

GEO - GERENCIADOR DE OBJETOS

**MARISA BECK FIGUEIREDO
CAETANO TRAINA JÚNIOR**

Nº 47

RELATÓRIOS TÉCNICOS DO ICMSC

São Carlos
Set./1996

SYSNO	909716
DATA	/ /
ICMC - SBAB	

GEO - Gerenciador de Objetos

Marisa Beck Figueiredo
Caetano Traina Júnior

Índice

CAPÍTULO 1.....	3
1.1 INTRODUÇÃO	3
 CAPÍTULO 2.....	 6
2.1 OPERADOR DE GERENCIAMENTO	6
 CAPÍTULO 3.....	 10
3.1 INCLUSÃO DE ESQUEMA E BASE DE DADOS NO GEO.....	10
3.2 EXEMPLOS	18
 CAPÍTULO 4.....	 40
4.1- UTILIZAÇÃO DAS ROTINAS	40
4.1.1- <i>Obtenção do nome dos tipos pertencentes a uma colônia</i> :.....	40
4.1.2 - <i>Obtenção do código dos tipos pertencentes a uma colônia</i> :.....	41
4.1.3 - <i>Parâmetros Globais</i> :.....	45
4.2 - OPERAÇÃO COM ROTINAS	52
4.2.1 - <i>Criação</i>	52
4.2.1.1- Como criar um objeto.....	52
4.2.1.2 - Como criar relacionamento binário ou triplo entre dois objetos :.....	53
4.2.1.3- Como criar colônias :	54
4.2.1.4 - Como colocar um atributo em um objeto :.....	55
4.2.1.5 - Como colocar um atributo em um relacionamento :	57
4.2.2 - <i>Exclusão</i>	58
4.2.2.1 - Como retirar um objeto :	58
4.2.2.2 - Como retirar um atributo de um objeto :	58
4.2.2.3 - Como retirar um relacionamento binário :	59
4.2.2.4 - Como retirar um relacionamento triplo :	59
4.2.3 - <i>Leitura</i>	60
4.2.3.1 - Como ler um atributo :	60
4.2.3.2 - Como ler um relacionamento :	61
4.2.4 - <i>Avanço</i>	61
4.2.4.1 - Como avançar para o objeto seguinte :	61
4.2.4.2 - Como avançar um atributo :	61
4.2.4.3 - Como avançar um relacionamento binário :	61

4.2.4.4 - Como avançar um relacionamento triplo :	62
4.2.5 - <i>Posicionamento</i>	62
4.2.5.1 - Como posicionar em um objeto :	62
4.2.5.2- Como posicionar em um relacionamento :	63
4.2.5.3- Como posicionar em uma colônia :	64
4.2.6 - <i>Busca</i>	65
4.2.6.1 - Como definir sequência de busca pelo tipo :	65
4.2.6.2 - Como definir sequência de busca na colônia :	68
4.2.6.3 - Como definir sequência de busca pelos sinônimos :	69
4.2.7 - <i>Abertura</i>	70
4.2.7.1 - Como abrir uma colônia :	70
4.2.8 - <i>Listagem</i>	71
4.2.8.1 - Como listar todos os objetos de uma colônia :	71
4.2.8.2 - Como listar todos os objetos de um determinado tipo, de uma colônia :	72
4.2.8.3 - Como listar todos os objetos, que possuam determinado atributo com um valor determinado, de uma colônia :	73
4.2.8.4 - Como listar todos os valores de todos atributos de um objeto :	74
4.2.8.5 - Como listar todos os valores de um tipo de atributo de um objeto :	74
4.2.8.6 - Como listar todos os valores de todos atributos de todos objetos :	75
4.2.8.7 - Como listar todos os valores de um tipo de atributo de todos objetos :	76
4.2.8.8 - Como listar todos os relacionamentos binários de um objeto :	77

Índice de Figuras

Figura 2.1- Colônia Esquema	08
Figura 3.1- Definição de um Esquema “Campeonatos”	11
Figura 3.2- Definição de um Esquema “Universidade”	21
Figura 4.1- Seleção de Voleibol	42
Figura 4.2- Inclusão de Relacionamentos	49
Figura 4.3- Parâmetros Globais	51
Figura 4.4- Equipes	60
Figura 4.5- Definição das Equipes na Colônia Global	66
Figura 4.6- Definição dos Atletas da Equipe Suzano	66
Figura 4.7- Definição dos Atletas da Equipe Pirelli	67
Figura 4.8- Definição dos Atletas da Equipe Banespa	67
Figura 4.9- Definição dos Atletas da Equipe Palmeiras.....	68
Figura 4.10- Nomes de Equipes	68
Figura 4.11- Nomes Principais de Equipes	69
Figura 4.12- Nomes Principais e Sinônimos de Equipes.....	69
Figura 4.13- Listagem de Equipe e Atleta	75
Figura 4.14- Listagem de Relacionamento Binário	77

