XIV Conferencia Latinoamericana de Informática
17avas. Jornadas Argentinas de Informática
e Investigación Operativa.
Buenos Aires, Setiembre 1988.

REVER: Uma Interface Semântica para Bancos de Dados Relacionais

Mônica de Lima Nogueira*
Claudia Bauzer Medeiros†
DCC - IMECC - UNICAMP - CP 6065
CEP 13.081 Campinas, SP - BRASIL
Tel (0192) 391301, ramal 3442

Sumário

Este artigo descreve as características de uma interface E-R para um banco de dados relacional - o sistema REVER. O usuário do sistema pode utilizá-lo tanto para fazer o projeto do banco de dados quanto para consultas e atualizações. A visão do usuário é a de criação de elementos de um diagrama E-R estendido, e de manipulação de suas ocorrências. O sistema se encarrega de mapear as solicitações do usuário para operações sobre o esquema e instâncias do banco de dados relacional correspondente.

Abstract

This paper describes the features of an E-R interface to a relacional database - the REVER system. The system allows the user to design the database using only the E-R specification, as well as to query and update it. The user's view is always that of an E-R (extended) diagram, in the design and in the operational phases. The system maps the user's requests on the E-R scheme and instances to operations on the corresponding relational database.

*Tecnóloga em Eletrônica (UTAM 1982) e Eng. Elétrica/Eletrônica (FUA 1983). Mestranda em Ciência da Computação (UNICAMP 1988). Área de interesse: Banco de Dados.

†Eng. Eletrônica (PUC/RJ 1976), Mestre em Informática (PUC/RJ 1979), PhD. in Computer Science (Waterloo 1985). Áreas de interesse: Banco de Dados, Eng. Software.

1 Introdução

Até o surgimento da proposta de [CHEN76] para modelagem do mundo real em termos de entidades, relacionamentos e atributos, os bancos de dados existentes não eram capazes de expressar em correpondência direta os objetos de uma aplicação. Na realidade, o ambiente era moldado para um modelo de estrutura mais ou menos rígida.

Algumas das vantagens do Modelo Entidade-Relacionamento (E-R) segundo [CHUN83] são: permitir capturar e preservar algumas das importantes informações semânticas do mundo real, possibilitar linguagens de computador baseadas em E-R mais naturais para usuários não-técnicos. A facilidade de tradução entre o modelo E-R e os modelos de dados convencionais indicam que o modelo deveria ser padronizado para modelagem de esquema conceitual nas arquiteturas ANSI/SPARC e para manipulação da tradução de dados em ambientes de banco de dados heterogêneos.

Desde a proposta de Chen [CHEN76] inúmeras extensões têm sido defendidas ([ELMA85], [FELD86], [PARE86], [SCHE80], [SMIT77], [TEOR86], [WAGN87]) para o Modelo Entidade-Relacionamento. No entanto, não existe um consenso a respeito da Linguagem de Manipulação de Dados (DML) mais adequada para este modelo, que englobe todas essas extensões. Outro problema apontado é a fraqueza de definição semântica e a falta de poder para expressar propriedades temporais.

Muitos dos modelos propostos são distintos entre si nas definições e uso dos conceitos de relacionamento e atributo. Em [CHEN83b] é apresentado um quadro para classificar as extensões E-R existentes e é discutido como estas podem ser traduzidas de uma para a outra.

[TEIC83] apresenta uma retrospectiva dos trabalhos baseados no E-R-Approach (ERA) e resultados obtidos na sua aplicação. O enfoque do modelo ERA surge como o mais geral, capaz de fornecer uma visão mais natural dos sistemas e de preencher a lacuna entre análise de sistemas, modelagem de dados e análise funcional.

Outro modelo com o mesmo tipo de enfoque é EAS e a linguagem EAS-E, sistemas experimentais de desenvolvimento de aplicações baseados no formalismo de entidades, atributos, conjuntos e eventos. Segundo [MARK83] as visões E-R e EAS são similares no aspecto que possibilitam ao usuário modelar o mundo em vez de obrigá-lo a especificar como o computador deve representar este mundo e ambas fornecem um conjunto simples mas geral de conceitos de modelagem. O poder de cada visão é também sua fraqueza. Na visão EAS é necessário que o usuário declare antecipadamente os relacionamentos, o que leva à produção de um programa que executa mais eficientemente. Ainda comparado ao modelo E-R, o sistema EAS normalmente envolve mais problemas quando são efetuadas mudanças nas definições do banco de dados.

Há, além disso, o problema de que o modelo E-R é um modelo semântico, que deve depois ser analisado por gerentes de banco de dados para especificar o banco de dados que o represente. Assim, embora ao leigo seja facilitado expressar suas especificações no modelo semântico, a implementação fica a cargo de especialistas. Além disto, uma vez gerado o banco de dados, o usuário não mais dispõe das abstrações iniciais.

