7º seminário integrado de software e hardware

21 a 25 de julho de 1980
FEC/UNICAMP

Promoção
- SBC/Sociedade Brasileira de Computação

Patrocinio
- SEI/Secretaria Especial de Informática
- FEC/Faculdade de Engenharia de Campinas
- UNICAMP/Universidade Estadual de Campinas

TITULO

PRESERVAÇÃO DE RESTRIÇÕES DE INTEGRIDADE E SEGURANÇA EM BANCO DE DADOS

AUTORES: LUCAS MORTIMER MACEDO
A. L. FURTADO

APRESENTADOR: LUCAS MORTIMER MACEDO

ENTIDADE: Pontifícia Universidade Católica do Rio de Janeiro

RESUMO

Este trabalho apresenta, através de um exemplo, uma metodologia para projeto de banco de dados levando em consideração:

- integridade de dados
- segurança de acesso
- facilidades operacionais para os usuarios

No exemplo adotou-se o modelo relacional e foi utilizado o sistema HYADES de gerencia de banco de dados, desenvolvido na PUC/RJ. Entretanto a metodologia independe da escolha do modelo e pode ser usada com qualquer sistema de gerencia de banco de dados.

Agradecimentos

Esta pesquisa teve o apoio financeiro da Plandata Consultoria e Processamento Ltda., do CNPq e da FINEP. Os autores agradecem a A. Rubinstein pela formatação do texto.

1. Introdução

Em todo banco de dados devem ser consideradas regras que determinam se os dados estão ou não corretos. Essas regras são denominadas restrições de integridade. Uma restrição de integridade pode ser violada pela execução (criação, remoção ou modificação de registros).

Por outro lado, existem regras que definem quais usuarios tem direito de ter acesso a quais partes do banco de dados. Essas regras são chamadas de restrições de autorização. Uma restrição de autorização pode ser violada pela execução de operações de consulta ou de atualização por usuarios não autorizados. Da obediencia a essas restrições depende a segurança do banco de dados.

Varios métodos tem sido propostos para impedir a violação dos dois tipos de restrições, alguns tendo sido incorporados aos sistemas de gerencia de banco de dados (SGBDs), como o Sistema-R [AST] e o sistema INGRES [STO]. Ambos são de disponibilidade muito limitada. Os SGBDs de uso corrente no mercado dispõem de facilidades menos poderosas, em grau variado.

Quando não se pode ou não se deseja (por considerações de custo/beneficio) impedir cada execução indevida de operações, uma segunda solução é a de auditorar o banco de dados periodicamente. Isso significa que algum programa para este fim é executado de tempos em tempos, examinando os dados e detetando as violações de restrições ocorridas. Esta orientação implica em permitir que o banco de dados esteja incorreto durante o intervalo de tempo entre duas auditorias, e também que manipulações por usuarios não autorizados não sejam surpreendidas imediatamente.

Uma terceira solução, proposta neste trabalho, consiste em limitar o acesso ao banco de dados a um determinado número de funções (sub-programas, procedimentos, rotinas) pré-programadas, sendo permitido a cada usuario (ou classe de usuarios) o uso de apenas um sub-conjunto especificado dessas funções.

Para que se avalie o que significa esta abordagem é interessante comparar dois tipos de SGBDs

- os sistemas abertos, tais como IMS, TOTAL, ADABAS, DMS-II, IDMS, etc., que contem comandos básicos através dos quais se pode compor qualquer transação (conjunto de operações) de atualização ou consulta;
- os sistemas fechados, tais como os sistemas de reserva de passagens aereas, reserva em hoteis, etc., que contem um conjunto fixo pré-determinado de transações.

Essencialmente a solução que propomos nos coloca em um ponto intermediario quanto a esses dois tipos de sistemas. Estaremos usando um SGBD aberto para nele programar funções correspondendo a um elenco pré-determinado de transações; entretanto sempre que houver necessidade é possivel (embora demande a tomada de certas providencias pelo Administrador de Banco de dados - ABD) aumentar esse elenco ou reduzi-lo.

Devemos advertir ao leitor de que os problemas que surgem com o uso concorrente e com sistemas distribuidos não serão considerados neste trabalho.

A seção 2 discute a metodologia em detalhe, a seção 3 a apresenta através de um exemplo e a seção 4 contem as conclusões. O apêndice contem uma função de consulta e uma de atualização.

2. Descrição da metodologia

Para o projeto de banco de dados vamos seguir a arquitetura em tres niveis de esquemas, conforme a proposta de padronização do grupo ANSI/X3/SPARC [ANS]:

- esquema conceitual - descrição em termos lógicos de banco de dados completo;
- esquemas externos - descrição em termos lógicos de banco de dados a que cada usuario (ou classe de usuarios) tem acesso;
- esquema interno - descrição em termos de implementação física de banco de dados completo.

