

## VISÃO PARCIAL: UM MECANISMO PARA ABSTRAÇÃO DE DADOS NO MODELO ENTIDADE RELACIONAMENTO

Marco A. Casanova\*  
Luiz Tucherman\*\*

### SUMÁRIO

Uma linguagem de definição de dados que segue o Modelo Entidade-Relacionamento é apresentada. Esta linguagem incorpora, entre várias facilidades, o conceito de visão parcial. Uma visão parcial é como uma visão tradicional, exceto que ela pode ter novos atributos armazenados. Visões parciais são definidas de forma a serem uma generalização de certos princípios de abstrações que auxiliam na organização do projeto conceitual de esquemas de bancos de dados tais como, hierarquias de generalização e agregação. Assim como para os mecanismos de abstração, o projetista do banco de dados pode usar as visões parciais na especificação do esquema conceitual onde antes somente eram permitidos esquemas de entidades e de relacionamentos.

### ABSTRACT

A data definition language, following the Entity-Relationship Model, is described. The language incorporates, among other features, the concept of partial view and statements that govern the propagation of operations. A partial view is just like a traditional view, except that it may have new stored attributes. Partial views are meant to be a rigorously defined generalization of certain abstraction principles that help organize the design of a conceptual schema, such as the generalization hierarchy and the aggregation abstraction. As with the usual abstraction mechanisms, the database designer may use partial views in the specification of a conceptual schema where only entity or relationship schemes were previously allowed.

\* 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, Estrada da Canoa, 3520, 22.610, 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, Estrada da Canoa, 3520, 22.610, Rio de Janeiro, RJ

## 1. INTRODUÇÃO

Entre as extensões mais interessantes feitas ao Modelo Entidade-Relacionamento estão aquelas que propõe princípios de abstração que auxiliam na organização do projeto de esquema conceitual. Classificamos neste grupo as hierarquias de generalização, a abstração de agregação (veja, por exemplo, [BCAZ,DC,SSW,TYF]) e o agrupamento de classes de entidades [KL]. Visões parciais são uma generalização destes mecanismos de abstração.

Uma visão parcial é semelhante a uma visão tradicional, exceto que ela pode ter novos atributos armazenados. Da mesma forma que os outros mecanismos de abstração, o projetista do banco de dados pode usar uma visão parcial na especificação de um esquema conceitual, onde antes somente era permitido esquemas de entidades e relacionamentos.

Este trabalho apresenta a especificação de uma linguagem de definição de dados seguindo o Modelo Entidade-Relacionamento a qual incorpora, entre outras facilidades, o conceito de visão parcial. Propõe também uma linguagem de consulta entidade-relacionamento que será usada para definir os mapeamentos das visões.

No contexto do Modelo ER, a definição de visões foi investigada em [LD,Li] e linguagens de consulta em [EL,MR,Ve]. Uma proposta para uma linguagem de definição de dados pode ser encontrada em [SKK].

Este trabalho está organizado da seguinte forma. A seção 2 apresenta informalmente a motivação para o conceito de visões parciais. A seção 3 define a linguagem básica de definição de dados do modelo entidade-relacionamento. A seção 4 introduz a linguagem de consulta e formaliza o conceito de visão parcial. Finalmente, a seção 5 contém as conclusões.

## 2. MOTIVAÇÃO

Esta seção introduz informalmente o conceito de visões parciais comparando-o com os conceitos familiares de hierarquia de generalização, agregação e agrupamento de classes de entidades.

Primeiramente definiremos uma visão  $V$  no Modelo ER como um mapeamento que transforma cada estado do banco de dados em:

- um conjunto  $v$  de objetos (que são, tanto entidades como relacionamentos);
- uma lista de funções de valoração para atributos que mapeia cada objeto em  $v$  nos valores dos seus atributos.

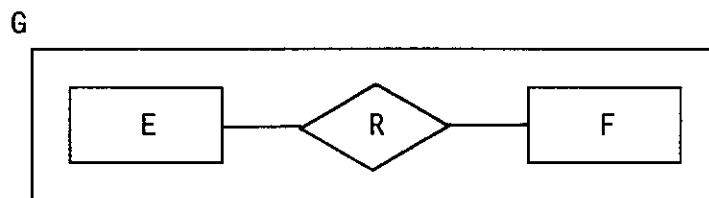
Isto é justificado porque os objetos no Modelo ER, ao contrário do Modelo Relacional, existem independentemente dos valores dos seus atributos.