Dentre trabalhos que abordam este problema situam-se [SHAV81], que discute uma ferramenta para modelagem conceitual baseada em elementos E-R; o Projeto GAMBIT [BRAE85], que dentre outros pontos facilita ao usuário a definição de restrições de integridade; e o Sistema CHRIS [FURT87], uma implementação em PROLOG de uma interface E-R para o banco de dados relacional.

O presente trabalho enfoca alguns dos aspectos mencionados nos parágrafos anteriores, tentando solucionar o problema através de uma interface E-R para manipulação de um banco de dados relacional. REVER (RElacional Via Entidade-Relacionamento) é a interface proposta, que amplia

o universo de usuários que virão a usufruir das qualidades do Banco de Dados Relacional sem que para isso lhes seja impingida a necessidade de se especializarem nessa área. Como resultado, o usuário poderá desfrutar das vantagens da modelagem semântica E-R aliadas ao rigor teórico existente para o Modelo Relacional.

Pressupõe-se que o usuário da interface já haja procedido à análise do ambiente e suas necessidades, havendo elaborado a especificação do sistema, ou esquema funcional [ROUS84]. Uma vez estabelecido um modelo funcional, ou seja, quando identificadas as principais atividades ou funções e as associações existentes entre elas, pode-se gerar um esquema conceitual E-R usando a interface.

Um dos objetivos deste trabalho é integrar técnicas de projeto de banco de dados e de projeto de programas, dado que um sistema de BD não é só um esquema (componente estrutural) ou só um conjunto de transações (componente procedural), mas sim um esquema com transações sendo aplicadas a este esquema [ROUS84].

Elmasri [ELMA83], em sua proposta de uma linguagem (GORDAS) para o modelo E-R, classifica os diferentes tipos de interfaces em: procedurais, dedicadas a profissionais para definição dos programas de aplicação que serão usados pelos usuários; não-procedurais, para interação de usuários técnicos e semi-técnicos com o DBMS a partir de um terminal; e amigáveis, que inclui menus para orientar usuários não-técnicos e uso de linguagens naturais. O sistema REVER se enquadra nesta última categoria. Ele está sendo implementado como parte de um projeto de tese de mestrado no DCC/UNICAMP [NOGU88].

Este artigo possui 5 seções adicionais. A primeira delas trata dos conceitos e primitivas do modelo E-R estendido utilizado e seu mapeamento para o modelo Relacional; a seguir tem-se a especificação da interface, sendo discutidas as linguagens de definição dos objetos do modelo e de consulta/atualização. A seção 4 trata de alguns detalhes da implementação e a seção 5 discute alguns problemas de atualização. A seção final apresenta problemas e conclusões.

2 Conceitos e Nomenclatura

2.1 Modelo Entidade-Relacionamento

Esta seção pressupõe que o leitor já conhece as características básicas do modelo E-R. Serão apenas ressaltados alguns termos utilizados neste artigo. Para maiores detalhes, sugere-se a leitura de [CHEN76] e de [MAIE85].

Uma *"entidade"* representa um conjunto de ocorrências de informações de uma mesma espécie, com características: *"atributos"*. As diferentes ocorrências de uma entidade são distinguidas através de um ou mais atributos identificadores (*"chave"*).

Os atributos podem ser: 1. compostos - formados por mais de um atributo, identificadores ou não (descritores); 2. multivalorados - quando para um único atributo identificador ou chave correspondem mais de um atributo descritor ou não-chave.

Um *"relacionamento"* é a representação das associações que ligam uma (auto-relacionamento) ou mais entidades, podendo ter atributos próprios.

Um dos aspectos mais importantes que diferenciam o modelo original de Chen de suas extensões é a definição dos relacionamentos. Um exemplo são os conceitos de 'grau'(unário,binário.ternário), 'conectividade' (1:1, 1;n, m:n) e 'classe-de-membro' (mandatório ou opcional) dos relacionamentos conforme adotados por [TEOR86].

Um relacionamento (entre entidades A e B) é dito *"total"* de B em relação a A quando a

existência de cada ocorrência da entidade B *"depende"* da existência de uma ocorrência correspondente da entidade A. Essa entidade com dependência de existência é chamada de entidade *"fraca"* no modelo original de Chen. Na extensão E-R adotada, não se identifica uma entidade fraca, mas sim o relacionamento total, indicado graficamente através de um círculo escuro conforme exemplifica a figura 1. A modificação de representação (em função do relacionamento) é particularmente interessante, pois marca o relacionamento que estabelece a relação de dependência de existência, evitando ambiguidades quando essa mesma entidade participa de outros relacionamentos onde tal dependência não existe.

Caso a existência de uma entidade seja completamente independente da existência de outra entidade, o relacionamento que as interliga é dito *"parcial"*.

Um relacionamento total é representado por uma *"dependência de inclusão"* do modelo relacional e implica que a inclusão de uma tupla em uma relação depende da existência de uma tupla correspondente em outra relação (aquela à qual está ligada através do relacionamento total).