Capítulo 1

Introdução

Capítulo 1

1.1 Introdução

Este manual é baseado em um Sistema Gerenciador de Banco de Dados Orientado a Objeto - SGBDOO, chamado **Gerenciador de Objetos - GEO**. O GEO tem sido construído no Instituto de Ciências Matemáticas de São Carlos com o objetivo de validação prática e estudo de como os conceitos de orientação a objetos podem ser implementados em um Sistema de software real. É baseado no modelo de dados denominado **Modelo de Representação de Objetos - MRO**. O principal objetivo de seu desenvolvimento é permitir a construção de Sistemas de Gerenciamento de Base de Dados para suportar a representação de informações dos sistemas de apoio a projetos de engenharia (CAD), de ambientes de suporte ao desenvolvimento de sistemas (CASE), e sistemas de coleta e tratamento de dados científicos/estatísticos.

Para implementar o Gerenciador de forma que este possa manipular eficientemente a Base Extensional e a Base Intencional, o GEO foi dividido em 5 módulos básicos :

1. **Núcleo de Acesso a Registros** : consiste de um sistema de Gerenciamento de Arquivos Lógicos, fornecendo suporte aos demais módulos. É composto pelo Subsistema de Gerenciamento de Registros (SGR). Subsistema de Gerenciamento de Arquivos (SGA) e Subsistema de Gerenciamento de Memória (SGM). Estes módulos realizam todo o gerenciamento de memória secundária, simulação de uma memória cache para disco em memória e controle de acesso e proteção dos dados, através do gerenciador de transações.

2. **Subsistema Básico de Gerenciamento** : suporta a manipulação de Objetos sem restrições de ordem semântica. Envolve o tratamento de Identificadores de Objetos, manipulação de Composições de Objetos, contexto e navegação na Base.

3. **Subsistema Básico de Gerenciamento de Relacionamentos**: suporta a manipulação de relacionamentos (binários ou triplos) definidos pelo usuário. O GEO gerencia relacionamento intrínsecos e relacionamento extrínsecos : os intrínsecos são aqueles cuja semântica é conhecida pelo

Gerenciador, tal como as Hierarquias de Generalização e de Composição, sendo gerenciadas pelo Subsistema Básico de Gerenciamento de Objetos; os extrínsecos são aqueles cuja semântica é definida pelo usuário e são gerenciados por este módulo em questão.

4. Subsistema Básico de Gerenciamento de Atributos : permite a manipulação de Atributos com características de propriedades e comentários, tanto para objetos quanto para relacionamentos.

5. Gerenciador de Esquemas de Dados - GED : consiste nas primitivas de manipulação, acesso e verificação de Esquemas de Dados, as quais são ativadas pelos demais módulos.

Este manual apresenta informações de execução do Operador de Gerenciamento - OPRGER e uma descrição das primitivas do Gerenciador de Objetos - GEO.

As referências adicionais sobre o Modelo de Representação de Objetos e o Gerenciador de Objetos podem ser encontradas em :

[Traina_91] Traina Jr., C. "GEO : Um sistema de base de dados orientado a objetos - Estado atual de desenvolvimento e implementação. In: Simpósio Brasileiro de Banco de Dados, Manaus, maio 1991, p78-93. Anais

[Traina_92] Traina Jr., C. "MRO - Um Modelo de Representação de Objetos", Notas Internas (104) ICMSC-USP, fevereiro 1992.

[Traina_93] Traina Jr., C.; Camolesi Jr., L. "Gerenciador de Esquemas de Dados Baseado no Modelo MRO", Notas Internas (131), ICMSC-USP, março 1993.

Capítulo 2

Operador de Gerenciamento

Capítulo 2

2.1 Operador de Gerenciamento

Definição de um esquema :

O programa OPRGER.EXE é executado da seguinte forma :

C : > OPRGER

Após executá-lo, aparecerão as opções referente a este :

- 0 => Fim das Modificações
- 1 => Inserir um Registro
- 2 => Modifica um registro.
- 3 => Lista o Arquivo.
- 4=> Criar uma Base MRO.

A definição de uma base é feita no OPRGER da seguinte maneira :

1- Criar uma base de dados, onde são informados :

- nome da base;
- código do usuário;
- senha.

Primeiramente devemos escolher a opção de " Inserir Registro ", para posteriormente " Criar a Base ", pois é preciso que a base seja reconhecida pelo sistema.

Na opção "Inserir Registro ", deve ser informado o :

- nome da base;
- estado;
- código;
- fila entrada;

- fila saída.

as informações acima incluídas serão guardadas em um registro existente no arquivo GERGEO (que é criado quando é inserido um registro), que além do nome possui informações gerenciais sobre essa base.