Definimos uma *visão parcial*  $P$  como uma visão estendida com uma lista de *atributos nativos*. Dado um estado do banco de dados, o mapeamento associado a  $P$  leva este estado em um conjunto  $p$  de objetos e em um conjunto de funções de valoração para atributos. Entretanto, ao contrário das visões usuais, o estado do banco de dados deve especificar diretamente, para cada objeto em  $p$ , os valores dos atributos nativos de  $P$ . Assim, uma visão parcial pode ser equipada com novos atributos não derivados dos outros objetos do esquema (por isso o nome "visão parcial").

Em adição à introdução das visões parciais, estendemos o conceito de um esquema conceitual ER permitindo que estas visões parciais possam ser usadas no lugar das entidades e dos relacionamentos. Isto é, dentro da perspectiva da modelagem conceitual, sugerimos que o projetista do banco de dados possa especificar separadamente o esquema conceitual para um setor da empresa, resumir os dados em uma ou mais visões (parciais) e usar estas visões (parciais) no resto do projeto do esquema.

Vamos analisar agora, sob a perspectiva dos conceitos de visão parcial, as abstrações de dados que nos são mais familiares.

Consideraremos primeiramente a abstração de agregação. Tal abstração é usualmente empregada quando queremos elevar um conjunto de relacionamentos R ao 'status' de um conjunto de entidades, talvez por quisermos definir um outro conjunto de relacionamentos sobre R. Assim sendo, agregação evita considerarmos relacionamentos sobre relacionamentos. Esta abstração é usualmente introduzida através do seguinte diagrama:



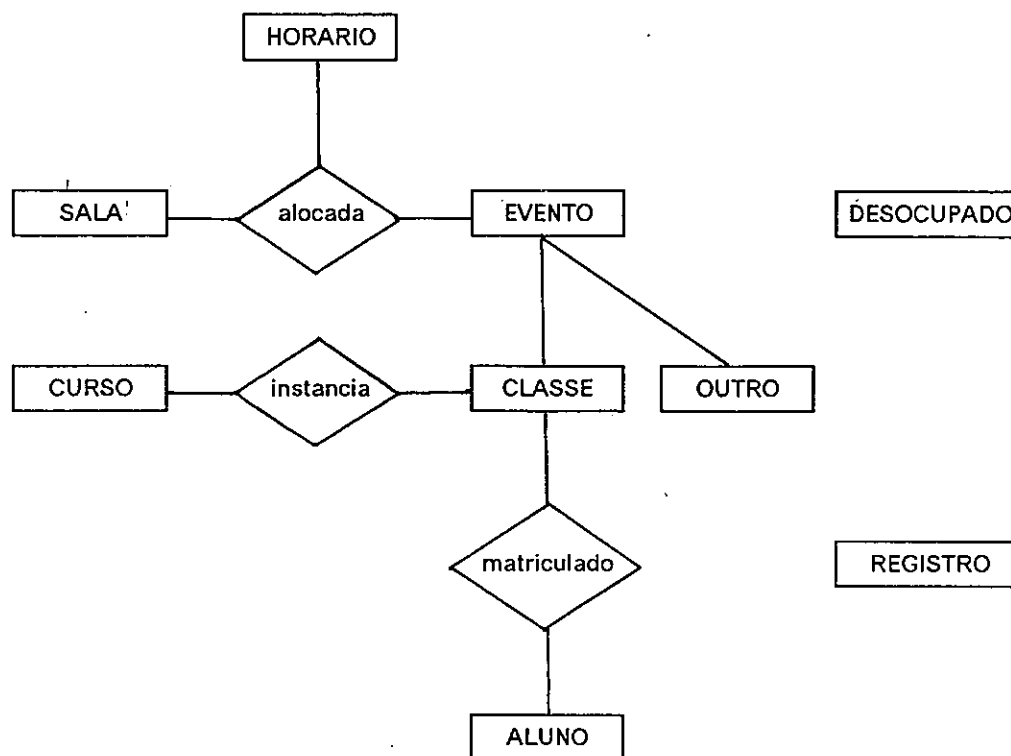
A semântica associada com o diagrama acima é que G é o relacionamento R, juntamente com os atributos de E e de F, elevado ao 'status' de um conjunto de entidades. Mas isso é equivalente a definirmos G como uma visão obtida, usando a terminologia relacional, pela junção natural de E, R e F projetado sobre R. Por esse motivo, é razoável estender esta idéia e usar visões comuns da mesma forma como as agregações são usadas. Entretanto, este passo não é livre de problemas, especialmente se considerarmos a propagação das operações. A seção 4 abordará este ponto.