2.2 Modelo E-R Estendido

Das inúmeras contribuições existentes para o modelo E-R, a de Wagner e Medeiros reuniu um grupo de proposições cujo objetivo principal é ampliar a riqueza semântica do modelo através do aumento dos níveis de abstração.

O modelo E-R estendido aqui utilizado é o apresentado em [WAGN87], composto de blocos adicionais mais complexos baseados nas primitivas do modelo original, aqui chamados objetos complexos: 'Agregação', 'Generalização' e 'Hierarquia de Tipos'. A figura 2 mostra a simbologia usada para tais elementos conforme foi proposta por Navathe e Cheng [NAVA83]. Maiores detalhes e exemplos da extensão adotada podem ser encontrados em [WAGN87].

Entende-se *"agregação"* como o encapsulamento de uma ou mais entidades e um relacionamento que as interliga (figura 3). Na realidade, as ocorrências do agregado correspondem às associações expressas pelo relacionamento que encapsula.

A entidade complexa *"generalização"* pode ser caracterizada por uma árvore de 2 níveis onde a raiz engloba os argumentos comuns a um conjunto de elementos (as folhas). Cada elemento-folha tem como atributos aqueles herdados da raiz, bem como atributos que o distinguem das demais folhas (figura 4). A cada ocorrência de um elemento-raiz corresponde uma ocorrência de alguma folha; as folhas correspondem a conjuntos mutuamente exclusivos. Diz-se que a raiz é uma *"generalização"* das folhas e que estas são uma *"especialização"* da raiz. Esta caracterização é recursiva, ou seja, cada folha pode ser por sua vez a raiz de uma nova árvore: o grafo correspondente adquire assim as características de uma árvore de nós heterogêneos.

Esta definição de entidades de uma generalização é semelhante àquela aplicada por [TEOR86] e [BRAE85]. Em [WELD80] a generalização é representada por um objeto geral com as características comuns de uma coleção de objetos (mais específicos e mutuamente exclusivos entre si). A existência de categorias não exclusivas implica em diferentes 'agrupamentos' de generalizações, sendo que em cada agrupamento as categorias são distintas.

A generalização tem uma variante denominada *"hierarquia de tipos"*. Se em generalização ocorre exclusão mútua entre os elementos-filho participantes, o mesmo não é necessariamente verdade em *"hierarquia de tipos"*, daqui por diante referenciada simplesmente como *"hierarquia"*, figura 5. O trabalho de [FURT87] implementa o conceito de hierarquia de tipos que também é conhecida por 'hierarquia is-a'. Em [TEOR86] temos o mesmo conceito indicado como 'hierarquia de subconjuntos' onde as entidades-folha não possuem exclusão-mútua ocorrendo sobreposição de papéis sem que

haja qualquer prejuízo da estrutura; os únicos atributos repetidos no caso são aqueles identificadores das entidades, ou chaves.

A princípio nada impede a existência de árvores cujos nós são relacionamentos (vide Wagner e Medeiros [WAGN87]), ou mesmo árvores heterogêneas (onde os nós correspondem a elementos de natureza diversa, p.ex., folhas compostas por hierarquias e agregações).

2.3 Modelo Relacional

Simplificadamente, um banco de dados relacional é composto por um conjunto de relações ou tabelas. Um *"esquema"* R(I) é formado pelo conjunto de (nomes de) atributos {A(1)...A(N)} onde cada um destes é definido sobre um domínio, DOM{A(I)}. Um *"esquema relacional"* RR é composto de um conjunto de esquemas R(I). Por analogia a esquema de relação este artigo denomina esquema E-R à descrição dos objetos no projeto conceitual no modelo E-R estendido, onde se especificam os objetos do projeto e suas interligações, sem preocupação com ocorrências.

Uma *"relação"* R(I) é uma ocorrência de um esquema R(I). Diz-se também que uma relação é uma tabela formada por um conjunto de tuplas T. Uma *"tupla"* T(I) é um conjunto de valores ordenados segundo os atributos da relação, com significação válida dentro do domínio dos atributos correspondentes. De novo, por analogia, o artigo referencia ocorrências de um objeto E-R (por exemplo, dada uma entidade MÉDICO, cada ocorrência é o conjunto de dados relativo a um médico). Um *"banco de dados relacional"* é um conjunto de relações {R(I)}. Um estado do banco de dados é uma ocorrência do esquema relacional RR, sobre o qual está definido.

Supõe-se que o leitor conheça as definições de *"chave"* e *"dependências funcionais e multivaloradas e 1NF"*.

Um relacionamento total é representado por uma *"dependência de inclusão"* do modelo relacional e implica que a inclusão de uma tupla em uma relação depende da existência de uma tupla correspondente em outra relação (aquela à qual está ligada através do relacionamento total).