Na criação da base, há a geração dos :

Arquivo com extensão .MRO => Cria somente a estrutura vazia;

Nesta estrutura estão :

- Besquema (colônia global) :

Usuário

- nome
- senha

Obs.: Informações baseadas no que foi informado anteriormente.

Máquina

- vídeo
- máquina

Obs.: Informações baseadas no que foi informado anteriormente

- Colônia Esquema :

- atributo
Tempo Físico ;
- atributo
Tempo Válido ;

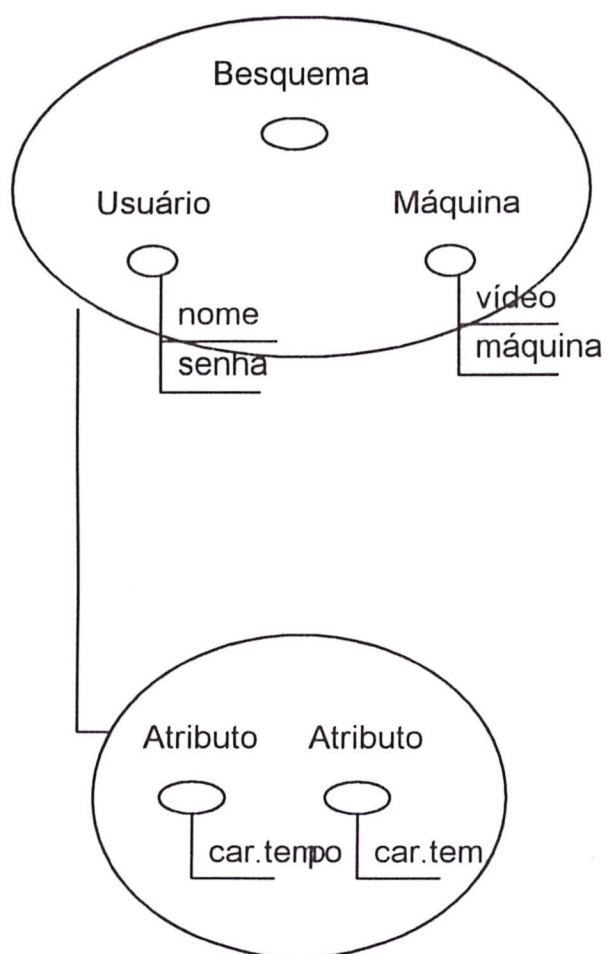


Figura 2.1 - Colônia Esquema

Capítulo 3

3.1 Inclusão de Esquema e Base de Dados no GEO

GEO_INC . EXE :

O Gerenciador de Objetos permite que sejam definidos Esquemas, onde são informados os vários objetos com seus relacionamentos e atributos, montados e baseados no modelo criado pelo Modelo de Representação de Objetos.

Para acessar o GEO_INC basta digitar :

C : > GEO_INC

Na execução deste é possível estabelecer parâmetros, tais como :

- l => este parâmetro permite que liste em STDOUT cada linha lida.
- p => faz com que o prompt não apareça a cada linha lida.
- v (verbose) => realiza impressões de mensagens adicionais.

Os parâmetros devem estar logo após a chamada do executável :

geo_inc [-lpv]

independente da ordem.

Caso não seja informado nenhum parâmetro, o geo_inc será usado como um editor de linha normal.

Após a digitação ou não dos parâmetros, aparecerão :

Nome da Base

Usuário

Senha

onde serão informados os mesmos incluídos no OPRGER. Caso contrário, não será permitida a inclusão do Esquema, já que a base não foi criada.

Após cada comando é necessário a colocação do ponto-e-vírgula (;), exceto do comando " ARQUIVO ". Os comandos podem ocupar qualquer número de linhas, sendo fundamental que no final do comando seja colocado um separador. Os separadores existentes são :

, . ; : + - * / \ () ? ! @ # \$ % & " ' branco \n

A inclusão de comentários é possível através do sinal #, o qual indica que o restante da linha deve ser tratado como comentário. Caso utilize # # então significa que além do comentário, o restante da linha deve ser impressa no arquivo STDOUT.

Exemplo :

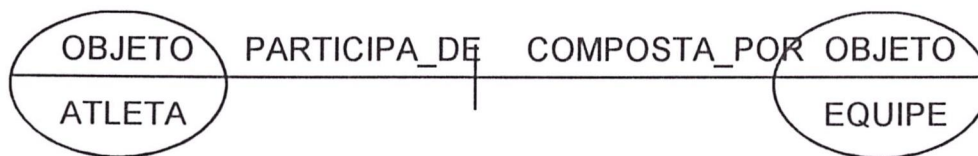


Figura 3.1 - Definição de um esquema " Campeonato "

Para incluir um esquema é necessário primeiramente, posicionar na colônia Esquema, que é feito através do comando LIGA META. A inclusão de dados na base é possível através do comando LIGA BASE. Porém, para isso é necessário primeiramente utilizar o comando para desligar o META :

desliga meta

Na definição de um esquema é necessário a criação de colônias, objetos e a seguir os relacionamentos entre cada objeto e relacionamento.