A figura F-2.1 representa essa arquitetura, sendo que as setas indicam que os esquemas respectivos devem ser exprimiveis um em termos do outro.

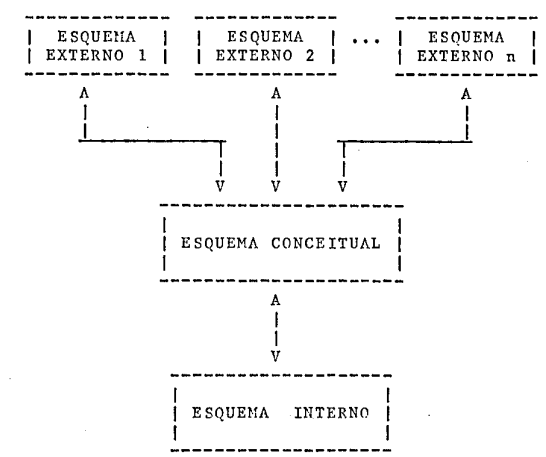

Figura F-2.1 Arquitetura em tres niveis de esquema.

Como usaremos no exemplo o modelo relacional [COD], o esquema conceitual toma a forma de um conjunto de relações base. No esquema interno cada relação base corresponde a um segmento (arquivo) de dados, sendo tambem utilizados segmentos de inversões para acesso mais rápido aos segmentos de dados; segmentos de dados e de inversões são disponiveis no sistema HYADES [PAS], utilizado para implementar o exemplo.

Cada esquema externo consiste de um conjunto de visões. Uma visão é obtida através de projeções, junções, uniões, restrições, partições ou de -partições [FKE] realizadas sobre relações base. No entanto, para um usuario, uma visão tem o mesmo aspecto de uma relação base já que os usuarios não precisam ter conhecimento de como as visões são formuladas. Esse aspecto é o de uma simples tabela.

Agora vejamos como são especificadas as funções de consulta e de autorização.

A cada visão corresponde uma função de consulta, que essencialmente obtem a visão a partir das relações base e a exibe ao usuario (por impressão ou através da tela do terminal). Na realidade nossa implementação é um pouco mais flexivel, permitindo que o usuario especifique um ou mais valores dos atributos da visão de modo que somente as tuplas (linhas da tabela de que consiste a visão) contendo esses valores sejam exibidas. Nas Conclusões indicaremos como uma flexibilidade ainda maior poderia ser obtida.

As funções de atualização se aplicam tambem sobre visões. Ao programa-las devem ser previstos os seguintes elementos, de modo a garantir a preservação das restrições de integridade:

- condições - que, se não forem verificadas, levam a função a simplesmente retornar um código de advertencia avisando ao usuario de que a atualização não pode ser efetuada;
- efeitos - que correspondem às operações de atualização que o usuario pretende executar;
- efeitos colaterais - que são outras atualizações complementares, não diretamente pretendidas pelo usuario, mas que são requeridas para a preservação da integridade.

Enquanto para consulta o usuario tem acesso a todos os atributos das visões de seu esquema externo, para atualização ele poderá estar limitado a alguns atributos apenas.

Ao escrever um programa de aplicação, o usuario incluirá uma ou mais chamadas a uma ou mais funções de consulta e/ou atualização. Na implementação que efetuamos usando o sistema HYADES um programa de aplicação tem o aspecto de um programa qualquer em PL/I. Como consequencia secundaria da metodologia adotada ficam assim reduzidas as necessidades de treinamento dos programadores, os quais não precisam aprender o uso do SGBD utilizado, bastando que saibam como chamar as funções; a implementação tambem envolve um procedimento catalogado na linguagem de controle (JCL) que dispensa a declaração dos arquivos do HYADES e de outras informações requeridas pelo HYADES e pelo sistema operacional.

Afora isso, o programador só deverá ter um cuidado especial, requerido para garantir as restrições de autorização: o primeiro comando executavel deverá ser uma chamda ao módulo HYAFCIN. Esse módulo efetua uma série de inicializações, das quais a mais importante para a presente discussão é trazer para posições da memoria principal compartilhadas por HYAFCIN e pelas funções (mas não pelo programa de aplicacão do usuario) informações sobre o usuario que estão gravadas em um segmento de dados especial constituindo a relação de autorizações.

Cada tupla (linha,registro) da relação de autorizações corresponde a um usuario, indicando para esse usuario:

- nome do usuario;
- sua senha;
- seu cargo;
- funções que está autorizado a utilizar.

Este último elemento tem a forma de um vetor de bits em que cada conjunto de bits (um byte na implementação) corresponde a uma visão e cada bit do conjunto corresponde a uma função sobre a visão (na implementação sobraram bits nos diversos bytes, possibilitando expansões futuras por acréscimo de novas funções). O primeiro bit de cada conjunto designa a (única por visão) função de consulta, e os demais as de atualização. Se o bit tem valor 1, o usuario está autorizado a chamar a função.

