Projeto de Dados em Bancos de Dados Distribuídos[1]

Eduardo José Soler Mesquita[2]

PriceWaterhouseCoopers
International Consulting Division
São Paulo - SP

Marcelo Finger[3]

Departamento de Ciência da Computação
Instituto de Matemática e Estatística
Universidade de São Paulo

Resumo

Este trabalho propõe o projeto de distribuição de dados no nível conceitual, a partir da utilização de um modelo semântico, o modelo Entidade-Relacionamento.

Nos modelos de dados existentes na literatura, o projeto de distribuição de dados é realizado no nível lógico, o que torna o esquema conceitual obsoleto a partir da geração do esquema lógico. Nestes casos, o esquema de fragmentação é construído sobre o esquema lógico e o modelo conceitual torna-se uma documentação desatualizada do projeto.

Nosso trabalho propõe utilizar a estrutura do modelo conceitual como base das fragmentações realizadas no projeto de distribuição de dados. Tal informação está ausente no Modelo Relacional, cuja estrutura é totalmente chata. Desta forma, pretendemos incorporar o modelo conceitual à documentação ativa do projeto.

Abstract

This work proposes a method for designing data distribution for distributed databases from the Conceptual Level, using the Entity-Relationship semantic model.

In the literature, the design of data distribution is normally performed at the level of relations, which makes its associated conceptual model obsolete once the distribution schema is generated. Fragmentation, which is the usual process of breaking and distributing data, is applied only at the level of relations, rendering the conceptual level schema out-of-date.

By bringing the fragmentation concept up to the Conceptual Level, our work intends to make the conceptual level a living entity throughout the distribution design process. We propose to use the strucutral information available at the Conceptual Level (but absent in the flat relational model) as the basis for this fragmentation process.

1. Introdução

O projeto de um banco de dados tradicionalmente se divide em três partes [EN89][EN89][KS95][Set86]: o *projeto conceitual*, que produz uma representação em alto nível da realidade do Universo de Discurso; o *projeto lógico*, que traduz esta representação em especificações que podem ser processadas por um sistema de computação; e *projeto físico*, que determina as estruturas de armazenamento físico e os mecanismos para manipulá-las.

No projeto de um banco de dados distribuídos, tradicionalmente, começa-se a tratar da *fragmentação* de um esquema global e *alocação* dos dados fragmentados a partir do nível lógico [CP84][OV91]. Neste contexto, o projeto conceitual se presta apenas à geração do esquema global no nível lógico. Como conseqüência, o modelo obtido no nível conceitual torna-se desatualizado e não mais reflete a evolução do sistema a partir da fragmentação dos dados. E, portanto, este modelo não pode mais ser usado como documentação viva do projeto.

---

[1] Agradecemos aos comentários de Alberto H. F. Laender a este trabalho.
[2] Trabalho apoiado pela FAPESP (Proc. 95/06586-7)
[3] Trabalho parcialmente apoiado pelo CNPq, bolsa (PQ) 300597/95-9.

O objetivo deste trabalho é trazer o conceito de *fragmentação de dados* para o nível conceitual de projeto de dados e, desta forma, possibilitar o projeto de distribuição de dados neste nível. Pretendemos com isso fazer com que o modelo conceitual:

- seja parte da documentação "viva" do projeto de distribuição de dados;
- por ser mais abstrato, traga o projeto mais próximo do projetista e mais distante da máquina;
- seja mais independente de plataforma, mais fácil de ser entendido e portado.

A principal vantagem desta "elevação no nível" do projeto de dados reside no fato do modelo conceitual ter maior capacidade de refletir a *estrutura* do Universo de Discurso que os demais níveis – nos métodos propostos na literatura, o nível lógico é o Modelo Relacional, cuja estrutura é chata. Esta informação sobre a estrutura dos dados, como iremos demonstrar, tem papel central no nosso método de projeto de distribuição de dados.

Neste trabalho iremos utilizar o Modelo Entidade-Relacionamento (MER) de Chen [Che76] no nível conceitual. Existem várias extensões e variações notacionais do modelo original, e neste trabalho vamos nos ater às extensões, notação gráfica e notação de restrições de cardinalidade conforme apresentadas em [BCN92].

Os resultados aqui apresentados aparecem na forma estendida em [Mes98].

1.1 Comparações com a literatura

Modelos conceituais para o projeto de dados para o projeto de dados vêm sendo desenvolvido desde a proposta original de Chen [Che76]. Mais recentemente, procurou-se dar também a este modelo a possibilidade de manipular dados através de linguagens de consulta e manipulação (e.g. [Mes98][SP85][BCN92]), e existem implementações em que o projeto de dados (centralizado) e todos os acessos ao banco de dados são feitos apenas neste nível, como no projeto TEMPORA [LMS+91].

O método predominante na literatura para projeto de bancos de dados parte da fragmentação de relações no modelo relacional [CP84][OV91], operando transformações no esquema e instanciação no modelos chamadas de fragmentações *horizontal*, *vertical*, *horizontal derivada* e *mista*. Estas transformações são baseadas em *consultas* e as instanciações dos fragmentos devem obedecer a restrições que garantem sua corretude: *completude*, *reconstrutibilidade* e *disjunção*.

Similarmente, nosso trabalho irá transpor as transformações por fragmentação horizontal, vertical e mista para entidades do MER baseado em consultas ao MER. No entanto, precisaremos estender este conceito para lidar com hierarquias de generalização, fragmentação derivada de relacionamentos, e a fragmentação derivada de classes de entidades através de caminhos no Diagrama ER (DER). Preservaremos os critérios de corretude em todos os casos, garantindo completude, reconstrutibilidade e disjunção no nível conceitual.

Um trabalho na literatura que trata da transformação de DERs é [BCN92]. Neste, o objetivo das transformações é o projeto do DER por meio de *refinamentos sucessivos* e as transformações são agrupadas em métodos de refinamento: *top-down*, *bottom-up* e *misto*. Estas transformações também devem obedecer a condições: *correção* e *completude* (que apesar da coincidência nominal, é um conceito distinto da completude da fragmentação). Diferentemente do nosso caso, as transformações dos refinamentos não são guiadas por consultas às instanciações do DER.

Para podermos realizar fragmentações no MER necessitamos ter bem definidas as noções de instanciação e consultas ao DER; a Seção 2 cobre este requisito. As fragmentações primárias de elementos do DER são tratadas na Seção 3. As fragmentações derivadas são

detalhadas na Seção 4. Discutimos a geração das imagens físicas dos fragmentos na Seção 5, e por fim, de concluímos nosso trabalho na Seção 6.