Agrupamento de classes de entidades [KL] é outra abstração que pode ser facilmente explicada em termos do uso de visões. Por exemplo, o projetista do banco de dados pode definir, em um banco de dados de estoque, o conjunto de entidades ITEM como sendo a união dos conjuntos de entidades FITA, CASSETE, FOLHETO and 'FLOPPY'. Só que neste caso ITEM é somente uma visão.

Finalmente, consideremos o conceito de especialização. Na maioria dos casos, dado um conjunto de entidade E, o projetista do banco de dados pode definir novos conjuntos de entidades  $F_1, \dots, F_n$  como sendo subconjuntos de E que satisfazem as condições  $P_1, \dots, P_n$ , respectivamente. Então, da mesma forma,  $F_i$  é somente uma visão de E que herda todos os atributos de E. Entretanto, o projetista usualmente equipa  $F_i$  com um novo conjunto de atributos não derivados de E. Desse modo,  $F_i$  deve ser modelada através de uma visão  $F_i'$ , com os atributos derivados de E, e de um conjunto de entidades  $F_i''$ , com os novos atributos de  $F_i$ , de tal modo que  $F_i'$  e  $F_i''$  sempre denotem o mesmo conjunto de entidades. Preferimos considerar  $F_i'$  e  $F_i''$  como uma estratégia de implementação e tratar  $F_i$  como uma visão parcial que contém os atributos derivados de E juntamente com os novos atributos armazenados. Portanto, é neste ponto que o conceito de visão parcial se torna necessário. O conceito dual de generalização também pode ser facilmente reduzido ao conceito de visão parcial.

Vamos definir informalmente o fragmento de um esquema conceitual de um banco de dados acadêmico que será referenciado ao longo deste trabalho.

O diagrama ER do esquema conceitual é o seguinte:



O diagrama ER acima é auto explicativo, exceto pelos seguintes pontos. Primeiro, EVENTO é especializado em dois conjuntos disjuntos, CLASSE e OUTRO, definidos como visões parciais baseadas nos valores de TIPO em EVENTO. A visão parcial DESOCUPADO indica, para cada sala, quando ela não está sendo usada e, portanto, ela pode aparecer, por exemplo, quando modelarmos a parte do esquema conceitual referente ao Departamento de Manutenção. Finalmente, a visão parcial REGISTRO indica os cursos que cada estudante está fazendo e é a janela de comunicação com a parte do esquema que interessa, por exemplo, ao Departamento de Admissão e Registro.

### 3. LINGUAGEM DE DEFINIÇÃO DE DADOS BÁSICA

Esta seção apresentará a sintaxe de uma LDD básica para o Modelo Entidade-Relacionamento na qual incorporaremos mais tarde o conceito de visões parciais.

Para evitar vários detalhes tediosos, não especificaremos a sintaxe dos nomes de esquemas e atributos e assumimos que é dado um conjunto **D** de *tipos de domínio*. Como usualmente, todo componente opcional será colocado dentro de colchetes.

Um *ER-esquema* é um comando da forma

```

esquema S
      L1, ..., Lk
fimesquema
  
```

onde S é o *nome* e L<sub>1</sub>, ..., L<sub>k</sub> é a lista dos *ER-comandos* do esquema.

Os ER-comandos válidos, junto com as restrições que eles devem satisfazer no contexto de um dado ER-esquema, são definidos da seguinte maneira.

Primeiro, para descrever conjunto de entidades, introduzimos o *esquema de entidade*, que são comandos com o seguinte formato:

```
entidade E
    [atributos A1 D1, ..., An Dn]
    [chave K]
```

onde:

- E0. E deve ser um nome válido e distinto de todos os outros nomes de esquemas de entidades ou de relacionamentos do ER-esquema;
- E1. A<sub>1</sub>, ..., A<sub>n</sub> devem ser nomes de atributos válidos e distintos;
- E2. D<sub>1</sub>, ..., D<sub>n</sub> devem ser tipos de domínios válidos em D;
- E3. K deve ser uma lista de nomes de atributos distintos de E.