Voltemos à execução dos programas de aplicaçao. O módulo HYAFCIN lê o nome e senha do usuario (fornecidos por cartão ou teclado) e os utiliza como chave para localizar a tupla do usuario na relação de autorizações, trazendo-a para a memoria. Se não se tratar de usuario reconhecido pelo sistema a execução é interrompida. Caso contrario prossegue e, ao ser chamada cada função, testes programados no inicio da função verificam por consulta ao vetor de bits se o usuario está autorizado a chama-la; se não estiver, a execução da função se interrompe sendo retornado um código de advertencia.

Uma medida adicional de segurança adotada na implementação foi a codificação ("encryption") do nome e senha do usuario, efetuada pelo sistema, preliminar ao acesso à relação de autorizações.

Certamente as medidas adotadas não tornam totalmente impossivel a violação à segurança, mas a dificultam a um nivel consideravel. O sucesso da metodologia depende ainda:

- de os usuarios serem instruidos sobre o uso das funções;
- de a empresa impor como norma que tais instruções devem ser seguidas rigorosamente;
- de os comandos básicos do SGBD não serem disponiveis aos usuarios de aplicações.

A metodologia dá ênfase ao papel do ABD, ao qual (ou a cuja equipe) compete especificar e programar as funções usando o SGBD e criar e modificar a relação de autorizações; para este último fim a implementação prevê funçoes privilegiadas que só o ABD pode utilizar.

Para programar as funções o ABD precisa conhecer as relações base (esquema conceitual) e como estão implementadas (esquema interno). Em particular, cabe-lhe decidir que arquivos de inversões devem existir para otimizar o tempo de acesso, levando em conta a frequencia das operações. Por outro lado, o programador de aplicações não tem conhecimento disso e, se o banco de dados for reestruturado, seus programas de aplicação não precisam ser reescritos, desde que as sequencias de chamada de suas funções (nome da função e parâmetros) não tenham sido alteradas; isso é verdade mesmo que o corpo das funcões tenha sido alterado (pelo ABD). Em consequencia, um outro beneficio indireto da metodologia é uma maior independencia dos programas quanto à estrutura de dados ("data independence").

O ABD conhecerá também todos os esquemas externos, estando assim em condições de disciplinar a interferencia entre usuarios, fenômeno que ocorre quando dois ou mais esquemas externos compartilham partes do banco de dados.

3. Um exemplo

O exemplo implementado é o caso hipotético da area de pessoal de uma pequena companhia industrial [FSA].

Supõe-se, que feita a análise do sistema de informações, foram identificadas as seguintes entidades e seus atributos:

- empregados
  - N - nome do empregado
  - S - salario
  - C - cargo
  - H - habilitação
- projetos
  - P - nome do projeto
  - T - tarefa
  - L - lider do projeto
  - H - habilitação requerida por tarefa

O esquema conceitual consiste das seguintes relações base:

- EMP(N,S,C) - nome, salario e cargo do empregado
- REQ(T,H) - requisito de habilitação para executar tarefa
- ATR(N,T,P) - atribuição de empregado em projeto a tarefa
- SUP(P,L) - supervisão de projeto por um lider
- CAP(N,H) - capacitação (habilitações) possuidas ou adquiridas por empregados

Os usuarios do banco de dados são:

- gerente de pessoal
- gerente de engenharia
- gerente de treinamento
- líderes de projetos

A análise do sistema de informações tambem deteta que dados deverão ter acesso autorizado a quais usuarios para simples consulta ou para consulta e atualização, e que restrições de integridade serão impostas.

Os quadros Q-3.1 e Q-3.2 mostram:

- os esquemas externos dos usuarios;
- para cada função de atualização, as condições, efeitos e efeitos colaterais respectivos.

USUÁRIO/VISÃO		DESCRIÇÃO
GPE	V-LIVRE (N,S,C)	empregados não associados a projetos. É obtida de EMP, selecionando as tuplas (n,s,c) tais que n não aparecem nenhuma tupla de ATR. operações:ADMITA, DESPEÇA.
	V-OCUP (N,S,C)	empregados associados a pelo menos um projeto. É obtida de EMP selecionando as tuplas tais que n aparece em pelo menos uma tupla de ATR. Sem operações.
GEN	V-NECESS (T,H)	habilitação necessária para desempenhar uma tarefa. É idêntica a REQ. operações: REQUEIRA, REMOVA.
	V-QUAL (N,C,H)	qualificações dos empregados. É obtida concatenando as tuplas de EMP e CAP com o mesmo n. Sem operações.
	V-PROJ (P,L)	projetos e seus líderes. É idêntica a SUP. operações: INICIE, SUBSTITUA, SUSPENDA, REINICIE, ENCERRE.
	V-DISTR (N,T,P)	distribuição de empregados a projetos e tarefas. É idêntica a ATR. operações: ASSOCIE, DESASSOCIE.
GRH	V-REC,HU (N,C,H)	recursos humanos. É obtida de EMP e CAP concatenando tuplas com o mesmo n. operações: ADQUIRA, PERCA.
	V-UTIL (H)	habilitações requeridas por pelo menos uma tarefa. É obtida tomando o domínio H de REQ. sem operações.
LPJ	V-EQUIPE-p (N,C,H)	empregados no projeto p. É obtida de EMP e CAP concatenando tuplas com o mesmo n e tomando somente as tuplas tais que (n,-,p) está em ATR e (p,l) está em SUP, sendo l o usuário (somente o líder do projeto está autorizado a ter tais informações referentes ao seu projeto). Sem operações.
	V-ALOC-p (N,T)	alocação de empregados a tarefas no projeto p. É obtida de ATR tomando apenas as tuplas (n,-,p) tais que (p,l) está em SUP, sendo l o usuário, e tomando apenas os domínios N e T. Operações: ENCARREGUE, LIBERE.
	V-TAREFA-p (T,H)	exigência de habilitação para tarefa. É idêntica a REQ. Sem operações.