2. Pré-Requisitos à Fragmentação no MER

2.1 Instanciação no MER

Cada elemento do MER será semanticamente instanciado através da função *I* que representa uma instância de todo o diagrama ER. Seja $ID = \{id_1, id_2, ..., id_n\}$ um conjunto contável de *identificadores* e $DER = \{E, A, R, H, C\}$, um diagrama formado por todos os elementos pertencentes ao MER, onde: *E* é um conjunto de classes de entidades; *A* é um conjunto de atributos; *R* é um conjunto de classes de relacionamentos; *H* é um conjunto de hierarquias de generalização; e *C* é uma função de configuração do diagrama de acordo com:

- para cada entidade $e \in E$, $C(e) = \langle A_d, A_{nd} \rangle$, onde $A_d \subseteq A$ é conjunto de atributos determinantes de *e*, e $A_{nd} \subseteq A$ o conjunto de atributos não determinantes de *e*.
- para cada relacionamento $r \in R$, $C(r) = \langle e_1, e_2, c_{12}, c_{21} \rangle$, onde c_{12} e c_{21} correspondem às restrições de cardinalidade.
- para cada hierarquia de generalização $h \in H$, $C(h) = \langle e, \{e_i\}, tipo \rangle$, onde $e \in E$ corresponde à classe de entidades "pai" da hierarquia, $\{e_i\} \subseteq R$ corresponde ao conjunto de "filhos" e *tipo* indica se a hierarquia é <u>total</u> (representado pelo trinângulo preenchido no DER) ou <u>parcial</u> (triângulo vazio).

Se *I* é uma instância de um diagrama ER, então, para todo $e \in E$, $a \in C(e)$, e $r \in R$:

- $I(e) \subseteq ID$, ou seja, a instância de uma classe de entidades $I(e)$ é um conjunto finito de identificadores (ou entidades);
- $I(a)$: $I(e) \rightarrow dom(a)$, ou seja, a cada atributo *a* está associado a um certo domínio $dom(a)$ e a instanciação de um atributo é uma função sobre um conjunto de identificadores em seu domínio.
- $I(r) \subseteq ID^n$ é um conjunto de *n-uplas* $\langle id_1, ..., id_n \rangle$, onde *r* é uma classe de relacionamentos entre $n \geq 2$ entidades $e_1,..., e_n$ tal que $id_1 \in I(e_1), ..., id_n \in I(e_n)$, respeitando-se as restrições de cardinalidade.

A instanciação de uma hierarquia de generalização que tem *e* como classe de entidades "pai" e o conjunto $\{e_i\}$ como classes de entidades "filhos", sendo *I* uma instância do diagrama ER obedece às seguintes restrições:

$$I(e_i) \subseteq I(e), \text{ para } i=1, 2, ..., n \quad \text{e} \quad I(e_i) \cap I(e_j) = \varnothing, \text{ para } i \neq j.$$

Segue que $I(e) \supseteq I(e_1) \cup ... \cup I(e_n)$. No caso de hierarquia total, $I(e) = I(e_1) \cup ... \cup I(e_n)$.

2.2 Linguagem de Consulta ao MER

A fragmentação de um banco de dados requer que consultas sejam efetuadas sobre o seu esquema conceitual. A álgebra relacional é a linguagem utilizada para a fragmentação das relações que compõem um esquema relacional de dados. No modelo entidade-relacionamento, várias linguagem de consulta e manipulação de dados foram propostas, como em [SP85],

[Set86],[Mes98][MNP+91]. Baseando-nos nestas linguagens, optamos por utilizar uma linguagem de consulta ao qual denominamos *Linguagem de Definição de Caminhos*.

Esta linguagem utiliza-se da estrutura do grafo que representa um DER para a denotar caminhos neste grafo. Para a apresentarmos a especificação desta linguagem, utilizaremos o diagrama apresentado na Figura 1 como exemplo.

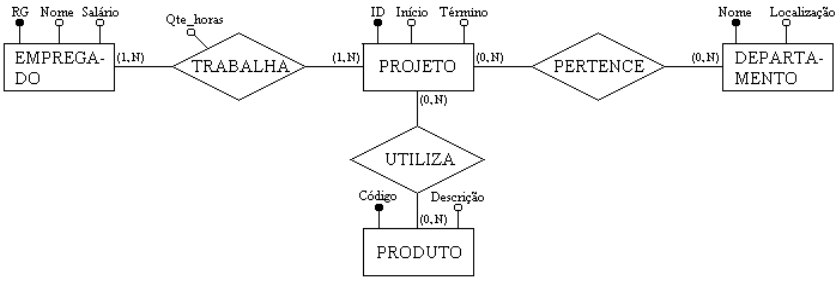

Figura 1: Diagrama ER

Um caminho no grafo pode ser classificado segundo o número de classes de entidades, ou hierarquias, nele contidos. Os nós que compõem estes caminhos podem conter restrições, que podem ser vazias ou não. Restrições são composições lógicas formadas por predicados sobre os atributos, sejam eles pertencentes a classes de entidades ou de relacionamentos.

Sejam $e_i \in E$ uma classe de entidades, $R_{ij} \in R$ um relacionamento que conecta e_i com e_j, e sejam r_i uma restição sobre os atributos da classe de entidades e_i e r_{ij} uma restrição sobre os atributos do relacionamento R_{ij}. Desta forma, definimos recursivamente os caminhos:

- $e_i.r_i$ é um caminho (unário);
- $e_i.r_i \quad R_{ij}.r_{ij} \quad e_j.r_j$ é um caminho (binário);
- Se $e_i.r_i \ldots e_k.r_k$ é um caminho, então $e_i.r_i \ldots e_k.r_k \quad R_{kj}.r_{kj} \quad e_j.r_j$ também é um caminho.

Com base no diagrama da Figura 1, representam exemplos de caminhos:

- *EMPREGADO.Salário > 500,00* (restrição simples) e *EMPREGADO.{Salário > 500,00* and *Salário < 1.000,00}* (restrição composta pelo operador lógico and).
- *EMPREGADO.Salário > 500,00 TRABALHA.Qte_horas ≥ 10 PROJETO.Id = 5* .
- *EMPREGADO.Salário > 500,00 TRABALHA PROJETO PERTENCE DEPARTAMENTO.Nome = "R.H."*. Note que as restrições a entidades e relacionamentos podem ser omitidas quando são VERDADEIRO.

Caminhos podem ser combinados com operadores lógicos, de forma a gerarmos uma linguagem de consulta sobre o diagrama ER, que não detalharemos aqui; ver [MNP+91].

3. Fragmentação Primária do Diagrama ER

Iniciamos descrevendo as diversas fragmentações possíveis de classes de entidades e hierarquias de generalização. A fragmentação de um diagrama ER completo, ou seja, com relacionamentos entre diversas entidades, é tratada no próxima seção.

A idéia principal da geração da fragmentação do diagrama entidade-relacionamento é a aplicação de consultas sobre seus elementos, utilizando-se da linguagem de definição de caminhos apresentada na Seção 2.2. São três as propriedades que garantem a corretude de uma fragmentação no MER, assim como no modelo relacional [CP84]:

- *Completude:* cada elemento da instanciação do DER original deve estar presente na instanciação de algum dos fragmentos.

- *Reconstrução:* o DER original deve ser reconstrutível a partir dos fragmentos.
- *Disjunção:* cada elemento da instanciação do DER original deve estar presente em apenas um fragmento. Esta condição é relaxada na fragmentação vertical.