1- Para incluir os objetos deve-se informar o :

- tipo
- nome

onde os tipos a serem definidos podem ser :

- objeto ⇒ Define um objeto;
- atributo ⇒ Define um atributo;
- binário ⇒ Define um relacionamento;
- modalidade ⇒ Define uma modalidade :
 - modalidade1 = para relacionamento binário
 - modalidade2 = para relacionamento triplo

A sequência de comandos para incluir um objeto deve ser :

objeto objeto atleta equipe

2- Após a inclusão de objetos, podemos iniciar a de atributos :

objeto propriedade altura

Nesta inclusão é necessário que se defina o tipo de propriedade, ou seja, de acordo com alguns parâmetros definidos distinguiremos uma propriedade de um sinônimo.

De acordo com o exemplo mostrado abaixo, consideramos a existência de um atributo do tipo sinônimo, no caso o cognome da Equipe. Este será definido como propriedade porém com característica de sinônimo, ou seja :

propriedade característica 1

Como a característica, há o parâmetro referente ao tipo-de-dado, que permite a formatação dos dados a serem incluídos.

Por exemplo, considerando que o atributo a ser incluído deva aceitar apenas valores inteiros,, usaremos apenas o parâmetro referente a este, que no caso é 2:

propriedade tipo-de-dado 2

Para cada propriedade possuímos as seguintes características e os seguintes tipos-de-dados :

Características :

1 = Sinônimo	7 = Regra
2 = Propriedade	8 = Vis
3 = Comentário	9 = Prc
4 = Tempo	10 = Imagem
5 = Str	11 = Som
6 = Gráfico	

Tipo-de-Dado :

1 = Byte	7 = Unsigned Char
2 = Pal2	8 = Struct B_2dgraf
3 = Long	9 = Struct B_Data
4 = Codigo	10 = Struct B_Hora
5 = Float	11 = Struct Item_Sel
6 = Double	

3- Posteriormente, serão definidos os tipos de colônias a que os objetos pertencem, para que dentro de cada uma possam ser incluídos os objetos com seus atributos e relacionamentos.

Considerando como no nosso exemplo, que existem dois objetos e os mesmos pertencem a colônia global, é necessário inicialmente ao se definir a colônia colocar o relacionamento com a modalidade HABITA, indicando todos os objetos pertencentes a colônia, juntamente com seu tipo, ou seja :

relacionamento habita colonia global;

Porém, considerando que seja incluído o objeto ATLETA em uma colônia AVALIAÇÃO, será necessário após o comando acima :

1o.) Criar a colônia :

objeto colonia avaliacao

2o.) Criar um relacionamento do tipo HABITADA_POR, quando há necessidade da criação da colônia; e HABITA quando a colônia já está corrente.

Além disso, é necessário definirmos o objeto que constringe o objeto criado :

relacionamento constrita objeto atleta

Obs.: Isso deve ser feito somente quando a colônia não for global, pois quando for o caso, já existirá um objeto alfa que a constringe. Para todo objeto que possuir um relacionamento com um objeto de uma outra colônia, deverá ter o relacionamento "constrita ".

E finalmente para identificar a criação de uma colônia, informamos :

propriedade indice 3;

Os índices utilizados devem ser a partir do número 3, indicando a posição onde a colônia será criada. Seguindo o mesmo esquema caso seja incluída uma nova colônia, esta deverá ter a informação:

propriedade indice 4;

indicando uma nova posição onde a colônia deve ser criada.

Na definição dos relacionamentos são especificadas as modalidades com seus respectivos relacionamentos.

Para isso é necessário :

Definição da Modalidade $\Rightarrow$ incluí-la como um objeto de tipo **modalidade1**, quando se tratar de um relacionamento binário e **modalidade2**, para relacionamento triplo. Como no nosso exemplo, estamos utilizando relacionamentos binários, então usamos :

objeto modalidade1 composta_por

Definição dos Relacionamentos $\Rightarrow$ Para localizar a modalidade (através do comando LOCALIZA)

localiza modalidade1 treinada_por

e definir o relacionamento oposto

relacionamento oposto modalidade1 treina

para finalmente definir o(s) relacionamento(s) existente(s), através dos seguintes comandos

objeto binário R1

relacionamento composto modalidade1 composta_por

relacionamento origem objeto equipe

relacionamento destino objeto atleta

4- O comando nome permite que sejam feitas variações na formatação do nome. Por exemplo :

4.1) Caso seja mostrar somente o nome :

nome so

baseado no objeto equipe, teremos o seguinte resultado :

equipe

4.2) Se a opção for pela visualização de tipo e nome :

nome

este comando trará o seguinte resultado :

(global) equipe

4.3) A visualização do tipo, contexto e nome é feita por :

nome relativo

causando a seguinte saída :

(global)./contexto/equipe)

4.4) A opção de visualização do tipo, contexto e nome é feita pelo comando :

nome absoluto

que resulta em :

[(global)]/contexto/equipe)

5- Depois de incluído no esquema, então podemos consultar o conteúdo de cada colônia com seus respectivos objetos, através do comando MOSTRA:

mostra Esquema

Além disso, é possível visualizar o objeto corrente através de :

mostra corrente

E finalmente, se quisermos visualizar algum objeto específico, basta informar o seu tipo :

mostra atleta

e serão mostrados todos os objetos do tipo Atleta.