2.4 Mapeamento E-R ↔ Relacional

A interface REVER pressupõe regras de mapeamento entre o modelo E-R estendido adotado e o modelo relacional. A interface tem dois níveis de mapeamento: para geração do esquema e para consultas/atualizações de relações. No primeiro caso, o usuário especifica seu projeto semântico em termos de objetos E-R e a interface gera o esquema relacional correspondente segundo regras pré-definidas de mapeamento. No segundo caso, o usuário consulta/atualiza objetos E-R, o que é traduzido pela interface em consultas e atualizações às relações base. Há dois aspectos a se considerar: 1. encontrar a correspondência entre os dois modelos; e 2. sintetizar algumas propriedades dos dois modelos em uma forma compreensível [CHEN83a].

A cada atributo do modelo E-R corresponderá um único atributo relacional. Seguindo o proposto por [TEOR86] os atributos multivalorados devem ser transformados em entidades ainda na fase de projeto E-R. Portanto, não há necessidade de mapeá-los para o modelo relacional.

A fase de mapeamento englobará a transformação de entidades, relacionamentos e dos objetos complexos: generalização, hierarquia e agregação. A geração do esquema relacional segue as seguintes regras, não pressupondo otimização ou minimização de esquema.

- Cada entidade simples é convertida em uma relação com esquema igual ao da entidade original, identificando atributos chave e não-chave.

- Cada relacionamento é transformado em um esquema composto das chaves das entidades que liga acrescido dos atributos específicos do relacionamento. Esse mapeamento é adotado por que ressalta as seguintes vantagens: 1. o uso de uma relação separada para relacionamentos permite armazenar relacionamentos sem modificar outras relações do BD relacional; 2. esta implementação uniforme não precisa ser modificada para relacionamentos de cardinalidades diferentes (1:1, 1:n, m:n); e 3. possibilita de forma simples a criação e remoção de relacionamentos sem modificação das tabelas das entidades.

- As entidades complexas: Generalização e Hierarquia, seguem o padrão, adotado por [WELD80]:

  - o nó 'pai' é transformado em um esquema com a chave e atributos (comuns a todas as entidades 'filho') desse nó;
  - os nós 'filhos' são transformados em esquemas contendo cada um a chave da entidade 'pai' mais os atributos específicos de cada filho.

- A entidade complexa Agregação é convertida obedecendo os mesmos critérios usados para relacionamentos.

O modelo semântico proposto considera dependências de chave e de existência. Estas dependências são mapeadas, no modelo relacional, para dependências funcionais (dentro de uma relação) e dependências de inclusão (correspondentes a relacionamentos totais e manutenção de árvores com heranças de propriedades).

A herança de propriedades em uma árvore será garantida pela inclusão da chave do pai entre os atributos dos filhos, sendo mantida portanto através dos mecanismos de integridade associados a chaves. (A chave de cada filho englobará obrigatoriamente a chave do pai).

A manutenção de dependências de inclusão e de propriedades como exclusão mútua (de ocorrências de filhos em uma generalização) apresentam problemas teóricos complexos. A análise teórica destes problemas escapa ao âmbito deste trabalho. A seção 2.5 apresenta algumas simplificações do trabalho que permitem eliminar alguns destes problemas.

2.5 Simplificações para o Presente Trabalho

A interface considera apenas relacionamentos binários. As relações criadas a partir da especificação E-R estarão, pelo menos, na Primeira Forma Normal. Atributos chave compostos não são considerados nesta versão do REVER.

Este trabalho se restringirá ao manuseio de árvores homogêneas (compostas do mesmo tipo de nó). Árvores de relacionamentos serão igualmente ignoradas, embora se aceite árvores de agregados.

Para minimizar os problemas advindos de manutenção de integridade para dependência de inclusão, temos:

- Uma dada entidade só pode estar associada a no máximo um relacionamento total (ou seja, só pode participar de no máximo uma dependência de inclusão), figura 6.

- Agregados que participem de árvores não podem englobar relacionamentos totais (a participação em uma hierarquia já implica uma dependência de inclusão), figura 7.

- Agregados que englobem relacionamentos totais não podem por sua vez estar associados a outros elementos através de relacionamentos totais, figura 8.

- Folhas de uma árvore não podem estar associadas por relacionamentos totais, figura 9.
- Nenhum elemento pode pertencer a mais de uma árvore, figura 10.

3 Especificação da Interface

A interface tem dois tipos de operação: operações sobre o esquema e sobre os dados. Em ambos os casos, são permitidas consultas e atualizações. Saliente-se que é possível modificar o esquema mesmo após criado o banco de dados. No estágio atual do REVER, as modificações permitidas são ainda limitadas (por exemplo, não é possível mudar o tipo de um atributo uma vez criada a relação).

As operações de geração e consulta permitidas sobre o esquema do modelo E-R estendido aqui utilizado, nos leva à discussão da linguagem de consulta apropriada.

Inúmeras linguagens de consulta E-R foram propostas sobre uma base de dados relacional (por ex. [MALH86]). Todas tentam aliar a riqueza semântica do modelo E-R ao formalismo relacional.

Chen [CHEN83b], por exemplo, baseia-se no cálculo relacional para mostrar que linguagens de consulta para o modelo E-R podem ser 'completas', que é um requisito fundamental das linguagens relacionais. Fornece duas definições formais: linguagens E-R completas e completas simplificadas.