Um aspecto importante quando tratamos de fragmentação de um diagrama ER é a associação entre o diagrama original e o diagrama fragmentado. Tal associação permite a reconstrução do diagrama original a partir do diagrama fragmentado. Para que esta reconstrução seja possível, introduziremos as operação de união e agregação.

3.1 União e Agregação

Seja $e \in E$ uma classe de entidades com $I(e) = \{id_1,..., id_n\}$. Após uma fragmentação horizontal aplicada sobre e, foram produzidos os seguintes fragmentos com seus respectivos conjuntos de identificadores: e_1 com $I(e_1) = \{id_{11},..., id_{1a}\},\ldots, e_k$ com $I(e_k) = \{id_{k1},..., id_{kz}\}$. A função $Un(e_1,\ldots, e_k)$ é definida como sendo o conjunto união entre as instâncias de cada um dos fragmentos, ou seja,

$$I(Un(e_1,\ldots, e_k)) = I(e_1) \cup ... \cup I(E_t)\} = \{\{id_{11},..., id_{1a},..., \{id_{k1},..., \}.$$

Se há completude na fragmentação, então $I(Un(e_1,\ldots, e_k)) = I(e) = \{id_1, id_2, ..., id_n\}$.

No caso da fragmentação vertical, os fragmentos resultantes não possuem o mesmo esquema do elemento inicial, mas possuem seus esquemas individuais como subconjuntos do esquema do elemento fragmentado, além dos mesmos identificadores que o instanciavam. Seja E uma classe de entidades (hierarquia) com $I(e) = \{id_1, id_2, ..., id_n\}$ seu conjunto finito de identificadores (instâncias). Após uma fragmentação vertical aplicada sobre e, foram produzidos os fragmentos $e_1, ..., e_m$ com $I(e_1) = I(e_2) = ... = I(e_m) = \{id_1, id_2, ..., id_n\}$.

Admitamos que o conjunto finito dos atributos de e seja $\{k_1, k_2, ..., k_s, a_1, a_2, ..., a_r\}$, sendo k_i os atributos chaves de e para $1 \leq i \leq s$, e a_i os atributos não-chaves de e para $1 \leq i \leq r$. Dessa maneira, o conjunto de atributos dos fragmentos será da forma $E_j = \{k_1, k_2, ..., k_s\} \cup A_j$, sendo $A_j \subseteq \{a_1, a_2, ..., a_r\}$, para $1 \leq j \leq m$.

A função de agregação reconstruirá o elemento fragmentado a partir da geração de uma nova classe de entidades (hierarquia) com seu conjunto de atributos formado através da união entre os subconjuntos de atributos dos fragmentos, além dos atributos chaves comuns a todos os fragmentos, ou seja, uma entidade e instanciada por $I(e) = \{id_1, id_2, ..., id_n\}$ com atributos: $Ag(e_1,\ldots, e_m)= \{k_1, k_2, ..., k_s\} \cup A_1 \cup A_2 \cup ... \cup A_m = \{k_1, k_2, ..., k_s, a_1, a_2, ..., a_r\}$.

3.2 Fragmentação de Classes de Entidades

A fragmentação de entidades é formada por dois tipos básicos: a fragmentação horizontal e a fragmentação vertical. Ambos os tipos de fragmentação produzem novas classes de entidades. Porém, o que as difere da classe de entidades original é a incorporação de restrições no caso da fragmentação horizontal, e o esquema da classe da entidade no caso da fragmentação vertical.

3.3 Fragmentação Horizontal de Classes de Entidades

A fragmentação horizontal de entidades consiste no particionamento do conjunto de identificadores que instanciam uma entidade original em subconjuntos de identificadores que

instanciam os fragmentos. Este particionamento é realizado a partir de consultas geradas sobre o esquema da classe de acordo com algum critério desejado; por exemplo, o geográfico.

Para definirmos a fragmentação horizontal de entidades acrescentamos uma nova propriedade sintática à classe de entidades, o conjunto embutido de restrições. Estas restrições são expressões booleanas de predicados sobre os atributos da classe de entidades, e atuam como um "filtro" para as suas instâncias. Uma classe de entidades em seu estado original possui um conjunto vazio de restrições.

A aplicação de uma consulta de fragmentação sobre uma entidade produz outras entidades com o mesmo esquema da entidade original, diferindo apenas na incorporação de restrições por parte das novas entidades. A incorporação destas restrições é realizada a partir da conjunção booleana (AND) das restrições anteriormente pertencentes à classe de entidades de origem com as novas restrições impostas pela fragmentação propriamente dita.

A Figura 2 exibe um exemplo de fragmentação horizontal segundo um critério geográfico, gerando duas outras classes de entidades, agora fragmentos.

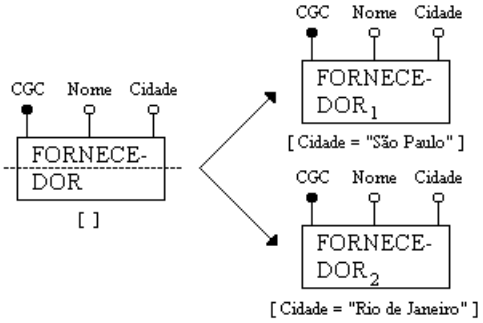

Figura 2: Fragmentação horizontal

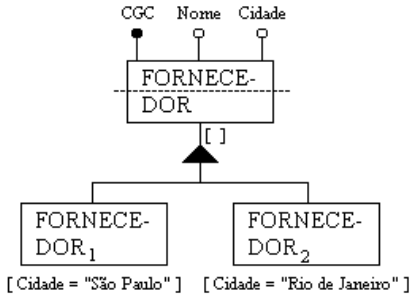

Figura 3: Associação implícita

Às entidades *FORNECEDOR*$_1$ e *FORNECEDOR*$_2$ foram atribuídas as restrições [*Cidade = "São Paulo"*] e [*Cidade = "Rio de Janeiro"*], respectivamente.

A associação existente entre a entidade de origem e as novas entidades neste tipo de fragmentação é representada por uma hierarquia de generalização total implícita, onde a entidade de origem representa a entidade "pai" dessa hierarquia e as novas entidades representam as entidades "filhas", conforme a Figura 3.

Podemos verificar a completude desta fragmentação se e somente se, *"São Paulo"* e *"Rio de Janeiro"* forem os únicos dois valores possíveis para o atributo *Cidade*, caso contrário existiriam localidades que não estariam especificadas nesta fragmentação.

A regra de reconstrução é verificada por construção. A verificação da regra de disjunção é trivial, visto que as restrições [*Cidade = "São Paulo"*] e [*Cidade = "Rio de Janeiro"*] são mutuamente excludentes.