Os nomes dos objetos impressos pelo comando **mostra**, dependem do formato indicado no comando **nome**.

A utilização de todos os comandos apresentados acima, podem ser reunidos para exemplificar um Esquema. O exemplo demonstrado abaixo é referente ao Esquema de uma Equipe de Voleibol.

Agora se quisermos incluir dados na base, utilizaremos os comandos definidos a seguir. Considerando que seja feita a inclusão de uma Equipe de Voleibol Feminina com seus respectivos atletas (com seus atributos).

1- Iniciamos incluindo as Equipes Banespa e Suzano, com seus respectivos atletas :

objeto equipe Banespa, Suzano

objeto atleta Milene, Claudia, Monica, Carla, Soraya, Marcia

2- Após a inclusão dos objetos, podemos iniciar a de atributos. Isto é feito por meio de vários comandos, dependendo do tipo de atributo do objeto.

2.1) Se for feita a inclusão de um atributo do tipo sinônimo :

propriedade nom_guerra fofão

Neste exemplo o atributo nom_guerra é um sinônimo, apesar de estar definido como propriedade. Isso ocorre pois no Esquema foi definido como uma propriedade com característica de sinônimo.

2.2) Caso seja feita a inclusão de atributos para o objeto atleta :

propriedade idade 20

propriedade altura 175

propriedade posicao meio-de-rede

propriedade funcao a

A inclusão dos atributos para os respectivos objetos, deve ser feita inicialmente através do posicionamento no objeto desejado.

Caso seja feita inclusão de objeto em outra colônia que não seja a global, é necessário utilizar o comando Considera que tem a função de requisitar acesso e abrir uma colônia. Esta é utilizada da seguinte forma :

considera Federacao equipe banespa

neste caso a colônia Federação estará aberta sendo constricta pelo objeto do tipo Equipe com nome Banespa. Um exemplo completo é mostrado a seguir.

3.2 Exemplos

Equipe de Voleibol

Esquema

liga meta

objeto objeto atleta equipe

localiza colonia global

relacionamento habitada_por objeto atleta equipe

objeto modalidade1 composta_por participa_de

localiza modalidade1 composta_por

relacionamento oposto modalidade1 participa_de

objeto binario R1

relacionamento composto modalidade1 composta_por

relacionamento origem objeto equipe

relacionamento destino objeto atleta

objeto binario R2

relacionamento composto modalidade1 participa_de

relacionamento origem objeto atleta

relacionamento destino objeto equipe

objeto modalidade1 treinada_por treina

localiza modalidade1 treinada_por

relacionamento oposto modalidade1 treina

objeto binario R3

relacionamento composto modalidade1 treinada_por

relacionamento origem objeto equipe

relacionamento destino objeto atleta

objeto binario R4

relacionamento composto modalidade1 treina

relacionamento origem objeto atleta

relacionamento destino objeto equipe

objeto atributo nome,cognome;

localiza atributo nome;

propriedade tipo_de_dado 7;

propriedade característica 2;
localiza atributo cognome;
propriedade tipo_de_dado 7;
propriedade característica 1;
localiza objeto atleta;
objeto atributo nome, idade, altura, funcao, posicao;
localiza atributo nome;
propriedade tipo_de_dado 2;
propriedade característica 2;
localiza atributo idade;
propriedade tipo_de_dado 2;
propriedade característica 2;
localiza atributo altura;
propriedade tipo_de_dado 2;
propriedade característica 2;
localiza atributo funcao;
propriedade tipo_de_dado 7;
propriedade característica 2;
localiza atributo posicao;
propriedade tipo_de_dado 7;
propriedade característica 2;
liga relacionamento
mostra esquema
desliga meta

Base de Dados

objeto atleta Claudia Milene Leila Monica Carolina Cibebe
localiza atleta Claudia
propriedade nom_guerra fofão
propriedade idade 20
propriedade altura 175
propriedade posicao meio-de-rede
propriedade funcao a

localiza atleta Milene
propriedade nom_guerra Mi
propriedade idade 21
propriedade altura 185
propriedade posicao saida
propriedade funcao a
localiza atleta Leila
propriedade nom_guerra Leiloca
propriedade idade 19
propriedade altura 165
propriedade posicao entrada
propriedade funcao a
localiza atleta Monica
propriedade nom_guerra Mo
propriedade idade 25
propriedade altura 165
propriedade posicao levantadora
propriedade funcao a
localiza atleta Carolina
propriedade nom_guerra Carol
propriedade idade 20
propriedade altura 175
propriedade posicao saida
propriedade funcao a
localiza atleta Cibebe
propriedade nom_guerra Bele
propriedade idade 20
propriedade altura 180
propriedade posicao meio-de-rede
propriedade funcao a

No exemplo acima vimos a descrição de um Esquema com a existência da colônia Global contendo dois objetos (Atleta e Equipe).