Segundo [DATE77], uma linguagem completa *"significa para o usuário que, se uma informação desejada está num banco de dados, então ela pode ser recuperada através de uma simples consulta"*. As linguagens completas, entretanto, não possuem função de agregação e capacidade aritmética implícitas, que são propriedades desejadas.

A interface REVER se comunica com o usuário através de menus, de forma interativa. Quaisquer operações (consultas ou atualizações) sobre dados são especificadas pelo usuário em termos do projeto E-R. Desta forma, o sistema precisa primeiro verificar que elemento(s) do esquema relacional corresponde(m) a um objeto E-R para depois efetuar a operação desejada sobre as relações correspondentes (ocorrências do esquema E-R) e mostrar o resultado ao usuário.

Existem vários problemas no que diz respeito a operação sobre ocorrências de objetos E-R. Por exemplo, uma consulta a um objeto complexo E-R pode corresponder a consultas a várias relações. Optou-se, neste caso, por informar ao usuário sobre o esquema consultado para que ele possa optar sobre que dados deseja realmente ver. Assim, por exemplo, se consultada uma generalização, o usuário recebe como resposta quais os nós componentes, podendo então escolher de qual nó examinar as ocorrências.

Da mesma forma, atualizações sobre um objeto podem ser propagadas a outros (razão pela qual se optou pelas restrições mostadas na seção anterior). A seção 5 discute alguns problemas neste sentido. Uma maior discussão sobre o tópico e soluções apresentadas pode ser encontrada em [NOGU88], escapando ao âmbito deste artigo por sua complexidade.

As operações de atualização sobre o esquema E-R são, na verdade, de dois tipos: operações permitidas sobre um esquema vazio e operações permitidas quando o banco de dados já contém dados. Muitas atualizações permitidas sobre o esquema de um banco de dados vazio deixam de sê-lo a partir do momento em que dados são carregados. Isto parte do pressuposto que a definição do esquema E-R pode sofrer alteração (por exemplo, atributos de uma entidade, restrições de integridade ou interrelacionamentos entre objetos), enquanto que modificações de esquema se tornam mais difíceis após a geração de dados. Assim, as únicas alterações de esquema permitidas após a carga do banco de dados são as que implicam geração ou remoção de relações inteiras.

A formação do esquema relacional é processada como um conjunto de listas encadeadas. Apenas quando o usuário se considera satisfeito com o projeto do esquema E-R é que são gerados os comandos para criação das relações correspondentes.

4 Implementação da Definição do Esquema

O esquema relacional é armazenado em um arquivo com 4 registros diferentes, ordenados via um ÍNDICE. Cada conjunto de registros é identificado por um campo comum chamado TIPO. Os formatos de registro se encontram a seguir, onde IND é um identificador para o índice, T é um identificador de tipo, CAD(ATR) é um apontador para uma cadeia de enumeração de atributos do objeto, CAD(REL) é um apontador para uma cadeia que armazenará todos os índices dos relacionamentos aos quais o objeto está ligado e CAD(COMPLEX) é um apontador para uma cadeia que conterá todos os índices dos objetos complexos dos quais o objeto participa.

Ind	E	Nome	Chave	Cad(atr)	Cad(rel)	Cad(complex)

O registro referente a relacionamentos abaixo engloba também a indicação se é um relacionamento TOTAL/PARCIAL, a DIREÇÃO para relacionamentos totais e os índices IE1 e IE2 dos objetos que liga.

Ind	R	Nome	Chave	T/P	Dir	IE1	IE2	Cad(atr)	Cad(complex)

O registro de agregações abaixo indica o relacionamento que identifica a agregação - IREL.

Ind	AG	Nome	IRel	Cad(rel)	Cad(complex)

As estruturas de árvores tem registros semelhantes para armazenar, nó a nó, os seguintes campos: NOME da árvore (generalização/hierarquia), IND(PAI) - índice do nó-pai; CAD(IND-FILHOS) - um apontador para uma cadeia que engloba os índices dos nós-filho do nó em questão.

Ind	T	Nome	Ind(pai)	Cad(Ind_filhos)

Com as informações constantes desses registros é possível navegar em todos os sentidos para a recuperação/alteração de dados. Optou-se por dar às relações geradas os mesmos nomes dos elementos E-R.

5 Problemas de Atualização

São consideradas apenas atualizações de dados do tipo inserção, remoção e modificação de valores não-chave. Qualquer pedido de atualização é suposto como sendo requisitado em cima dos seguintes elementos: a) entidades simples; b) relacionamentos; c) agregados; d) nó de uma árvore (que pode ser de um dos tipos anteriores). De agora em diante, qualquer atualização em um elemento é entendida como mapeada para a relação correspondente.

O caminho de propagação de atualizações é obtido através de navegação dentro da estrutura de dados do esquema, acompanhado, quando necessário, de consultas ao usuário sobre opções a serem tomadas.