3.3.1 Fragmentação Vertical de Classes de Entidades

A fragmentação vertical de classes de entidades consiste no particionamento do conjunto de atributos que caracterizam uma classe de entidades de origem em subconjuntos de atributos que caracterizam os fragmentos. Cada uma dessas novas classes contém, necessariamente, os atributos chaves (ou determinantes) da classe de entidades de origem.

Os mesmos identificadores que instanciavam a classe de entidades inicial, agora, instanciam cada um dos fragmentos.

Para obtermos a completude da operação, cada atributo deve ser mapeado em um atributo de cada fragmento. A reconstrução da classe de entidades original ocorre por meio de

uma operação de agregação sobre os fragmentos, considerando-se a associação implícita entre eles na forma de um relacionamento do tipo 1:1.

A disjunção entre os fragmentos não é requerida para este tipo de fragmentação, pois a reconstrução da classe de entidades original depende da replicação de seus atributos determinantes em cada um dos fragmentos. Em geral, duplicam-se apenas os atributos determinantes.

A fragmentação é realizada considerando-se que os atributos a serem agrupados têm características desejáveis em comum. Na Figura 3, supomos que salários e fundos de garantia são tratados em separado dos demais atributos. Assim, uma fragmentação possível seria:

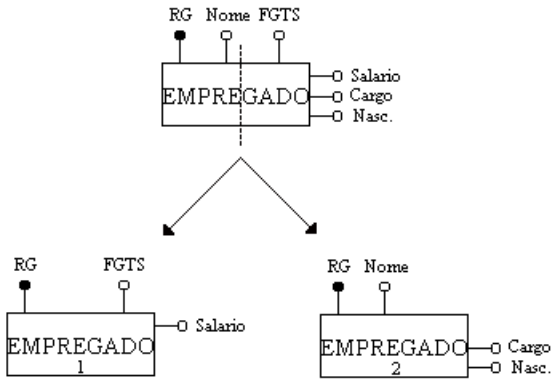

Figura 3: Fragmentação Vertical de Classes de Entidades

A completude é trivial neste tipo de fragmentação, pois cada um dos fragmentos recebe todos os identificadores que instanciam a classe de entidades original.

A reconstrução da classe de entidades original é alcançada através da agregação dos fragmentos através de uma relação implícita que consiste de um relacionamento do tipo 1:1 entre eles, conforme Figura 4. A disjunção não é requerida neste tipo de fragmentação.

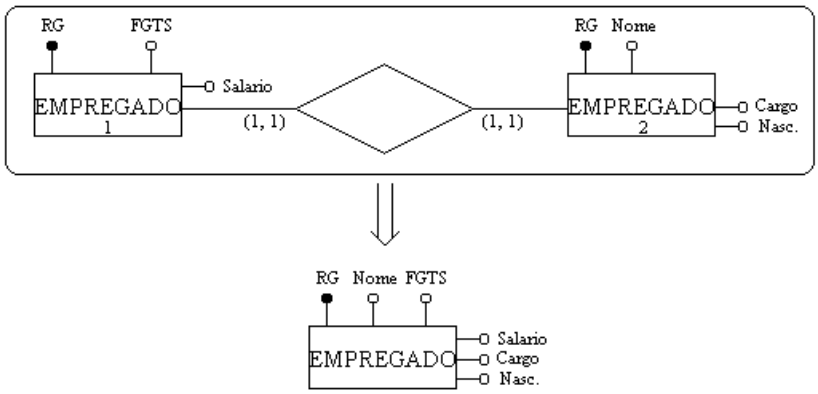

Figura 4: Operação de Agregação dos Fragmentos

3.4 Fragmentação de Hierarquias de Generalização

A fragmentação de hierarquias de generalização também pode ser dividida em dois tipos básicos, a horizontal e a vertical, que produzem novas hierarquias de generalização, e podem ser aplicados tanto à entidade "pai" quanto às "filhas" de uma hierarquia.

3.4.1 Fragmentação Horizontal de Entidades "Pai"

A fragmentação horizontal da entidade "pai" de uma hierarquia de generalização produz novas hierarquias (novas árvores), do mesmo tipo (*total* ou *parcial*). As novas entidades "pai"

de cada uma das novas hierarquias produzidas incorporam as restrições impostas pela fragmentação. Tais restrições são herdadas pelas entidades “filhas” de cada uma das hierarquias fragmentadas.

A instanciação de cada uma das hierarquias fragmentadas é igual a da hierarquia original, limitadas pelas restrições geradas na fragmentação. Um exemplo deste tipo de fragmentação pode ser observado na Figura 5. Note que esta definição é válida para ambos os tipos de hierarquia, por isso representamos a hierarquia por um triângulo semi preenchido.

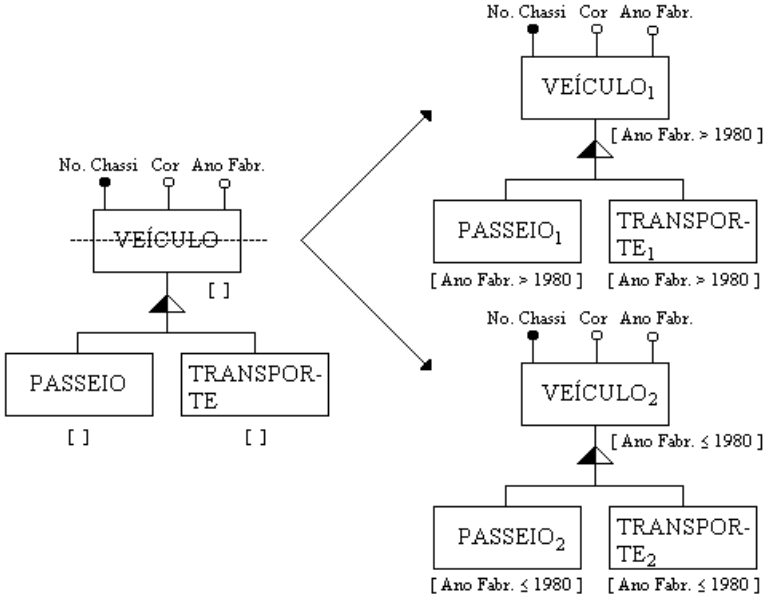

Figura 5: Fragmentação Horizontal de uma Entidade "Pai" em uma Hierarquia

A associação implícita existente entre os diagramas original e fragmentado é obtida da mesma maneira utilizada na seção 3.3. Tal associação está exemplificada na Figura 6.

