhada. Por exemplo, dado um esquema de relação $R(A(B(C)^*))$ e um estado σ tal que

$$\sigma(R) = \{(a_1, \emptyset), (a_2, \{(b_1, \emptyset), (b_2, \{c_1, c_2\})\})\}$$

então $R' = \mu_{(B(C)^*)}(R)$ terá como valor em σ a relação

$$\sigma(R') = \{(a_2, b_1, \emptyset), (a_2, b_2, \{c_1, c_2\})\}$$

Note que μ automaticamente elimina as tuplas cuja componente desaninhada é o conjunto vazio. Portanto, o valor nulo λ em NF² não precisa ser interpretado como "ausência de valor", passando a ser apenas "valor desconhecido".

Seja SS um esquema conceitual estendido NF^2 . Uma dependência de inclusão d em SS é NF^2 -trivializável se e somente se d for da forma (a) $S[L] \subseteq R[K]$ ou da forma (b) $S[L] \subseteq P[K]$, onde

- 1) S é um esquema básico de SS ;
- 2,3,4) Idênticas às condições (2),(3) e (4) de trivialização no modelo tradicional.

Note que no modelo NF^2 não é necessário que L seja chave de S e que S possua um atributo N , que não ocorre em L , que não admita valores nulos. A trivialização de d aninha S em R , criando um único esquema G . Cada tupla t de G codifica sempre a tupla de R com chave t_K e o conjunto das tuplas u de S tais que $u_L = t_K$. Se este conjunto for vazio, a projeção de t sobre os atributos de S aninhados será simplesmente o conjunto vazio, e não uma tupla de nulos, como no modelo tradicional. Este fato torna desnecessária a condição (3) da definição de trivialização no modelo tradicional.

O algoritmo de trivialização estende-se ao modelo relacional NF^2 como sugere o seguinte exemplo. Considere o diagrama entidade-relacionamento abaixo (onde DEP^* indica que o atributo DEP é multi-valorado):



O mapeamento imediato deste diagrama para o modelo relacional NF^2 , com uma população mínima de atributos, geraria o esquema conceitual S_0 definido como:

Esquema Conceitual: S_0

<i>Esquemas</i>	<i>Chaves</i>	<i>Dep. Nulidade</i>
DEPT(D#)	DEPT:D#	(n^0_1) DEPT: D# not null
EMP(E# (DEP)*)	EMP:E#	(n^0_2) EMP: E# not null
TRAB(D# E#)	TRAB:E#	(n^0_3) TRAB: E# not null

Dependências de Inclusão

- (d^0_1) TRAB[D#] \subseteq DEPT[D#]: remoção de DEPT propaga imediatamente, inserção em TRAB bloqueia imediatamente
- (d^0_2) TRAB[E#] \subseteq EMP[E#]: remoção de EMP propaga imediatamente, inserção em TRAB propaga imediatamente
- (d^0_3) EMP[E#] \subseteq TRAB[E#]: remoção de TRAB propaga imediatamente, inserção em EMP propaga imediatamente

Resumidamente, o processamento do algoritmo produziria a seguinte sucessão de esquemas estendidos:

Esquema estendido: SS_1

dep. trivializadas: d^0_2 e d^0_3 , simultaneamente

novo esquema: EMP_TRAB(E# D# (DEP)*), chave: EMP_TRAB:E#

novas visões: EMP(E# (DEP)*) = EMP_TRAB[E# (DEP)*]
TRAB(D# E#) = EMP_TRAB[D# E#]

- dep. modificadas:*
- (n^1_2) EMP_TRAB: E# not null (captura n^0_2 e n^0_3)
 - (d^1_1) EMP_TRAB[D#] \subseteq DEPT[D#]: remoção de DEPT propaga imediatamente, inserção em EMP_TRAB bloqueia imediatamente

Esquema estendido: SS_2

dep. trivializada: d^1_1

novo esquema: DEPT_EMP_TRAB(D# (E# (DEP)*)*)
chave: DEPT_EMP_TRAB:D#

novas visões: DEPT(D#) = DEPT_EMP_TRAB[D#]
EMP_TRAB(E# D# (DEP)*) =
($\mu_{(E\#(DEP)^*)}$)DEPT_EMP_TRAB[E# D# (DEP)*]

visões modificadas: EMP(E# (DEP)*) =
($\mu_{(E\#(DEP)^*)}$)DEPT_EMP_TRAB[E# (DEP)*]
TRAB(D# E#) =