Dizemos que E é o *nome*, A<sub>1</sub>, ..., A<sub>n</sub> é a lista de *nome de atributos* e K é a *chave* do esquema. Também dizemos que D<sub>i</sub> é o *tipo do domínio* de A<sub>i</sub>, para i = 1, ..., n. Conforme indicado, a especificação dos atributos e da chave é opcional.

Note que os conjuntos de entidades não são tipificados. Entretanto, dizemos que um esquema de entidade tem *aridade* igual a 1.

Por exemplo, o esquema de entidade SALA de nosso banco de dados acadêmico é definido da seguinte maneira:

```
entidade SALA
    atributos PREDIO#   char(4),
              SALA#    char(4),
              TAMANHO  integer
    chave PREDIO#, SALA#
```

Antes de introduzirmos os comandos que descrevem conjuntos de relacionamentos, alguns comentários se tornam necessários. Primeiro, é permitido definir conjuntos de relacionamentos sobre conjuntos de entidades ou de relacionamentos. Também é permitido um conjunto de entidades ou de relacionamentos participar mais de uma vez em um conjunto de relacionamentos, desde que a cada ocorrência seja dado o nome de um papel distinto. Um conjunto de relacionamento pode opcionalmente ser definido como total [TYF] sobre o papel de um conjunto de entidades ou de relacionamentos. Por último, é introduzido a noção de identificador como correspondente da noção de chave para os conjuntos de relacionamentos. Assim, por exemplo, se indicarmos que SALA e HORARIO são os identificadores do esquema de relacionamento ALOCADA que foi definido sobre SALA, HORARIO e EVENTO, estamos dizendo que cada sala em cada período de tempo pode ser alocada a um só evento. Usualmente os identificadores são expressos rotulando com "1" ou "n" os arcos que deixam o losango que representa um conjunto de relacionamentos em um diagrama ER.

Mais precisamente, um *esquema de relacionamento* é um comando com a seguinte forma:

relacionamento R

sobre  $O_1$  [como  $N_1$ ] [total], ...,  $O_m$  [como  $N_m$ ] [total]  
[atributos  $A_1, D_1, \dots, A_n, D_n$ ]  
[identificadores K]

Dizemos que R é o nome,  $A_1, \dots, A_n$  é a lista de *nomes de atributos* e K é a lista de *identificadores* do esquema. Também podemos dizer que o esquema de relacionamento R tem m *papeis*. O esquema no *i*-ésimo papel é  $O_i$  e que o nome do *i*-ésimo papel é  $N_i$ , se especificado, ou caso contrário, o próprio  $O_i$ .

Quando "total" é especificado para " $O_i$  [como  $N_i$ ]", dizemos que R é *total* sobre o *i*-ésimo papel. Finalmente, definimos a *aridade* de R como a soma das aridades de  $O_1, \dots, O_m$ . O conceito de aridade se torna necessário porque permitimos que um esquema de relacionamento seja definido sobre esquemas de relacionamentos.

A partir deste ponto, quando nos referirmos a um esquema queremos dizer indistintamente tanto um esquema de entidade quanto um esquema de relacionamento.

Exige-se que:

- R0. R deve ser um nome válido e distinto de todos os outros nomes de esquemas do ER-esquema;
- R1, R2 (da mesma forma que E1 e E2);
- R3.  $O_1, \dots, O_m$  devem ser nomes de esquemas válidos, não necessariamente distintos, mas definidos no ER-esquema, e  $m \geq 2$ ;
- R4. os nomes dos papeis de R devem ser distintos;
- R5. K deve ser uma lista de papeis distintos de R.

Note que, por R4, os nomes dos papeis são únicos. Assim sendo, podemos nos referir a um papel específico de R pelo seu nome ao invés de pela sua posição. Entretanto, observe que o mesmo esquema pode participar em mais de um papel em R.