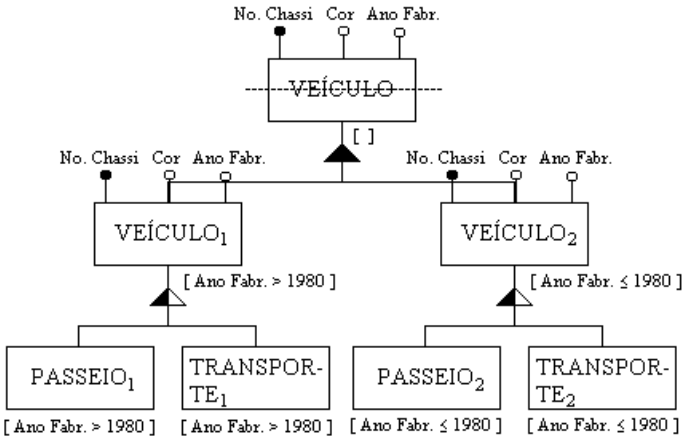

Figura 6: Associação Implícita entre Diagramas (Frag. Horizontal Hierarquias)

A completude da fragmentação é verificada, pois os identificadores que instanciam cada uma das entidades “filhas” *PASSEIO* e *TRANSPORTE* estão representados nos fragmentos “filhos” $PASSEIO_1$ | $PASSEIO_2$ e $TRANSPORTE_1$ | $TRANSPORTE_2$, respectivamente, dado que os conjuntos de restrições das hierarquias fragmentadas são complementares entre si.

A reconstrução da hierarquia original é garantida através de uma operação de união do conjunto de restrições de cada uma das novas hierarquias, a qual reproduzirá a hierarquia original. A disjunção entre os fragmentos é trivial, pois as restrições [*Ano Fabr.* ≤ *1980*] e [*Ano Fabr.* > *1980*] são mutuamente excludentes.

3.4.2 Fragmentação Horizontal de Entidades "Filhas"

A fragmentação horizontal de uma entidade "filha" de uma hierarquia de generalização não produz novas hierarquias, ela apenas acrescenta novas entidades/hierarquias "filhas" à hierarquia original. Um exemplo deste tipo de fragmentação pode ser observado na Figura 7.

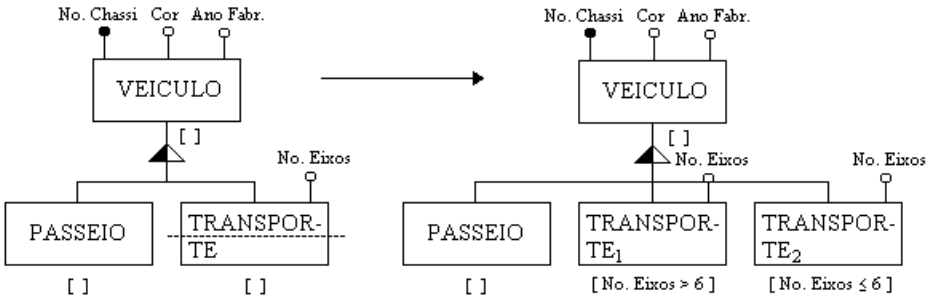

Figura 7: Fragmentação Horizontal de uma Entidade "Filha" em uma Hierarquia

O processo de fragmentação é o mesmo apresentado na Seção 3.3 e as restrições incorporadas pelas novas classes de entidades "filhas" são herdadas por todas as entidades pertencentes à sua sub-árvore, de acordo com o processo apresentado na Seção 3.4.1.

A associação implícita existente entre os diagramas original e fragmentado também é a mesma apresentada na Seção 3.3, conforme a Figura 8.

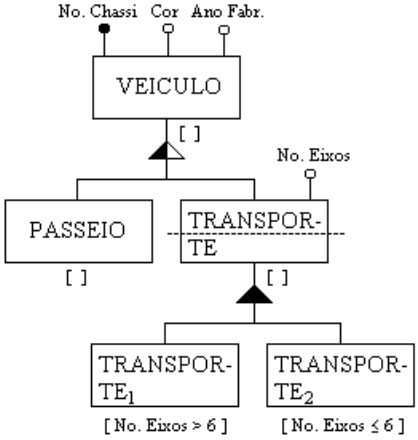

Figura 8: Associação Implícita entre Diagramas (Frag. Horizontal Hierarquias)

Os mesmos identificadores que instanciavam a entidade "filha" original instanciam cada uma das entidade "filhas" fragmentadas, obedecendo às restrições destas entidades.

A corretude da fragmentação é verificada de maneira análoga aos casos anteriores.

3.4.3 Fragmentação Vertical de Hierarquias de Generalização

A fragmentação vertical de uma hierarquia de generalização consiste na fragmentação vertical das classes de entidades que a compõem. Como entidades filhas herdam os atributos de sua entidade pai, a fragmentação vertical de uma entidade filha pode ser vista como o

desdobramento desta em um conjunto de nós “irmãos”, semelhantemente à fragmentação vertical sobre uma classe de entidades. A fragmentação vertical de uma entidade “pai” produz novas hierarquias (novas árvores), do mesmo tipo (*total* ou *parcial*) da hierarquia original. As novas hierarquias possuem a mesma estrutura de classes de entidades da hierarquia original. Seus esquemas são determinados pela fragmentação realizada sobre a entidade “pai” da hierarquia original.Os mesmos identificadores que instanciavam a hierarquia original instanciam cada uma das hierarquias fragmentadas. Por motivos de espaço abreviamos a discussão sobre fragmentação vertical aqui, remetendo a [Mes98] para maiores detalhes.

4. Fragmentação Derivada Estrutural do Diagrama ER

Quando analisamos a fragmentação de elementos que não se encontram “isolados” no diagrama, temos que considerar os efeitos produzidos por uma operação de fragmentação sobre todos os elementos que se relacionam com o elemento fragmentado. Uma vez que a fragmentação de um elemento resulta na fragmentação de outro(s) elemento(s), dizemos que tais fragmentações são derivadas da fragmentação inicial.

A fragmentação derivada é guiada pelas consultas mais executadas pelo usuário. Quando a fragmentação é feita no nível conceitual, a busca de tais consultas deve ser guiada pela *estrutura do banco de dados*. Nos parece óbvio que as consultas mais freqüentes são aquelas que percorrem caminhos no diagrama ER; se não o forem, podemos afirmar que a modelagem não foi feita de forma satisfatória. No modelo relacional, por sua simplicidade, esta estrutura não está disponível; temos aí uma grande vantagem da fragmentação no MER.

4.1 Fragmentação Derivada Primária

Esta seção apresenta a fragmentação automática dos relacionamentos diretamente conectados a elementos fragmentados. Esta fragmentação é uma decorrência obrigatória da fragmentação primária apresentada na Seção 3.

Suponhamos a configuração $e_1\ R\ e_2$, na qual e_1 e e_2 são classes de entidades, e R é um relacionamento de cardinalidades quaisquer que conecta e_1 e e_2. Se e_1 for fragmentada horizontalmente (ou verticalmente) em e_{11}, e_{12}, e_{13}, a fragmentação primária de e_1 provoca a fragmentação automática (derivada primária) de R em R_1, R_2 e R_3, tal que obtemos as configurações $e_{11}\ R_1\ e_2$, $e_{12}\ R_2\ e_2$, $e_{13}\ R_3\ e_2$.