Abaixo veremos a utilização de outras colônias, ou seja, a criação de outras colônias sem ser a Global.

Considerando o seguinte esquema :

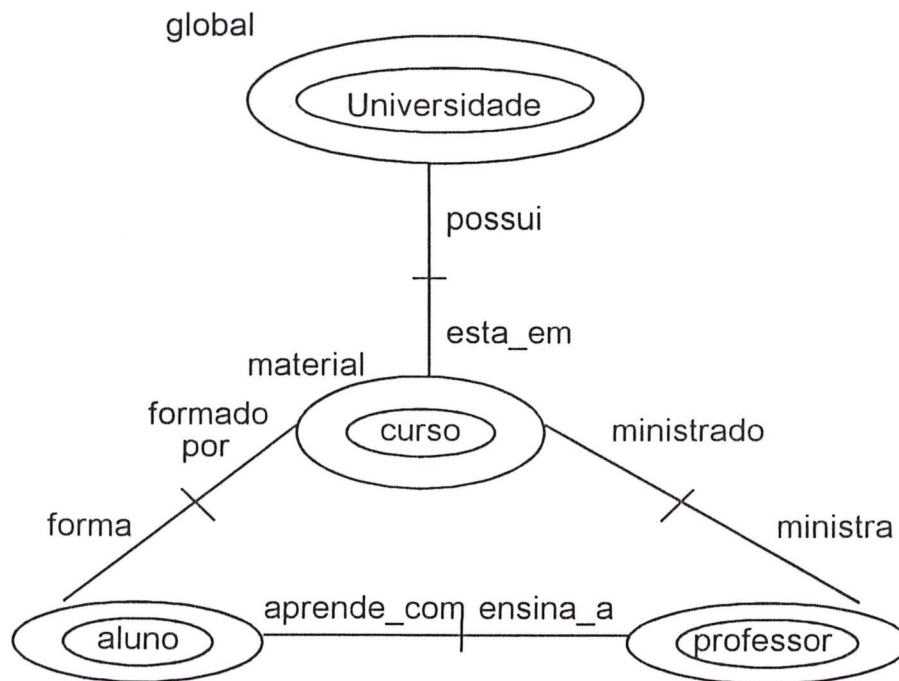


Figura 3.2 - Definição de um Esquema - Universidade

liga meta;

objeto objeto universidade curso aluno professor;

localiza objeto universidade;

relacionamento habita colonia global;

objeto colonia Material;

relacionamento habitada_por objeto curso;

relacionamento constrita objeto universidade;

propriedade indice 3;

objeto colonia Pessoal;

relacionamento habitada_por objeto aluno;

relacionamento constrita objeto curso;

relacionamento habitada_por objeto professor;

relacionamento constrita objeto curso;
propriedade indice 4;
objeto modalidade1 esta_em possui;
relacionamento oposto modalidade1 esta_em;
objeto binario relac1;
relacionamento composto modalidade1 possui;
relacionamento origem objeto universidade;
relacionamento destino objeto curso;
objeto binario relac2;
relacionamento composto modalidade1 possui;
relacionamento origem objeto universidade;
relacionamento destino objeto curso;
objeto modalidade1 forma formado_por;
relacionamento oposto modalidade1 forma;
objeto binario relac3;
relacionamento composto modalidade1 formado_por;
relacionamento origem objeto curso;
relacionamento destino objeto aluno;
objeto binario relac4;
relacionamento composto modalidade1 forma;
relacionamento origem objeto aluno;
relacionamento destino objeto curso;
objeto modalidade1 ensina_a aprende_com;
relacionamento oposto modalidade1 ensina_a;
objeto binario relac5;
relacionamento composto modalidade1 aprende_com;
relacionamento origem objeto aluno;
relacionamento destino objeto professor;
objeto binario relac6;
relacionamento composto modalidade1 ensina_a;
relacionamento origem objeto professor;
relacionamento destino objeto aluno;
objeto modalidade1 ministra ministrado_por;
relacionamento oposto modalidade1 ministra;
objeto binario relac7;