Quadro Q-3.1 Visões dos usuários autorizados.

Para exemplificar, consideremos a visão V-LIVRE do esquema externo do gerente de pessoal. Para consulta existe a função LIVRE com tres parâmetros: N, S, C. Se for chamada por

COD = LIVRE('joão da silva',-,-)

será exibida a tupla (única) correspondente ao empregado com o nome indicado. COD recebe o código indicativo de término normal ou de advertencia. A chamada com

COD = LIVRE(-,-,'motorista')

causa a exibição de todas as tuplas de empregados com o cargo indicado. A chamada com todos os parâmetros em branco ocasiona a exibição de toda a visão.

Sobre essa visão podem ser executadas as operações ADMITA(N,S,C) e DESPECA(N), evidentemente com o efeito pretendido de acrescentar ou remover uma tupla de V-LIVRE. As restrições de integridade que nos interesam aqui são:

a. Os salarios devem ser pelo menos iguais ao salario mínimo.

b. Um empregado contratado deve ter um e apenas um salario e cargo.

c. Somente empregados contratados podem ter suas habilitações registradas no banco de dados.

d. Somente empregados que não estão no momento associados a projeto algum podem ser despedidos.

A visão V-LIVRE contem apenas empregados não associados a projetos (V-OCUP contem os que se ocupam de pelo menos um projeto). A função ADMITA verifica, como condição, se já há alguma tupla correspondente ao empregado que se quer contratar (restrição b) e se o salario a ser concedido é inferior ao mínimo (restrição a), casos em que ADMITA falha, retornando diferentes códigos de advertencia. Satisfeitas as condições, a inserção da tupla é feita na relação base EMP (da qual V-LIVRE é derivada como visão).

DESPEÇA só se aplica se o empregado não está associado a projetos (restrição d) e tem como efeito colateral remover as tuplas da relação base CAP que contenham habilitação do empregado demitido (restrição c). O efeito pretendido é obtido pela remoção das tuplas do empregado da relação base EMP.

Os programas das funções LIVRE e DESPEÇA constam do Apêndice. Linhas de comentarios permitem examinar como funcionam os programas sem requerer conhecimento do sistema HYADES; merece referencia o uso do procedimento SELINV do HYADES, para consulta a segmentos de inversões.

A implementação tambem inclui um programa genérico para usuarios de aplicações, conveniente para uso através de terminal por usuarios não programadores. O programa requer que o usuario indique a função a executar e, depois de chama-la e exibir o resultado, volta a ficar disponivel para nova chamada de função, até que o usuario encerre a sessão.

Quadro Q-3.2 Operações de atualização.

4. Conclusões.

A metodologia proposta pode ser usada com qualquer SGBD, não dependendo de que este disponha de recursos sofisticados.

As restrições de integridade e de autorização são preservadas às custas de uma certa perda de flexibilidade. Em especial, se novas funções de atualização se tornam necessarias mais tarde (ex: para modificar salarios de empregados), é necessario que o usuario interessado solicite sua especificação e programação ao ABD.

Na prática será necessaria uma análise de custo/beneficio para determinar para que restrições é compensatorio utilizar a presente metodologia (ou outra qualquer) para garantir sua preservação. Em certos casos pode ser conveniente a politica de auditoria periódica, ou até mesmo a de permitir a ocorrencia e permanencia de erros de menor importancia.

Quanto à presente implementação, varios aperfeiçoamentos podem ainda ser introduzidos. Um deles é a saida das funções de consulta, que poderia consistir na gravação de um arquivo (que assim seria um instantaneo - "snapshot" - da visão) ao invés de imprimir a visão ou exibi-la em terminal; tambem se poderia adicionar como parametro uma função lógica (predicado), a ser programada separadamente pelo usuario, impondo condições à mais para exibir ou não cada tupla da visão (ex: só exibir tuplas de empregados com salario maior do que um certo valor e cujo cargo seja motorista ou mecânico de veículos, etc.).