Para melhor entendimento do que se segue, serão definidos os seguintes termos:

- Caminho em uma árvore - mesmo conceito utilizado em estruturas de dados do tipo árvore.
- Caminho ancestral de um nó - caminho que sai da raiz de uma árvore até atingir o nó em questão.
- Irmãos de um nó - dado o ancestral (pai) do nó em uma árvore, são os outros filhos daquele pai, no mesmo nível.

Para inserção em nós de árvores, é necessário ao usuário indicar valores para todos os atributos no caminho na árvore ao qual o nó pertence.

Para remoção de nó em generalizações, serão eliminados todos os elementos a ele relacionados no caminho na árvore; para remoção em hierarquias, serão eliminados todos os elementos a ele relacionados no caminho da raiz até o nó em questão, se o elemento a remover não tiver correspondente em seus irmãos na árvore; e serão eliminados todos os elementos da sub-árvore da qual ele é raiz.

Modificações de valores de atributos não acarretam problemas.

Tendo em vista estas definições, e considerando-se as atualizações permitidas, observa-se o seguinte:

- modificações não exigem propagação, restringindo-se ao elemento considerado (pois não se permite mudar conteúdo de chaves).
- as inserções e remoções podem requerer propagação, em casos específicos. A discussão destes casos se encontra em [NOGU88].

Casos de propagação de atualizações:

a) Quando há dependência de inclusão, figura 11:

- se inserir em B e/ou R → inserir em A,
- se excluir de A e/ou R → excluir de B;

b) Em entidades, figura 12:

- se excluir de A/B → excluir de R,
- se incluir em R → incluir em A e B;

c) Em agregações:

- obedece as regras de (a) e (b);

d) Em hierarquias de tipo, figura 13:

- se inserir em EL:
  - Propagação no caminho ancestral:
    - se já existe irmãos com mesma chave do pai, não fazer nada;
    - senão, propagar pelo caminho ancestral até RZ.

- Propagar por um caminho em SA.

- se remover em EL:
  - Propagação no caminho ancestral:
    - se existe irmão com mesma chave, não fazer nada;
    - senão, propagar pelo caminho ancestral, respeitando a existência de irmãos no caminho.
  - Propagar por todos os caminhos necessários em SA.

e) Em generalizações, figura 13:

- se inserir em EL:
  - Propagar pelo caminho ancestral;
  - Propagar para algum caminho em SA.
- se excluir em EL:
  - Propagar pelo caminho ancestral;
  - Propagar para o caminho (de mesma chave) em SA.

6 Considerações Finais

Este artigo apresentou uma interface E-R para o modelo relacional, que está sendo implementada. No momento em que este artigo foi escrito, o manuseio do esquema se encontra pronto, restando associá-lo ao manuseio de ocorrências.

Motro [MOTR80] defende a idéia de que estruturas rígidas que possuem arquitetura estruturada, baseada em registros, facilitam a recuperação de informações, e no entanto, impedem que o BD acompanhe a evolução constante do ambiente que retrata. Propõe como solução uma arquitetura binária baseada em coleções desestruturadas de fatos que podem ser consultados através de uma linguagem 'padrão' fundamentada em predicados. A recuperação de um fato, definido da mesma forma que um agregado - duas entidades associadas via um relacionamento binário, é realizada através de dois mecanismos: pesquisa, exame da vizinhança do fato; e tentativa, consulta seguida de falha seguida de consulta.

O armazenamento do esquema via listas ligadas impõe um caminho a ser percorrido para localizar o dado, o que implica em problemas com o tempo de acesso. Outro ponto crucial é a necessidade de se trabalhar num ambiente heterogêneo, formado pelos dois modelos envolvidos, e extrair os aspectos mais favoráveis e fundamentais de cada um sem perdas profundas no outro lado.

Uma deficiência da interface aqui proposta é que implica num mapeamento único para implementação do banco de dados. Além disto, não há preocupação em otimizar o BD relacional gerado.

Como é finalidade deste trabalho usufruir da semântica do modelo E-R para proporcionar maior compreensão e documentação do sistema, e como as fases de planejamento e análise do desenvolvimento do sistema levam a representações extensas do modelo que requerem sucessivos refinamentos, acreditamos que um futuro melhoramento seria a implementação dos conceitos descritos no *"Modelo Entidade por Agrupamento"* de Feldman e Miller [FELD86].

Acredita-se que após a aplicação desse modelo por agrupamento de entidades, outro melhoramento seria a implementação de uma linguagem gráfica para a interface, pois um dos objetivos do agrupamento é reduzir o número de entidades num contexto (tela) para facilitar sua compreensão (representação visual).

O que julgamos mais adequado seria a existência de 3 linguagens a serem suportadas pela interface: a. gráfica - dirigida aos usuários não-técnicos; b. orientada por menus - para atender usuários mais familiarizados com a área de BD, capazes de estabelecer restrições de integridade, p.ex.; c. formal, semelhante no formalismo do cálculo ou álgebra relacional - para servir aos profissionais que administrem o banco de dados.