Uma vez que permitimos relacionamentos sobre relacionamentos, devemos também evitar que um esquema de relacionamento R seja definido sobre si mesmo já que do contrário a semântica de R se tornaria indefinida. Mais precisamente, exige-se adicionalmente que:

- R6. o ER-esquema não pode ter uma cadeia de esquemas de relacionamentos  $O_0, O_1, \dots, O_{n-1}, O_n$  tal que  $O_i$  seja sobre  $O_{i+1}$ , para  $i = 0, \dots, n-1$ , onde a soma é módulo n.

Por exemplo, o esquema de relacionamento ALOCADA é definido como:

relacionamento ALOCADA

sobre SALA, HORARIO, EVENTO  
atributos PESSOA\_CONTACTO char(20),  
TELEFONE\_CONTACTO char(7)  
identificadores SALA, HORARIO

Prosseguindo com a descrição da linguagem, para capturar a relação IS\_A, introduzimos o comando *especialização* com a seguinte forma:

$O$  is-a  $O_1, \dots, O_n$

onde

S0.  $O, O_1, \dots, O_n$  devem ser nomes distintos de esquemas de entidades ou relacionamentos, de mesma aridade, definidos no ER-esquema;

Neste caso dizemos que  $O$  *especializa*  $O_i$ , ou que  $O_i$  *generaliza*  $O$ , para  $i=1, \dots, n$ . Também dizemos que  $O$  *transitivamente especializa*  $O'$  sss  $O$  especializa  $O'$  ou existe um esquema  $O''$  tal que  $O$  especializa  $O''$  e  $O''$  transitivamente especializa  $O'$ .

Note que exigimos somente que todos os esquemas envolvidos em um comando de especialização tenham a mesma aridade. Assim, qualquer esquema de entidade pode especializar qualquer outro esquema de entidade e qualquer esquema de relacionamento pode especializar qualquer outro esquema de relacionamento, desde que ambos tenham a mesma aridade. Em adição, não exigimos que a relação de especialização seja uma hierarquia ou mesmo que ela seja irreflexiva. Na verdade, permitimos a existência de uma cadeia  $O_0, \dots, O_{n-1}, O_0$  tal que  $O_i$  especializa  $O_{i+1}$ , para todos  $i=0, \dots, n-1$  (soma é módulo  $n$ ). Neste caso todos estes esquemas denotarão o mesmo conjunto de objetos e qualquer ferramenta razoável de projeto, que suporte esta LDD, pode então dar uma mensagem de aviso.

Dizemos que um atributo  $A$  é *herdado por*  $O$  de  $O'$  sss  $A$  é o nome de um atributo de  $O'$  e  $O$  transitivamente especializa  $O'$ . Observe que devemos especificar o esquema de origem de  $A$  uma vez que  $O$  pode transitivamente especializar esquemas com atributos com o mesmo nome.

Resumidamente, um estado de um ER-esquema  $S$  mapeia cada esquema de entidade de  $S$  em um conjunto de entidades, obtidas de um conjunto comum  $E$ , cada esquema de relacionamento em um subconjunto do produto cartesiano dos conjuntos de entidades envolvidas, e cada atributo de um esquema em uma função que mapeia cada objeto em um valor obtido de um domínio apropriado. É muito importante salientar que, ao contrário do modelo relacional, entidades e relacionamentos tem existência independente dos valores dos atributos, os quais entretanto são necessários para localizá-los no banco de dados. Dizemos que um estado  $\alpha$  de um ER-esquema  $S$  é consistente sss: (i) para qualquer esquema  $O$  e  $O'$  de  $S$ , se  $O$  especializa  $O'$  então todo objeto no conjunto associado a  $O$  também pertence ao conjunto associado a  $O'$ ; (ii) se para qualquer esquema de entidade  $E$  de  $S$  as chaves forem únicas e (iii) se para qualquer esquema de relacionamento  $R$  de  $S$  seus identificadores forem únicos.

Podemos também aplicar a noção de herança de atributos em uma colocação muito mais geral, ainda que semanticamente definida. Sejam  $O$  e  $O'$  esquemas e seja  $A$  o nome de um atributo de  $O'$ . Então,  $A$  é *herdado por*  $O$  de  $O'$  sss, em qualquer estado do ER-esquema temos que todo objeto no conjunto associado a  $O$  também pertence ao conjunto associado a  $O'$ .