Referencias

[ANS] ANSI/X3/SPARC - "Interim Report ANSI/X3/SPARC Study Group on Data Base Management Systems" - boletim FDT da ACM (1975).

[AST] Astraham, M. M. e outros - "System R: relational approach to database management" - ACM transactions on Database Systems 1,2 (1976).

[COD] Codd, E. F. - "A relational model of data for large shared data banks" - Communications of the ACM (1970).

[FKE] Furtado, A. L. e Kerschberg, L. - "An algebra of quotient relations" - Anais da SIGMOD Conference (1977).

[FSA] Furtado, A. L. e Santos, C. S. - "Organização de Banco de Dados" - Ed. Campus (1979).

[PAS] Passos, S. M. M., Vasques, R. P. e Furtado A. L. - "Um suporte básico para implementação de sistemas de gerencia de banco de dados" - Anais do Panel Expodata V (1978).

[STO] Stonebraker, M. R. e outros - "The design and implementation of INGRES" - ACM transactions on Database Systems - 1,3 (1976).

Apêndice

```
LIVRE: PROC(NOME,SAL_CAR,CARGO) RETURNS(BIN FIXED);        /*23.02.80*/
/*     ROTINA DE CONSULTA DO EXEMPLO DE APLICACAO DO SISTEMA HYADES
       ===========================================================,
       VISAO  LIVRE  DO  DEPTº DE  PESSOAL.
       FORNECE OS EMPREGADOS DA FIRMA QUE NAO ESTAO ASSOCIADOS A PROJETOS.  E' OBTIDA DE 'EMP' SELECIONANDO AS TUPLAS      (N,S,C)
       TAIS QUE  N  NAO APARECE ; M NENHUMA TUPLA  (N,T,P)  DE  'ATR'.
     UTILIZACAO:
     ==========,
       EM PROGRAMAS ESCRITOS EM PLI=F, CHAMAR COM O COMANDO:
          CODIGO = LIVRE (NOME, SAL_CAR, CARGO)
       SENDO 'CODIGO' O VALOR DE RETORNO DA FUNCAO, QUE INDICARA'  O
       NUMERO DA MENSAGEM DE SUCESSO OU FRACASSO NO USO DA MESMA.
     PARAMETROS DA PROCEDURE:
     ========================,
       NOME    : NOME DO EMPREGADO                          ALFANUM.  (10)
       SAL_CAR: VALOR DO SALARIO DO EMPREGADO               ALFANUM.  (02)
       CARGO  : INICIAIS DO CARGO DO EMPREGADO              ALFANUM.  (03)
     VALORES DOS PARAMETROS:
     =======================,
       OS PARAMETROS DESTA PROCEDURE PODERAO CONTER VALORES ESPECIFICOS OU ESTAR EM BRANCO. QUANDO O VALOR DE UM PARAMETRO  ESTA'
       ESPECIFICADO NA CHAMADA A ROTINA DE CONSULTA  LIVRE,  SOMENTE
       SERAO EXIBIDAS AS TUPLAS COMOS VALORES SATISFACAM  OS  VALORES
       PEDIDOS.  SE O VALOR DE UM PARAMETRO ESTA' EM BRANCO, TODAS AS
       TUPLAS PARA AQUELE PARAMETRO SERAO EXIBIDAS.
     CODIGOS DE RETORNO:
     ===================,
       CODIGO = 40 - FUNCAO BEM SUCEDIDA.
       CODIGO = 62 - TENTATIVA DE UTILIZACAO DA  FUNCAO  POR  USUARIO
                     NAO AUTORIZADO.
       CODIGO = 63 - NAO EXISTEM TUPLAS PARA SEREM EXIBIDAS.
       CODIGO = 0 A 21 - ERRO NO SUPORTE HYADES DE BANCO DE DADOS.
     ESTRUTURA DA ROTINA:
     ====================,
       1. TESTAR AUTORIZACAO.
       2. INICIALIZACAO DE VARIAVEIS DE TRABALHO.
       3. SELECAO DA PRIMEIRA TUPLA.
       4. VERIFICACAO E TESTE DA TUPLA/ SELECAO DAS DEMAIS TUPLAS
       5. LIBERACAO DOS SEGMENTOS (RELACOES) UTILIZADAS.
*/
    /*DECLARACOES DOS PARAMETROS DA PROCEDURE                          */
    DCL  NOME                 CHAR (10);
    DCL  SAL_CAR              CHAR(02),
      1 SAL_D     DEFINED     SAL_CAR,
      2 SALARIO               PIC'99';
    DCL  CARGO                CHAR (3);
    /*DECLARACOES DAS VARIAVEIS EXTERNAS                               */
    DCL  VET_AUT(96)          BIT(1) EXTERNAL; /* VET. AUTORIZACOES  */
    DCL  HEXLIVR              BIN FIXED(15,0)  /* INDICE DO VETOR    */
                              EXTERNAL;
    DCL  SYSPRINT             FILE;
    /*DECLARACOES DAS PROCEDURES EXTERNAS ATIVADAS                     */
    DCL  SELDAD               ENTRY (CHAR(7),PIC'9',PIC'(6)9',
                                     CHAR(502))
                              RETURNS (BIN FIXED);
    DCL  SELINV               ENTRY (CHAR(7),PIC'9',CHAR(249),