Ou seja, a fragmentação de uma classe de produz novas entidades, que herdam todos os relacionamentos dos quais o elemento original era participante, e portanto forçam a fragmentação destes relacionamentos. A Figura 11 ilustra um exemplo onde ocorre a fragmentação horizontal de uma classe de entidades participante de um relacionamento no diagrama original. Tal fragmentação produz um novo diagrama (distribuído) composto por duas novas classes de entidades (*EMPREGADO*$_1$ e *EMPREGADO*$_2$), com seus respectivos conjuntos de restrições e relacionamentos, além da classe de entidades *DEPARTAMENTO* que agora participa de mais um relacionamento, sem perda de generalidade.

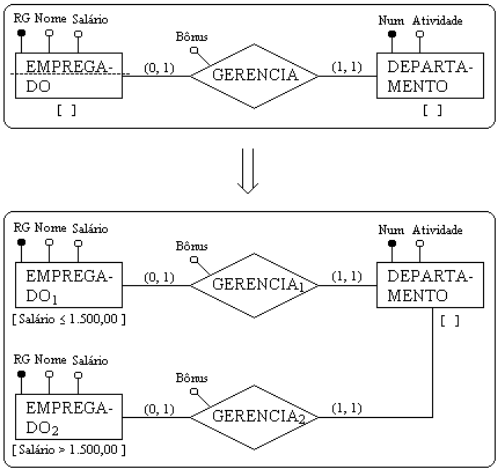

Figura 9: Frag. Horiz. Derivada Primária

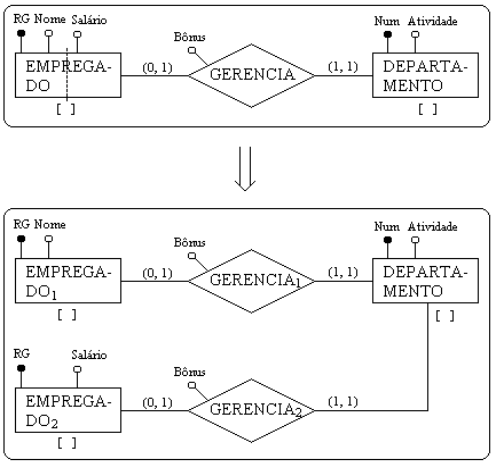

Figura 10: Frag. Vertical Derivada Primária

Analogamente, caso a fragmentação aplicada sobre a entidade *EMPREGADO* fosse vertical, o efeito produzido seria o mesmo para este caso primário de fragmentação derivada, que pode ser visto na Figura 10.

Para os dois casos de fragmentação, existe uma análise a ser realizada sobre as cardinalidades que caracterizam o relacionamento fragmentado.

Caso a cardinalidade máxima entre o relacionamento e a(s) entidade(s) não-fragmentada(s) for igual a *N*, o relacionamento não garante a disjunção entre os fragmentos, ou seja, uma mesma instância pertencente a uma entidade não-fragmentada pode se relacionar com instâncias pertencentes a diferentes fragmentos.

Caso a cardinalidade-máxima entre o relacionamento e a(s) entidade(s) não-fragmentada(s) for igual a *1*, o relacionamento garante a disjunção entre os fragmentos, ou seja, uma mesma instância pertencente a uma entidade não-fragmentada deverá se relacionar com, no máximo, uma instância pertencente a algum dos fragmentos.

Nos exemplos acima, as cardinalidades *(1, 1)* que caracterizam a participação da entidade não-fragmentada *DEPARTAMENTO* no relacionamento *GERENCIA* garantem a disjunção entre os fragmentos, ou seja, uma mesma instância da entidade *DEPARTAMENTO* deve se relacionar com apenas uma instância de um dos fragmentos.

Dependendo do contexto da aplicação de banco de dados, a fragmentação sobre *EMPREGADO* poderia resultar na possibilidade de fragmentação de *DEPARTAMENTO*. Isto representaria a *propagação recursiva* dos efeitos gerados pela fragmentação inicial.

A fragmentação de uma entidade contendo auto-relacionamento gera duas entidades, cada uma com um auto-relacionamento, e mais dois outros relacionamentos entre estas. Uma descrição pormenorizada deste caso pode ser encontrada em [Mes98].

4.2 Fragmentação Derivada Recursiva

Diferentemente da fragmentação derivada primária, a fragmentação derivada recursiva é opcional. Quando analisamos a propagação de uma fragmentação através do grafo que representa um dDER, devemos considerar o contexto da aplicação de banco de dados, a fim de estabelecermos limites de alcance para a propagação de uma fragmentação. Tais limites de alcance se referem à definição do caminho no DER que será afetado pela fragmentação.

A definição destes caminhos é única para cada tipo de aplicação de banco de dados e crucial para o sucesso de um projeto de distribuição de um banco de dados.

A fragmentação derivada em um diagrama ER é definida sobre os conjuntos de restrições dos elementos fragmentados. Por consequência, podemos afirmar que a propagação de uma fragmentação pelo diagrama ER somente faz sentido quando esta fragmentação é horizontal. Como visto na seção 4.1, a fragmentação vertical de um elemento do diagrama somente afeta os relacionamentos nos quais ele participa diretamente.

Reconsideremos o exemplo da Figura 9, supondo agora a fragmentação horizontal da entidade *DEPARTAMENTO*. Esta fragmentação poderia, dependendo do contexto da aplicação, exigir a fragmentação da entidade *EMPREGADO*. Caso esta fragmentação fosse necessária, ela seria uma fragmentação também horizontal e derivada da fragmentação horizontal da entidade *DEPARTAMENTO*.

Conforme visto na Seção 3, a fragmentação horizontal de uma entidade incorpora um conjunto de restrições baseadas nos atributos da entidade fragmentada. Desta maneira, ao fragmentarmos a entidade *DEPARTAMENTO* a partir de um conjunto de restrições baseado no atributo *Atividade*, deduzimos que a fragmentação da entidade *EMPREGADO* também deva ser realizada em relação à divisão departamental. Porém, a entidade *EMPREGADO* não possui nenhum atributo referente ao departamento a que suas instâncias pertencem. Assim, a fragmentação da entidade *EMPREGADO* deve ser realizada através da incorporação de um conjunto de restrições baseado nos atributos da entidade *EMPREGADO*, e não baseado nos seus próprios atributos. Estas restrições serão feitas na forma de uma fórmula de caminho, conforme visto anteriormente.