#### 4. ESTENDENDO A LINGUAGEM DE DEFINIÇÃO DE DADOS COM VISÕES PARCIAIS

Diferentemente da maioria das linguagens relacionais e mesmo das que seguem o modelo ER, uma consulta na nossa linguagem define tanto um conjunto de objetos quanto um conjunto de atributos para estes objetos. Por simplicidade conceitual, a linguagem não adotará o conceito de "caminho", encontrado por exemplo na linguagem LAMBDA [Ve]. Afora estes pontos, a sintaxe e a semântica das consultas se manterão muito próximas daquelas da linguagem SQL.

Intuitivamente, um termo ou aponta para uma entidade, ou para um relacionamento ou armazena o valor de um atributo. Nos dois primeiros casos, o tipo do termo coincide com o tipo do objeto para o qual ele aponta, enquanto que no terceiro caso o tipo do termo coincide com o tipo do seu valor. Os termos básicos serão as variáveis, que estarão ligadas a conjuntos de entidades ou relacionamentos. Uma consulta sempre especifica um conjunto de variáveis e o conjunto de objetos aos quais elas estão associadas.

Mais precisamente, sejam  $F_1, F_2, \dots$  variáveis ligadas a esquemas  $O_1, O_2, \dots$  de um ER-esquema  $S$ . O conjunto de *termos* de  $S$  é definido recursivamente como:

- (a) se  $F_i$  é uma variável ligada a um esquema de entidades ou a um esquema de relacionamentos  $n$ -ário  $O_i$ , então  $F_i$  é um termo do tipo entidade ou relacionamento  $n$ -ário; a aridade de  $F_i$  é aquela de  $O_i$ ;
- (b) se  $F_i$  é uma variável ligada a um esquema de relacionamentos  $O_i$  e se  $N$  é um papel de  $O_i$  associado a um esquema de entidades ou a um esquema de relacionamentos  $n$ -ário  $O_i'$ , então  $F_i.N$  é um termo do tipo entidade ou relacionamento  $n$ -ário; a aridade de  $F_i.N$  é aquela de  $O_i'$ ;
- (c) se  $A$  é um atributo de um esquema  $O_i$ , se  $F_i$  é uma variável ligada a  $O_i$ , e se  $A$  é do tipo  $D$ , então  $F_i.A$  é um termo do tipo  $D$ ;
- (d) se  $A$  é um atributo herdado de  $O_j$  por um esquema  $O_i$ , se  $F_i$  e  $F_j$  são variáveis ligadas a  $O_i$  e  $O_j$ , e se  $A$  é do tipo  $D$ , então  $F_i.F_j.A$  é um termo do tipo  $D$ .

O conjunto das *condições de pesquisa* sobre  $S$  é por sua vez definido recursivamente como:

- (a) se  $t$  e  $t'$  são termos de tipos  $T$  and  $T'$ ,  $k$  é uma constante de tipo  $T'$  e  $\theta$  é um operador de comparação do tipo  $(T, T')$ , então  $t\theta k$  e  $t\theta t'$  são ambas condições de pesquisa sobre  $S$ ;
- (b) se  $s$  é uma condição de pesquisa, então  $\neg s$  também é uma condição de pesquisa;
- (c) se  $s_1, \dots, s_m$  são condições de pesquisa, então as expressões  $(s_1 \text{ or } \dots \text{ or } s_m)$  e  $(s_1 \text{ and } \dots \text{ and } s_m)$  também são condições de pesquisa sobre  $S$ ;

Uma *consulta simples* sobre  $S$  é uma expressão da forma:

```
select (u1, ..., uk) [(t1, ..., tm)]
from O1 F1, ..., On Fn
[where P]
```

onde:

- Q0.  $O_i$  deve ser um nome de esquema de  $S$  e  $F_i$  deve ser uma variável, para  $i = 1, \dots, n$ ;
- Q1. cada  $u_i$  deve ser um termo sobre uma das variáveis  $F_1, \dots, F_n$  e deve ser do tipo entidade ou relacionamento  $1$ -ário;
- Q2. cada  $t_i$  deve ser um termo sobre as variáveis  $F_1, \dots, F_n$  e deve ser de um tipo tirado do conjunto  $D$ ;
- Q3.  $P$  deve ser uma condição de pesquisa sobre as variáveis  $F_1, \dots, F_n$ .