                                     CHAR(7),PIC'(6)9')
                              RETURNS (BIN FIXED);
    /*DECLARACOES DOS PARAMETROS DAS PROCEDURES EXTERNAS               */
    /* DESCRICAO DE RELACAO   -  SISTEMA HYADES DE BANCO DE DADOS      */
    /*       SEGMENTO :  EMP   - EMPREGADOS DA EMPRESA                 */
```

```
    /*      TUPLAS   : (N,S,C)-  NOME, SALARIO, CARGO                  */
    /*      MEMBRO   : REGEMP                                          */
    DCL  REG_EMP   EXTERNAL CHAR (502);    /* REG_EMP - EXTERNAL      */
    DCL  P_EMP              POINTER;       /* POINTER                 */
    DCL  1 REGISTRO_EMP     BASED (P_EMP), /* ESTRUTURA - 15 BYTES    */
           2 N              CHAR (10),     /* NOME DO EMPREGADO       */
           2 S              DEC FIXED (02),/* SALARIO                 */
           2 C              CHAR (03);     /* CARGO                   */
    /*------------------------------------------------------------------*/
    /* DESCRICAO DE RELACAO   -  SISTEMA HYADES DE BANCO DE DADOS      */
    /*       SEGMENTO :  ATR   -  ATRIBUICOES DE TAREFAS NOS PROJETOS */
    /*       TUPLAS   : (N,T,P)-  NOME, TAREFA, PROJETO               */
    /*       MEMBRO   : REGATR                                        */
    DCL  REG_ATR   EXTERNAL CHAR (502);    /* REG_ATP - EXTERNAL      */
    DCL  P_ATR              POINTER;       /* POINTER                 */
    DCL  1 REGISTRO_ATR     BASED (P_ATR), /* ESTRUTURA - 35 BYTES    */
           2 N              CHAR (10),     /* NOME DO EMPREGADO       */
           2 T              CHAR (15),     /* NOME DA TAREFA          */
           2 P              CHAR (10);     /* NOME DO PROJETO         */
    /*------------------------------------------------------------------*/
    DCL   ID_REGEMP         PIC'(6)9',     /* IDENTIFICADOR REG.EMP   */
          ID_REGATR         PIC'(6)9',     /* IDENTIFICADOR REG.ATR   */
          ID_ARQDUM         CHAR(7),       /* IDENTIFICACAO ARQUIVO   */
          TIP_SEL           PIC'9',        /* TIPO DE SELECAO INV.    */
          DUM239            CHAR(239)  INIT(' ');
    /*DECLARACOES DE VARIAVEIS AUXILIARES                              */
    DCL   COD               BIN FIXED(15), /*COD.RETORNO PRIMITIVOS   */
          CODIGO            BIN FIXED(15), /*COD.RETORNO FUNCOES      */
          X                 BIT(1),        /* VAR.ASSOCIADA A NOME    */
          Y                 BIT(1),        /* VAR.ASSOC. A SALARIO    */
          Z                 BIT(1);        /* VAR.ASSOC. A CARGO      */
    /*ESTPUTURA PRINCIPAL                                              */
    /* 1. TESTE AUTORIZACAO                                            */
    IF VET_AUT (HEXLIVR) = '0'B
    THEN CODIGO = 62;                     /* USUARIO NAO AUTORIZ.    */
    ELSE DO;
    /* 2. INICIALIZACAO DE VARIAVEIS DE TRABALHO                       */
         P_EMP = ADDR(REG_EMP);
         P_ATR = ADDR(REG_ATR);
         IF NOME ¬= ' '
         THEN X = '1'B;
         ELSE X = '0'B;
         IF SAL_CAR ¬= ' '
         THEN Y = '1'B;
         ELSE Y = '0'B;
         IF CARGO ¬= ' '
         THEN Z = '1'B;
         ELSE Z = '0'B;
         IF X = '1'B
         THEN TIP_SEL = 4;
         ELSE TIP_SEL = 1;
         CODIGO = 63;                     /* NAO EXISTE TUPLA        */
    /* 3. SELECAO DA PRIMEIRA TUPLA EM 'EMPT' ATRAVES DE 'HOMEEMP'     */
         COD = SELINV('HOMEEMP',TIP_SEL,NOME||DUM239,ID_ARQDUM,
                      ID_REGEMP);
    /* 4. VERIFICAO E TESTE DA TUPLA LIDA/SELECAO DEMAIS TUPLAS        */
         DO WHILE (COD=0);                /* EXISTE TUPLA (N,S,C)    */
              COD = SELDAD('EMPT   ',0,ID_REGEMP,REG_EMP);
              IF\ OD ¬= 0
              THEN CODIGO = COD;          /* ERRO HYADES             */