relacionamento composto modalidade1 ministrado_por;
relacionamento origem objeto curso;
relacionamento destino objeto professor;
objeto binario relac8;
relacionamento composto modalidade1 ministra;
relacionamento origem objeto professor;
relacionamento destino objeto curso;
objeto atributo area disciplina periodo tmp_duracao nr_cred_permitido especialidade nr_matricula
endereco nr_disciplinas vl_notas ds_disciplina carga_horaria tp_univers nr_alunos nr_cursos
ds_endere avalia;
localiza atributo area;
propriedade tipo_de_dado 7;
propriedade caracteristica 2;
localiza atributo disciplina;
propriedade tipo_de_dado 7;
propriedade caracteristica 2;
localiza atributo periodo;
propriedade tipo_de_dado 7;
propriedade caracteristica 2;
localiza atributo tmp_duracao;
propriedade tipo_de_dado 7;
propriedade caracteristica 2;
localiza atributo nr_cred_permitido;
propriedade tipo_de_dado 2;
propriedade caracteristica 2;
localiza atributo nr_cred_permitido;
propriedade tipo_de_dado 2;
propriedade caracteristica 2;
localiza atributo especialidade;
propriedade tipo_de_dado 7;
propriedade caracteristica 2;
localiza atributo nr_matricula;
propriedade tipo_de_dado 2;
propriedade caracteristica 1;
localiza atributo endereco;


```
    propriedade tipo_de_dado 7;
    propriedade caracteristica 2;
localiza atributo nr_disciplinas;
    propriedade tipo_de_dado 2;
    propriedade caracteristica 2;
localiza atributo vl_notas;
    propriedade tipo_de_dado 5;
    propriedade caracteristica 2;
localiza atributo ds_disciplina;
    propriedade tipo_de_dado 7;
    propriedade caracteristica 2;
localiza atributo carga_horaria;
    propriedade tipo_de_dado 2;
    propriedade caracteristica 2;
localiza atributo tp_univers;
    propriedade tipo_de_dado 7;
    propriedade caracteristica 2;
localiza atributo nr_alunos;
    propriedade tipo_de_dado 2;
    propriedade caracteristica 2;
localiza atributo nr_cursos;
    propriedade tipo_de_dado 2;
    propriedade caracteristica 2;
localiza atributo ds_endere;
    propriedade tipo_de_dado 7;
    propriedade caracteristica 2;
localiza atributo avalia;
    propriedade tipo_de_dado 7;
    propriedade caracteristica 2;
localiza objeto universidade;
    relacionamento atributos atributo tp_univers nr_alunos nr_cursos ds_endere avalia;
localiza objeto curso;
    relacionamento atributos atributo area disciplina periodo tmp_duracao nr_cred_permitido
especialidade;
localiza objeto aluno;
```


relacionamento atributos atributo nr_matricula endereco nr_disciplinas vl_notas;
relacionamento identificado atributo nr_matricula;
localiza objeto professor;
relacionamento atributos atributo ds_disciplina carga_horaria;
fim;

Base de Dados

objeto universidade usp ufscar unaerp unimep;
localiza universidade usp;
propriedade tp_univers estadual;
propriedade nr_alunos 113.490;
propriedade nr_cursos 25;
propriedade ds_endere rua vitoria giometti - 571;
propriedade avalia e;
localiza universidade ufscar;
propriedade tp_univers federal;
propriedade nr_alunos 107.231;
propriedade nr_cursos 31;
propriedade ds_endere rua dos anjos - 731;
propriedade avalia b;
localiza universidade unaerp;
propriedade tp_univers particular;
propriedade nr_alunos 93.118;
propriedade nr_cursos 40;
propriedade ds_endere rua das camélias-932;
propriedade avalia b;
localiza universidade unimep;
propriedade tp_univers particular;
propriedade nr_alunos 73.981;
propriedade nr_cursos 32;
propriedade ds_endere rua das hortencias-507;
propriedade avalia r;

localiza universidade puc;

- propriedade tp_univers particular;
- propriedade nr_alunos 131.437;
- propriedade nr_cursos 51;
- propriedade ds_endere rua das rosas-913;
- propriedade avalia b;

considera Material universidade usp;

objeto curso administracao analise educacao medicina;

localiza curso administracao;

- propriedade area humanas;
- propriedade disciplina adm.I calculo I;
- propriedade periodo noturno;
- propriedade tmp_duracao 4;
- propriedade nr_cred_permitido 40;
- propriedade especialidade financeira economica;

localiza curso analise;

- propriedade area exatas;
- propriedade disciplina estrutura dados algebra linear;
- propriedade periodo noturno;
- propriedade tmp_duracao 5;
- propriedade nr_cred_permitido 39;
- propriedade especialidade O&M sistemas;

localiza curso educacao;

- propriedade area humanas;
- propriedade disciplina atletismo voleibol;
- propriedade periodo diurno;
- propriedade tmp_duracao 3;
- propriedade nr_cred_permitido 40;
- propriedade especialidade condicionamento treinamento;

localiza curso medicina;