Dizemos que a consulta possui *aridade* igual à soma das aridades de  $u_1, \dots, u_k$ , possui *aridade para atributos* igual a  $m$  e possui *tipo de domínio*  $(D_1, \dots, D_m)$ , se  $t_j$  possui tipo  $D_j$ , para  $j = 1, \dots, m$ . Dizemos ainda que  $F_i$  é a variável *ligada* a  $O_i$  na consulta, para  $i = 1, \dots, n$ , e que  $P$  é a *condição de pesquisa* da consulta.

Em geral, uma consulta simples define

- um conjunto de relacionamentos sobre os objetos associados a  $u_1, \dots, u_k$ , se  $k > 1$ , ou um subconjunto do conjunto de objetos associados a  $u_1$ , se  $k = 1$ ;
- um conjunto de funções de valoração para atributos, definidas com a ajuda da lista de atributos resultantes.

Consultas mais complexas são definidas através da união, interseção ou diferença de consultas de mesma aridade.

Um *esquema de visão parcial* é um comando da forma:

```
visao parcial V
    [atributos nativos ( $A_1 D_1, \dots, A_k D_k$ )]
    [atributos derivados ( $B_1, \dots, B_m$ )]
    [chave K]
    where Q
```

onde:

- V0.  $V$  deve ser um nome válido e distinto de todos os outros nomes de esquema do ER-esquema;
- V1.  $A_1, \dots, A_k, B_1, \dots, B_m$  devem ser nomes distintos de atributos;
- V2.  $D_1, \dots, D_k$  devem ser domínios de tipos válidos;
- V3.  $K$ , se definido, deve ser uma lista de atributos nativos distintos;
- V4.  $Q$  deve ser uma consulta sobre o ER-esquema com aridade para atributos igual a  $m$

Dizemos que  $V$  é o *nome* da visão parcial, que o *domínio de tipos* de  $B_i$  é o  $i$ -ésimo tipo de domínio de  $Q$  e que a *aridade* de  $V$  é a de  $Q$ . Dizemos também, que  $V$  é uma *visão\_entidade* sss  $V$  tem aridade 1, caso contrário  $V$  é uma *visão\_relacionamento*.

Por exemplo, a visão parcial que define a especialização de EVENTO em um conjunto de objetos do tipo "classe" é a seguinte:

```
visao parcial CLASSE
    atributos nativos (TAMANHO integer)
    where
        select (e)
            from EVENTO e
            where e.TIPO = 'classe'
```

Note que, para cada estado  $\alpha$ , CLASSE define um subconjunto do conjunto de entidades associado a EVENTO por  $\alpha$ . Por conseguinte, de acordo com a definição de herança dada na seção 3, podemos dizer, corretamente, que CLASSE herda todos os atributos de EVENTO.

A visão parcial que especifica DESOCUPADO é definida por:

```

visao parcial DESOCUPADO
  atributos derivados (PREDIO#,SALA#,DIA,HORA)
  where
    (select (s,h) (s.PREDIO#,s.SALA#,h.DIA,h.HORA)
     from SALA s, HORARIO h
     -
     select (a.SALA,a.HORARIO)
     from ALOCADA a)

```

Para cada estado  $\alpha$ , DESOCUPADO define um conjunto de relacionamentos sobre os conjuntos de entidades, associados por  $\alpha$  a SALA e HORARIO, que contém todos os pares que não estão no conjunto associado por  $\alpha$  a ALOCADA.

Um *ER-esquema estendido* é um ER-esquema como previamente definido, exceto que esquemas de visão de entidades e de relacionamentos podem participar onde somente antes era permitida a participação de esquemas de entidades e de relacionamentos. Entretanto exige-se que:

C0. uma visão parcial não pode ser definida transitivamente sobre si mesma.

Do ponto de vista dinâmico tratamos um esquema de visão parcial  $V$  como uma visão comum. Assim sendo, todas as operações envolvendo os esquemas sobre os quais  $V$  é definida se propagam imediatamente para  $V$ . Atualizações sobre os atributos nativos de  $V$  também são permitidas uma vez que eles são armazenadas explicitamente. Entretanto, todas as outras operações sobre  $V$  são exatamente como operações sobre visões e, assim sendo, não podem ser imediatamente traduzidas em operações sobre os esquemas sobre os quais  $V$  é definida. Assim, é razoável proibir tais operações sobre  $V$ , exceto em alguns casos especiais [FC].