              ELSE DO;
                   IF (X = '0'B  |  NOME   = REGISTRO_EMP.N) &
                      (Y = '0'B  |  SALARIO = REGISTRO_EMP.S) &
                      (Z = '0'B  |  CARGO   = REGISTRO_EMP.C)
                   THEN DO;       /* TESTAR SE N EXISTE EM ATR       */
```

```
                    COD = SELINV('NOMEATR',4,
                                 REGISTRO_EMP.N||DUM239,
                                 ID_ARQDUM,ID_REGATR);
                    IF COD = 8          /* (N,T,P) NAO ESTA ATR */
                    THEN DO;            /* IMPRIMIR A TUPLA     */
                         PUT SKIP(2) EDIT ('* LIVRE * NOME= ',
                                           REGISTRO_EMP.N,
                                           ' /      SALARIO = ',
                                           REGISTRO_EMP.S,
                                           ' /      CARGO = ',
                                           REGISTRO_EMP.C)
                              (COL(1),A,A,A,P'99',A,A);
                         CODIGO = 40;
                         COD    = 0;
                         END;
                    ELSE IF COD ^= 8
                         THEN CODIGO=COD;/* ERRO HYADES          */
                    IF   X              /*SE NOME FOI ESPECIFIC.*/
                    THEN COD = 99;      /* PARA SAIR DO LOOPING*/
                    END;
               IF COD = 0
               THEN COD = SELINV('NOMEEMP',2,NOME||DUM239,
                                 ID_ARQDUM,ID_REGEMP);

                END;
          END;
     /* 5. LIBERACAO DOS SEGMENTOS (RELACOES) UTILIZADAS.          */
        COD = SELINV('NOMEEMP',9,NOME||DUM239,ID_ARQDUM,ID_REGEMP);
        COD = SELINV('NOMEATR',9,NOME||DUM239,ID_ARQDUM,ID_REGATR);
        COD = SELDAD('EMPT    ',9,ID_REGEMP,REG_EMP);
        END;
     RETURN(CODIGO);
END LIVRE;
```

```
DCL  VET_AUT(96)         BIT(1) EXTERNAL, /* VETOR AUTORIZACOES */
     HEXDESP       BIN  FIXED  EXTERNAL; /* INDICE DE VET_AUT  */
/*DECLARACOES DE VARIAVEIS AUXILIARES                           */
DCL (COD,
     CODIGO)             BIN  FIXED;
/*ESTRUTURA PRINCIPAL                                           */
IF VET_AUT(HEXDESP) = '0'B
THEN CODIGO = 62;                       /* USUARIO NAO AUTORIZ.*/
ELSE DO;
P_EMP = ADDR (REG_EMP);
P_CAP = ADDR (REG_CAP);
P_ATR = ADDR (REG_ATR);
COD = SELINV ('NOMEEMP',4,NOME || DUM239,ID_ARQDUM,ID_REGEMP);
IF   COD ^= 0
THEN CODIGO = 46;                       /* EMPREGADO N.EXISTE  */
ELSE DO;
     COD = SELINV ('NOMEATR',4,NOME||DUM239,ID_ARQDUM,ID_REGATR);
     IF   COD = 0
     THEN CODIGO = 51;                  /* N VINCULADO A PROJETO*/
     ELSE DO;
          IF   COD ^= 8
          THEN CODIGO = COD;            /* ERRO HYADES          */
          ELSE DO;
               COD = REMDAD ('EMPT    ',ID_REGEMP);
               IF   COD ^= 0
               THEN CODIGO = COD;       /* ERRO HYADES          */
               ELSE DO;
                    CODIGO = 40;        /* TERMINO NORMAL       */
                    COD = SELINV ('NOMECAP',4,NOME||DUM239,
                                  ID_ARQDUM,ID_REGCAP);
                    IF   COD = 0
                    THEN DO WHILE (COD = 0  &  ID_REGCAP ^= 0);
                              COD = REMDAD ('CAPT    ',ID_REGCAP);
                              IF   COD ^= 0
                              THEN CODIGO = COD;
                              ELSE COD = SELINV ('NOMECAP',5,NOME
                                   ||DUM239,ID_ARQDUM,ID_REGCAP);
                         END;
                    COD = SELINV ('NOMECAP',9,NOME||DUM239,
                                  ID_ARQDUM,ID_REGCAP);
                     END;
               END;
          END;
     COD = SELINV ('NOMEATR',9,NOME||DUM239,ID_ARQDUM,ID_REGATR);
     END;
COD = SELINV ('NOMEEMP',9,NOME || DUM239,ID_ARQDUM,ID_REGATR);