A inexistência de formalismos para o modelo E-R nos obriga a refletir sobre as questões de normalização e otimização das relações geradas pelo mapeamento imposto. O trabalho de Chung [CHUN83] define a noção de Relações E-R e introduz formas normais E-R para decompor o diagrama - esquema E-R até normalizá-lo e obter uma correspondência de 1:1 com o esquema relacional.

Agradecimentos A primeira autora desenvolveu este trabalho como bolsista da CAPES e posteriormente da IBM - Brasil.

Referências

[ATZE83] Atzeni, P. e Chen, P.P. *Completeness of Query Languages for the Entity-Relationship Model* in [CHEN83a].

[BRAE85] Braegger,R.P.; Dudler,A.M.;Rebsamen,J. e Zehnder,C.A. *Gambit: An Interactive Database Design Tool for Data Structures, Integrity Constraints, and Transactions.* IEEE Transactions on Software Engineering, SE-11(7), 1985.

[CHEN76] Chen, P.P. *The Entity-Relationship Model: Toward a Unified View of Data.* ACM TODS 1(1), 1976.

[CHEN83a] Chen,P.P. *Entity-Relationship Approach to Information Modeling and Analysis.* North-Holland, 1983.

[CHEN83b] Chen, P.P. *A Preliminary Framework for Entity-Relationship Models* in [CHEN83a].

[CODD79] Codd, E.F. *Extending the Database Relational Model to Capture More Meaning.* ACM Transations on Database Systems 4(4), 1979.

[CHUN83] Chung,I.; Nakamura,F. e Chen,P.P. *A Decomposition of Relations Using the Entity-Relationship Approach* in [CHEN83a].

[DATE77] Date, C.J. *An Introduction to Data Base Systems.* 2nd Ed., Addison Wesley, 1977.

[DELG86] Delgado, A.L.N. e Magalhães, L.P. *Assimilando a Semântica de PAC através de um Modelo de Dados.* Anais do VI Congresso da SBC, Recife, 1986.

[ELMA83] Elmasri,R. e Wiederhold,G. *GORDAS: A Formal High-Level Query Language for the Entity-Relationship Model* in [CHEN83a].

[ELMA85] Elmasri,R., Weeldreyer,J. e Hevner,A. *The Category Concept: An Extension to the Entity-Relationship Model.* North-Holland, 1985.

[FURT87] Furtado,A.L., Casanova,M.A. e Tucherman,L. *The CHRIS Consultant.* 6th International Conference on Entity-Relationship Approach, 1987.

[FELD86] Feldman, P. e Miller, D. *Entity Model Clustering: Structuring a Data Model by Abstraction.* The Computer Journal 29(1), 1986.

[MAIE85] Maier, D. *Theory of Relational Databases.* Computer Science Press, 1985.

[MALH86] Malhotra,A., Tsalalikhin,Y., Markowitz,H., Pazel,D. e Burns,L.M. *Implementing an Entity-Relationship Language on a Relational Data Base.* Relatório Técnico IBM, 1986.

[MARK83] Markowitz,H.M., Malhotra,A. e Pazel,D.P. *The ER and EAS Formalisms for System Modeling, and the EAS-E Language* in [CHEN83a].

[MOTR84] Motro,A. *Browsing in a Loosely Structured Database.* ACM SIGMOD, 1984.

[NAVA83] Navathe,S. e Cheng,A. *Database Schema Mapping* in [CHEN83a].

[PARE86] Parent, C. e Spaccapietra, S. *An Algebra for a General Entity-Relationship Model.* IEEE Transations on Software Engineering SE-11(7), 1986.

[NOGU88] Nogueira, Mônica de Lima - Tese de Mestrado, em preparação, 1988.

[ROUS84] Roussopoulos,N. e Yeh,R.T. *An Adaptable Methodology for Database Design.* Computer, May, 1984.

[SCHE80] Scheuermann,P., Schiffner,G. e Weber,H. *Abstraction Capabilities and Invariant Properties Modelling Within the Entity-Relationship Approach.* North-Holland, 1980.

[SHAV81] Shave, M.J.R. *Entities, Functions and Binary Relations: Steps to a Conceptual Schema.* The Computer Journal 24 (1), 1981.

[SMIT77] Smith,J.M. e Smith,D.C.P. *Database Abstractions: Aggregation and Generalization.* ACM Transations on Database Systems 2(2), 1977.

[TABO83] Taborier, Y. e Nànci, D. *The Occurence Structure Concept: An Approach to Structural Integrity Constraints in the Entity-Relationship Model* in [CHEN83a].

[TEOR86] Teorey,T.J., Yang,D. e Fry,J.P. *A Logical Design Methodology for Relational Databases Using the Extended Entity-Relationship Model.* Computing Surveys 28(2), 1986.

[WAGN87] Wagner, C.F. e Medeiros, C.B. *Um Modelo Binário Estendido para Entidade-Relacionamento.* Anais do VII Congresso da SBC, Salvador, 1987.