## 5. CONCLUSÕES

Nossa maior motivação foi prover uma definição uniforme para certos princípios de abstração que auxiliam na organização do projeto de um esquema conceitual. O ponto focal foi permitir que visões, possivelmente equipadas com novos atributos armazenados, representem o mesmo papel que os esquemas de entidades e relacionamentos representam no processo de projeto conceitual.

Apresentamos também uma definição precisa para a linguagem de definição de dados básica e para uma linguagem de consulta. Ambas as linguagens podem ser estendidas para incluir mais facilidades. Por exemplo, a LDD pode ser estendida para capturar novas classes de restrições de integridade e a linguagem de consulta para incluir funções de agregação tais como SUM e AVERAGE.

Um subconjunto da linguagem de definição de dados completa está sendo atualmente definido para servir como a interface básica para a ferramenta de modelagem conceitual que substituirá a ferramenta CHRIS descrita em [FCT].

## AGRADECIMENTOS

Agradecemos ao Prof. Antonio L. Furtado pela cuidadosa leitura da versão inicial deste texto.

## REFERÊNCIAS

- [BCAZ] U. Bussolati, S. Ceri, V. De Antonellis and B. Zonta, "Views Conceptual Design", in *Methodology and Tools for Data Base Design*, S. Ceri (ed.), North-Holland, 1983.
- [DC] A. Dogac and 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.
- [EL] R. Elmasri and J. Larson, "A Graphical Query Facility for ER Databases", *Proc. of the Four International Conference on Entity-Relationship Approach*, Chicago, Illinois (Oct. 1985), 236-245.
- [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), 127&-a.142.
- [FCT] A.L. Furtado, M.A. Casanova and L. Tucherman, The CHRIS consultant, Proc. 6th International Conference on Entity-Relationship Approach, November 1987 (pp. 479-486).
- [KL] W. Kozaczynski and L. Lillien, "An Extended Entity-Relationship (E<sup>2</sup>R) Database Specification and its Automatic Verification and Transformation into the Logical Relational Design", *Proc. of the Sixth International Conference on Entity-Relationship Approach*, New York (Nov. 1987), 497-513.
- [LD] J.A. Larson and P.A. Dwyer, "Defining External Schemas for an Entity-Relationship Database", in *Entity-Relationship Approach to Software Engineering*, C.G. Davis, S. Jajodia, P.A. Ng and R.T. Yeh (eds.), Elsevier Science Pub. (1983), 347-363.
- [Li] T.W. Ling, "A Three Level Schema Architecture ER-based Data Base Management System", *Proc. of the Sixth International Conference on Entity-Relationship Approach*, New York (Nov. 1987), 497-513.
- [MR] V.M. Markowitz and Y. Raz, "ERROL: An Entity-Relationship, Role based Query Language", in *Entity-Relationship Approach to Software Engineering*, C.G. Davis, S. Jajodia, P.A. Ng and R.T. Yeh (eds.), Elsevier Science Pub. (1983), 329-346.
- [SKK] H. Sakai, H. Kondo and Z. Kawasaki, "A Development of a Conceptual Schema Design Aid in the Entity-Relationship Model", in *Entity-Relationship approach to System Analysis and Design*, P.P. Chen (ed.), North-Holland (1981), 415-432.
- [SSW] P. Scheuermann, G. Schiffner, H. Weber, "Abstraction capabilities and invariant properties modelling within the Entity-Relationship approach", in *Entity-Relationship approach to System Analysis and Design*, P.P. Chen (ed.), North-Holland (1981), 121-140.
- [TYF] T.J. Teorey, D. Yang and J.P. Fry, "A Logical Design Methodology for Relational Databases Using the Extended Entity-Relationship Model", *Computing Surveys*, Vol.18, No.2 (June 1986), 197-222.
- [Ve] F. Velez, "Lambda: An Entity-Relationship based Query Language for the Retrieval of Structured Documents", *Proc. of the Four International Conference on Entity-Relationship Approach*, Chicago, Illinois (Oct. 1985), 236-245.