END;
RETURN(CODIGO);
END DESPECA;
```

```
DESPECA: PROC (NOME)  RETURNS (BIN FIXED);          /* 15.03.80 */
/***** EFEITO:
         RETIRA DOS ARQUIVOS OS DADOS REFERENTES A UM EMPREGADO,
         PARA QUE O EMPREGADO POSSA SER DESPEDIDO, E NECESSARIO
         QUE NAO ESTEJA VINCULADO A NENHUM PROJETO(S).
       PARAMETRO DA PROCEDURE:
         NOME: NOME DO EMPREGADO QUE DEVERA SER DESPEDIDO.
               ALFANUMERICO COM 10 POSICOES.
       CODIGOS DE RETORNO:
         CODIGO = 40 - FUNCAO BEM SUCEDIDA.
         CODIGO = 46 - TENTATIVA DE DEMITIR UM EMPREGADO INEXISTENTE.
         CODIGO = 51 - TENTATIVA DE DEMITIR UM EMPREGADO VINCULADO
                       A UM PROJETO.
         CODIGO = 62 - USUARIO NAO AUTORIZADO A UTILIZAR ESTA FUNCAO.
*****/
     /*DECLARACAO  DO  PARAMETRO  DA PROCEDURE                    */
     DCL  NOME                  CHAR (10);
     /*DECLARACOES DAS PROCEDURES EXTERNAS ATIVADAS               */
     DCL  SELDAD                ENTRY (CHAR(7),PIC'9',PIC'(6)9',
                                       CHAR(502))
                                RETURNS (BIN FIXED);
     DCL  REMDAD                ENTRY (CHAR(7),PIC'(6)9')
                                RETURNS (BIN FIXED);
     DCL  SELINV                ENTRY(CHAR(7),PIC'9',CHAR(249),
                                      CHAR(7),PIC'(6)9')
                                RETURNS (BIN FIXED);
     /*DECLARACOES DOS PARAMETROS DAS PROCEDURES EXTERNAS         */
     /* DESCRICAO DE RELACAO   -  SISTEMA HYADES DE BANCO DE DADOS  */
     /*      SEGMENTO :  EMP   -  EMPREGADOS DA EMPRESA            */
     /*      TUPLAS   : (N,S,C)-  NOME, SALARIO, CARGO             */
     /*      MEMBRO   : REGEMP                                     */
     DCL  REG_EMP   EXTERNAL  CHAR (502);    /* REG_EMP - EXTERNAL  */
     DCL  P_EMP               POINTER;       /* POINTER             */
     DCL  1 REGISTRO_EMP      BASED (P_EMP), /* ESTRUTURA - 15 BYTES */
            2 N               CHAR (10),     /* NOME DO EMPREGADO   */
            2 S               DEC FIXED (02),/* SALARIO             */
            2 C               CHAR (03);     /* CARGO               */
     /*=========================================================*/
     /* DESCRICAO DE RELACAO   -  SISTEMA HYADES DE BANCO DE DADOS  */
     /*      SEGMENTO :  ATR   -  ATRIBUICOES DE TAREFAS NOS PROJETOS*/
     /*      TUPLAS   : (N,T,P)-  NOME, TAREFA, PROJETO            */
     /*      MEMBRO   : REGATR                                     */
     DCL  REG_ATR   EXTERNAL  CHAR (502);    /* REG_ATR - EXTERNAL  */
     DCL  P_ATR               POINTER;       /* POINTER             */
     DCL  1 REGISTRO_ATR      BASED (P_ATR), /* ESTRUTURA - 35 BYTES */
            2 N               CHAR (10),     /* NOME DO EMPREGADO   */
            2 T               CHAR (15),     /* NOME DA TAREFA      */
            2 P               CHAR (10);     /* NOME DO PROJETO     */
     /*---------------------------------------------------------*/
     /* DESCRICAO DE RELACAO   -  SISTEMA HYADES DE BANCO DE DADOS  */
     /*      SEGMENTO :  CAP   -  HABILITACOES DOS EMPREGADOS      */
     /*      TUPLAS   : (N,H)  -  NOME, HABILITACAO                */
     /*      MEMBRO   : REGCAP                                     */
     DCL  REG_CAP   EXTERNAL  CHAR (502);    /* REG_CAP - EXTERNAL  */
     DCL  P_CAP               POINTER;       /* POINTER             */
     DCL  1 REGISTRO_CAP      BASED (P_CAP), /* ESTRUTURA           */
            2 N               CHAR (10),     /* NOME DO EMPREGADO   */
            2 H               CHAR (15);     /* HABILITACAO         */
     /*---------------------------------------------------------*/
     DCL (ID_REGEMP,
          ID_REGATR,
          ID_REGCAP)           PIC  '(6)9',
         ID_ARQDUM             CHAR (7),
```