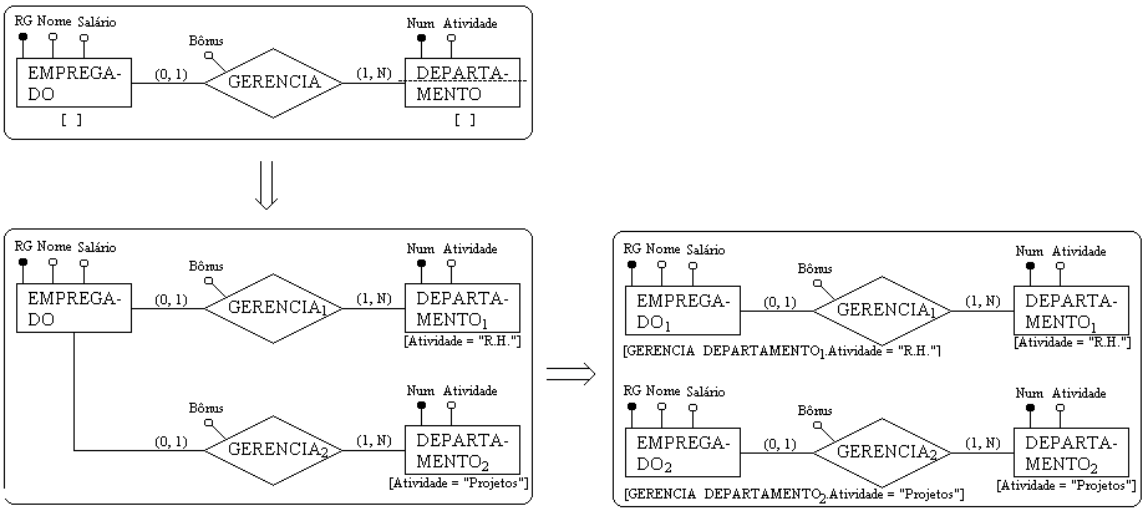

Figura 11: Propagação (1) da Fragmentação pelo Diagrama ER

O conjunto de restrições que caracterizará as novas classes de entidades é composto pela conjunção booleana (AND) entre as restrições anteriores pertencentes à classe de entidades original com as restrições referentes à fragmentação derivada. No exemplo acima, a entidade original *EMPREGADO* possuía um conjunto vazio de restrições, portanto as novas entidades serão caracterizadas somente pelas restrições referentes à fragmentação derivada. No caso, o caminho inicial que unia *DEPARTAMENTO* a *EMPREGADO* será desdobrado em dois, um para cada fragmento de *DEPARTAMENTO*:

EMPREGADO$_1$ *Gerencia DEPARTAMENTO*$_1$*.{Atividade = "RH"}* e
EMPREGADO$_2$ *Gerencia DEPARTAMENTO*$_2$*.{Atividade = "Projetos"}* ,

onde a parte sublinhada é a *restrição* à entidade fragmentada. A Figura 11 ilustra todo o processo de fragmentação, desde a fragmentação inicial aplicada sobre a entidade *DEPARTAMENTO* até a propagação da mesma sobre a entidade *EMPREGADO*.

As cardinalidades, além de determinarem a disjunção, determinam o número de novas entidades e novos relacionamentos que serão criados na fragmentação.

No caso da cardinalidade-máxima, que caracteriza a participação do elemento a sofrer a fragmentação derivada recursiva no relacionamento, ser igual a *1*, a fragmentação ocorre conforme apresentado na Figura 11; apenas dois novos relacionamentos são criados, um para cada fragmento.

No caso desta cardinalidade-máxima ser igual a *N*, em vez de duas entidades e dois relacionamentos, teremos três entidades e quatro novos relacionamentos. A criação de uma terceira entidade se faz necessária para haver a *disjunção* entre os fragmentos. Na Figura 14 estamos supondo a fragmentação primária de *EMPREGADO* e a fragmentação derivada de *DEPARTAMENTO*.

No diagrama inicial, temos que os departamentos podem ser co-gerenciados por vários empregados devido à cardinalidade máxima *N*, e os salários dos gerentes podem ser, a princípio, qualquer. Portanto a fragmentação derivada deve nos fornecer departamentos gerenciados apenas por empregados de salário inferior a 1.500,00, departamentos gerenciados apenas por empregados de salário superior a 1.500,00, e departamentos em que há pelo menos um gerente com salário acima e um gerente com salário abaixo de 1.500,00. Desta forma teremos três entidades. A existência de quatro relacionamentos decorre deste fato. A Figura 14 mostra todas as etapas deste tipo de fragmentação derivada.

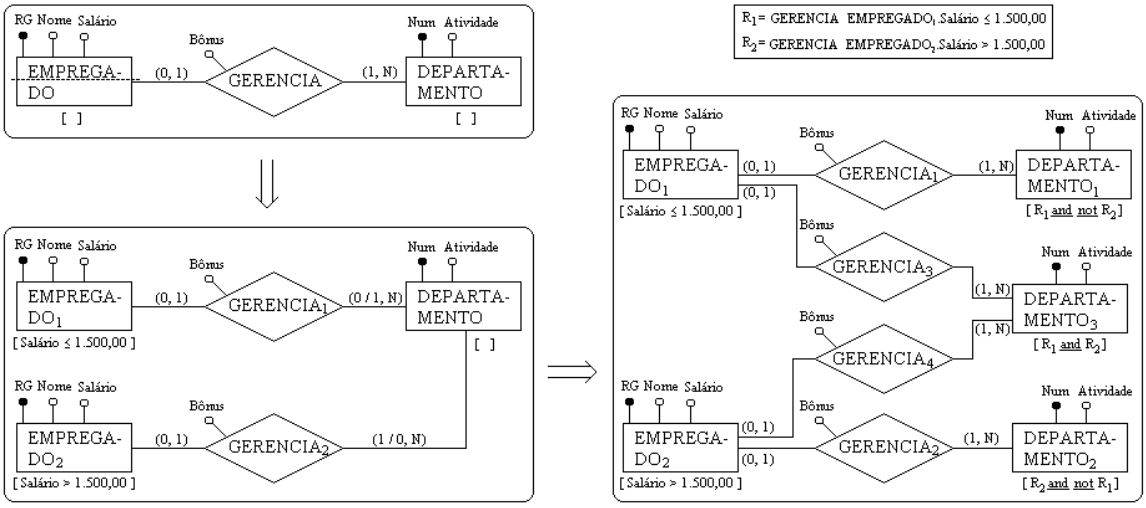

Figura 12: Propagação (2) da Fragmentação pelo Diagrama ER

Suponhamos que *Gerenciado_por* é um "alias" de *Gerencia* (prática comum no MER, e.g. [BCN92][MNP+91]) e:

$$R_1 = DEPARTAMENTO\ Gerenciado\_por\ EMPREGADO_1$$
$$R_2 = DEPARTAMENTO\ Gerenciado\_por\ EMPREGADO_2,$$

então, o fragmento que contem apenas co-gerentes com salários inferiores (superiores) a 1.500,00 é caracterizado pela restrição $R_1\ AND\ NOT\ R_2$ ($R_2\ AND\ NOT\ R_1$) e o fragmento com ambos os tipos de gerentes é caracterizado por $R_1\ AND\ R_2$. Assim, temos garantida tanto a disjunção desta fragmentação derivada, pela mútua exclusão lógica, como a completude, pois uma das restrições sempre é verdadeira. A reconstrutibilidade é garantida por construção.

Em geral, uma fragmentação primária de uma entidade em *N* outras gera, por fragmentação derivada através de relacionamento de cardinalidade máxima ilimitada, 2^N-1 novas entidades e $N.2^{(N-1)}$ novos relacionamentos. Informações semânticas adicionais podem reduzir consideravelmente tais números por garantir que as entidades e relacionamentos gerados na fragmentação derivada serão sempre vazios.