(μ(E#(DEP)*DEPT_EMP_TRAB)[D# E#]

dep. modificadas:

$\mu_{(E\#(DEP)^*)}$ DEPT_EMP_TRAB : E# D# (DEP)*
(oriunda da chave de EMP_TRAB)

(n^2_1) DEPT_EMP_TRAB: D# not null

(n^2_2) $\mu_{(E\#(DEP)^*)}$ DEPT_EMP_TRAB: E# not null

O algoritmo pára, retornando então SS_2 , sem a visão EMP_TRAB que é descartada. Observe que, na definição das visões, devemos desaninhar o esquema DEPT_EMP_TRAB sobre $(E\#(DEP)^*)$ para projetá-lo nos atributos apropriados.

Note que escolhemos a ordem de trivialização das dependências de tal forma que os esquemas de relação foram sucessivamente aninhados em um só. Em uma situação geral convém analisar com cuidado o benefício de aninhar um esquema S em outro quando existem dependências de inclusão envolvendo S que não serão trivializadas.

Da mesma forma que para o modelo tradicional, a definição de dependência trivializável induz uma forma normal para o modelo NF² e o algoritmo de trivialização traduz-se em um processo de normalização.

5. CONCLUSÕES

O método de trivialização, definido na seção 3, otimiza um esquema conceitual relacional na medida em que simplifica os testes necessários para manter as dependências de inclusão. Por possuir características semelhantes aos processos de normalização tradicionais - simplificação do tratamento de dependências por transformações estruturais - o método de trivialização pode ser reinterpretado como um processo de normalização baseado em dependências de inclusão.

A extensão do método para o modelo relacional NF², proposta na seção 4, leva mais adiante a eliminação de dependências de inclusão fazendo uso do aninhamento de relações permitido em NF². A forma normal introduzida pela definição de NF²-trivializável, por se basear em dependências de inclusão, difere completamente das anteriormente definidas [AMM,FG,OY,RKS], que são calcadas em MVDs e adaptações de FDs. Acreditamos porém que, quando comparadas com MVDs, as dependências de inclusão capturam de forma mais natural no modelo tradicional as relações de existência entre objetos expressas no modelo NF² pelo aninhamento das estruturas.

REFERÊNCIAS

- [AMM] H. Arisawa, K. Moriya, T. Miura, "Operations And The Properties On Non-First-Normal-Form Relational Databases", Proc. of the 9th International Conference on Very Large Data Bases, (1983), pp. 197-204.
- [Br] A.P. Braga, "Tradução de Esquemas Entidade-Relacionamento Estendido para Esquemas relacionais", IBM Centro Científico Rio, Relatório Técnico CCR 068, (Novembro 1988).
- [CFP] M.A. Casanova, R. Fagin, C. Papadimitriou, "Inclusion Dependencies and Their Interaction with Functional Dependencies", J. of Computer and System Sciences, Vol. 28, No. 1 (Fevereiro 1984), pp. 29-59.

- [CFT] M.A. Casanova, A.L. Furtado and L. Tucheran, "Enforcing Inclusion Dependencies and Referential Integrity", Proc. 14th International Conference on Very Large Data Bases, Los Angeles (Agosto 1988), pp. 13-25.
- [Da] C.J. Date, "Referential Integrity" in *Relational Database: Selected Writings* Addison-Wesley Publishing Company, (1986), (pp. 41-62).
- [FC] A.L. Furtado and M.A. Casanova, "Updating Relational Views", in *Query Processing in Database Systems*, W. Kim, D.S. Reiner and D.S. Batory (eds.), Springer Verlag (1985), pp. 127-142.
- [FG] P. Fisher, D. Van Gucht, "Determining When A Structure Is A Nested Relation", Proc. of the 11th International Conference on Very Large Data Bases, (1985), pp. 171-180.
- [JS] G. Jaeschke, H. J. Schek, "Remarks on the Algebra of Non First Normal Form Relations", Proc. of the ACM Symposium on Principles of Database Systems, ACM, New York, (1982), pp. 124-138.
- [OY] Z. M. Ozsoyoglu, L. Yuan, "A Normal Form for Nested Relations", Proc. of the 4th ACM SIGACT-SIGMOD Symposium on Principles of Database Systems, (Março 1985), pp. 251-260.
- [RKS] M. A. Roth, H. F. Korth, A. Silberschatz, "Theory of non-first-normal form relational databases", TR-84-36, Dept. of Computer Science, Univ. of Texas at Austin, (1984).