[WELD80] Weldon, J.L. *Using Data Base Abstractions for Logical Design.* The Computer Journal 23(1) 1980.

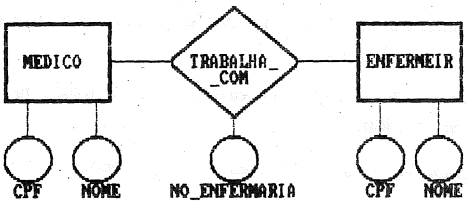

FIG. 1

FIG. 2

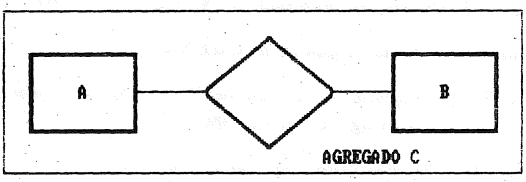

FIG. 3: AGREGACAO

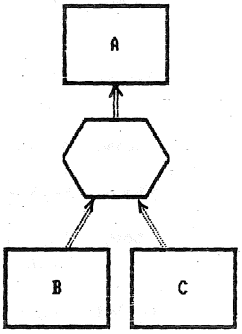

FIG.4: GENERALIZACAO

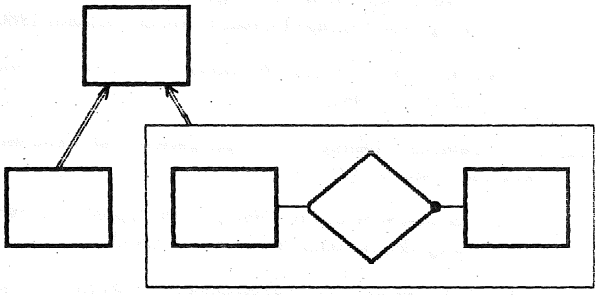

FIG. 7

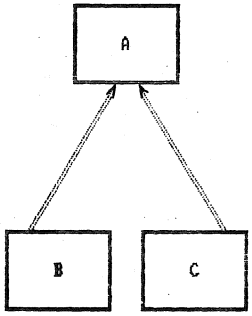

FIG.5:HIERARQUIA DE TIPOS

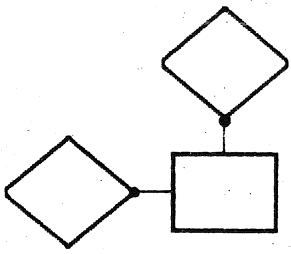

FIG.6(a)

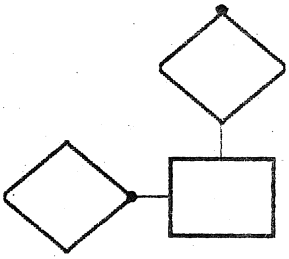

FIG.6(b)

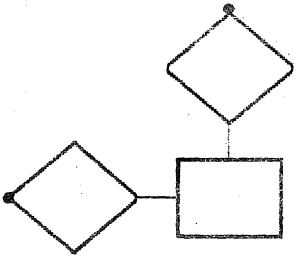

FIG.6(c)

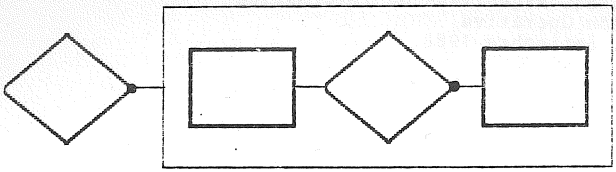

FIG. 8

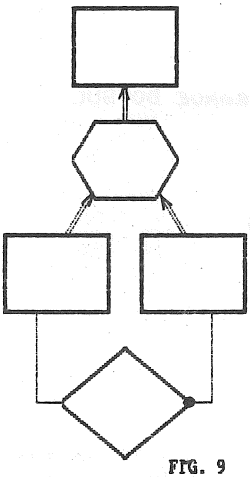

FIG. 9

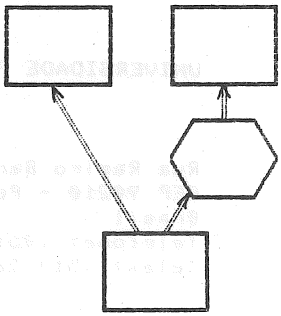

FIG. 10

FIG. 11

FIG. 12

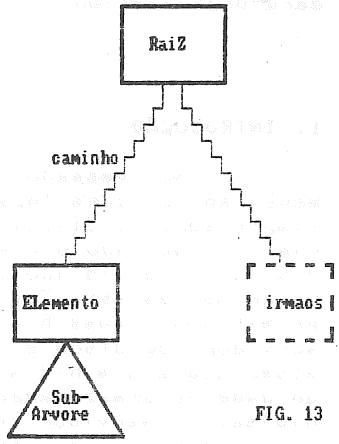

FIG. 13