5. Discussão: Imagens físicas de fragmentos

Mostramos como transformar um diagrama ER em outro diagrama, mais complexo que original. No entanto, este diagrama gerado ainda não é o produto final, pois devemos distribuí-lo. Usando a terminologia da fragmentação relacional, precisamos gerar as *imagens físicas.* Definimos imagens físicas no MER como um subconjunto conexo do diagrama ER resultante da fragmentação. A união das imagens físicas deve reconstruir o diagrama final obtido. Em geral, uma imagem física contém no máximo um dos fragmentos de uma entidade fragmentada. Mas essa regra pode ser violada, por exemplo, no caso de haver *replicação* de entidades; replicações, no entanto, devem ser tratadas na fase de alocação dos fragmentos em máquinas hospedeiras, e portanto está fora do escopo deste trabalho.

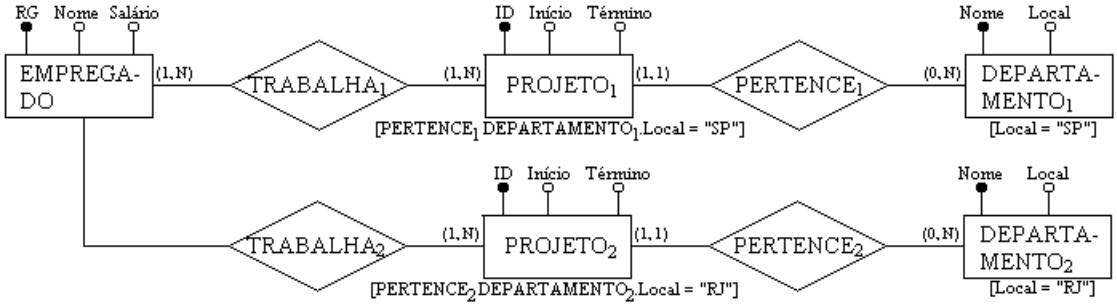

Figura 13: Diagrama ER para Geração da Imagem dos Fragmentos

A Figura 13 mostra um exemplo de um diagrama após o processo de fragmentação. A Figura 16 mostra uma possível geração de duas imagens físicas. Note que há replicação da entidade *EMPREGADO.* Se desejarmos evitar esta replicação poderemos indicar que uma destas entidades é "virtual", talvez representando-a com linha tracejada. Obviamente, em alguma das imagens físicas a entidade não deve ser virtual.

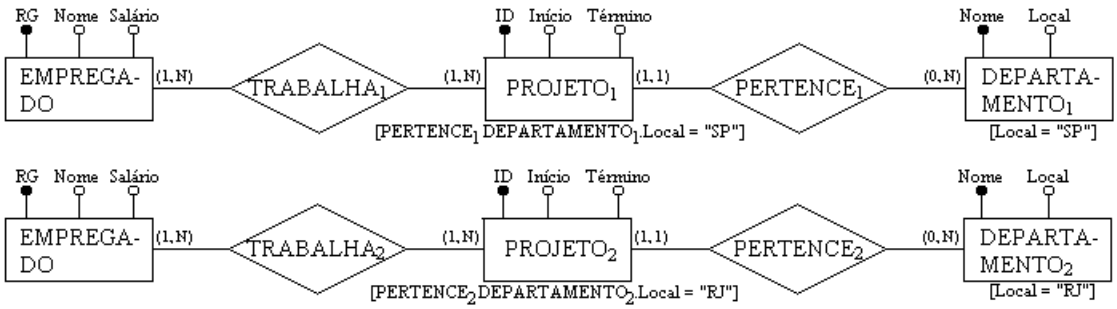

Figura 14: Imagem Física Conexa dos Fragmentos

6. Conclusão

Neste trabalho, apresentamos um método de projeto de distribuição de dados a partir do nível conceitual utilizando o MER. As vantagens deste método são:

1. Tornar o MER parte da documentação viva do projeto, e não mais apenas um item a ser descartado logo após a geração do esquema global.
2. Trazer o projeto da distribuição de dados mais próxima do projetista e mais distante da máquina.
3. Nosso método de projeto pode utilizar as características estruturais do modelo de dados, característica esta que se perde no modelo relacional.

Como trabalhos futuros, podemos citar o problema da alocação das imagens físicas no MER, com um tratamento das replicações; adaptação ao MER das técnicas relacionais de escolha sobre quais fragmentos devem ser criados durante o projeto da fragmentação e alocação de dados. E a construção de uma ferramenta gráfica que auxilie nestes processos. A base para a criação de tal ferramenta é apresentada em [Mes98].

7. Bibliografia

[BCN92] C. Batini, S. Ceri e S. Navathe. *Conceptual Database Design --- An Entity-Relationship Approach*. The Benjamin/Cummings Publishing Company, 1992.

[CP84] S. Ceri, G. Pelagatti. *Distributed Databases - Principles and Systems*. McGraw-Hill, 1984.

[Che76] P. Chen. The Entity-Relatioship model: Toward a unified view of data. ACM TODS, vol. 1, No. 1, 1976.

[EN89] R. Elmasri e S. Navathe. *Fundamentals of Database Systems*. The Benjamin/Cummings Publishing Company, 1989.

[KS95] H. Korth e A. Silberschatz. *Sistema de Bancos de Dados*. Editora McGraw-Hill, Ltda. 2a Edição revisada, 1995.

[LMS+91] P. Loucopoulos, P. J. McBrien, F. Schumacker, B. Theodoulidis, V. Kopanas, and B. Wangler. *Integrating database technology, rule-based systems and temporal reasoning for effective software: the TEMPORA paradigm*. Journal of Information Systems, 1(2), 1991.

[MNP+91] P. J. McBrien, M. Niezette, S. Pantazis, B. Theodoulidis, G. Tziallas, A. H. Seltveit, U.Sundin, and R. Wohed. *The TEMPORA external rule language*. In Proceedings of the Third Nordic Conference on Advanced Information Systems Engineering, vol. 498 of LNCS. Springer-Verlag, 1991.

[Mes98] E. Mesquita. *Projeto de Dados em Bancos de dados Distribuídos*. Dissertação de Mestrado, Departamento de Ciência da Computação, Instituto de Matemática e Estatística da Universidade de São Paulo. Março de 1998.

[Net82] A. M. Neto. *Uma Linguagem de Consulta para o Modelo ER e sua Completude*. Dissertação de Mestrado - IME - USP, 1982.

[OV91] M. T. Özsu e P. Valduriez. *Principles of Distributed Database Systems.* Prentice-Hall, 1991.

[SP85] S. Spaccapietra e C. Parent. *An Algebra for a General Entity-Relationship Model*. In IEEE Transactions on Software Engineering, vol. SE-11, No. 7, 1985.

[Set86] V. Setzer. *Projeto Lógico e Projeto Físico de Bancos de Dados.* In V Escola de Computação, Belo Horizonte - MG, 1986.