- propriedade area biologicas;
- propriedade disciplina biologia I anatomia I;
- propriedade periodo integral;
- propriedade tmp_duracao 6;
- propriedade nr_cred_permitido 60;

propriedade especialidade pediatria ginecologia;
considera Material universidade ufscar;
objeto curso direito computacao fisioterapia;
localiza curso direito;
 propriedade area humanas;
 propriedade disciplina tributação civil;
 propriedade periodo noturno;
 propriedade tmp_duracao 4;
 propriedade nr_cred_permitido 36;
 propriedade especialidade criminalidade tributário;
localiza curso computacao;
 propriedade area exatas;
 propriedade disciplina fundamentos I programação;
 propriedade periodo integral;
 propriedade tmp_duracao 4.5;
 propriedade nr_cred_permitido 46;
 propriedade especialidade hardware software;
localiza curso fisioterapia;
 propriedade area biologicas;
 propriedade disciplina genetica bio II;
 propriedade periodo integral;
 propriedade tmp_duracao 5;
 propriedade nr_cred_permitido 70;
 propriedade especialidade deficiencia fisica;
considera Material universidade unaerp;
objeto curso farmacia matematica jornalismo;
localiza curso farmacia;
 propriedade area biológica;
 propriedade disciplina introducao farmacia II;
 propriedade periodo integral;
 propriedade tmp_duracao 3;
 propriedade nr_cred_permitido 60;
 propriedade especialidade farmacêutica;
localiza curso matematica;
 propriedade area exatas;

propriedade disciplina algebra trigonometria;
propriedade periodo diurno;
propriedade tmp_duracao 4;
propriedade nr_cred_permitido 50;
propriedade especialidade topologia;
localiza curso jornalismo;
propriedade area humanas;
propriedade disciplina fotografia técnicas I;
propriedade periodo diurno;
propriedade tmp_duracao 4;
propriedade nr_cred_permitido 60;
propriedade especialidade radialista;
considera Material universidade unimep;
objeto curso odontologia fisica química;
localiza curso odontologia;
propriedade area biológicas;
propriedade disciplina odonto I ortodontia;
propriedade periodo integral;
propriedade tmp_duracao 5;
propriedade nr_cred_permitido 50;
propriedade especialidade cirurgia;
localiza curso fisica;
propriedade area exatas;
propriedade disciplina fisica I eletricidade;
propriedade periodo integral;
propriedade tmp_duracao 4;
propriedade nr_cred_permitido 40;
propriedade especialidade fisico;
localiza curso química;
propriedade area exatas;
propriedade disciplina orgânica inorgânica;
propriedade periodo integral;
propriedade tmp_duracao 5;
propriedade nr_cred_permitido 40;
propriedade especialidade químico industrial;

considera Material universidade usp;

localiza universidade usp;

considera Pessoal curso administracao;

objeto aluno ana bartira claudio;

localiza aluno ana;

propriedade nr_matricula 498488;

propriedade endereco rua visconde do rio branco 812;

propriedade nr_disciplinas 6;

propriedade vl_notas 5.3 6.2 7.1;

localiza aluno bartira;

propriedade nr_matricula 498487;

propriedade endereco rua visconde do rio azul 810;

propriedade nr_disciplinas 4;

propriedade vl_notas 5.1 4.2 9.1;

localiza aluno claudio;

propriedade nr_matricula 493.312;

propriedade endereco rua xv de abril 701;

propriedade nr_disciplinas 8;

propriedade vl_notas 8.1 4.0 3.5;

considera Pessoal curso analise;

objeto aluno daniel eduardofabricio;

localiza aluno daniel;

propriedade nr_matricula 593.312;

propriedade endereco rua xv de maio 781;

propriedade nr_disciplinas 10;

propriedade vl_notas 9.1 2.0 7.5;

localiza aluno eduardo;

propriedade nr_matricula 793.312;

propriedade endereco rua xv de junho 671;

propriedade nr_disciplinas 9;

propriedade vl_notas 9.4 6.0 2.5;

localiza aluno fabricio;

propriedade nr_matricula 793.390;

propriedade endereco rua xv de julho 871;

propriedade nr_disciplinas 4;

propriedade vl_notas 9.4 6.7 9.5;

considera Pessoal curso educacao;

objeto aluno gisele helenia ieda;

localiza aluno gisele;

propriedade nr_matricula 793.356;

propriedade endereco rua xv de agosto 471;

propriedade nr_disciplinas 8;

propriedade vl_notas 9.4 6.6 9.9;

localiza aluno helenia;

propriedade nr_matricula 793.453;

propriedade endereco rua xv de setembro 971;

propriedade nr_disciplinas 6;

propriedade vl_notas 9.2 6.1 9.9;

localiza aluno ieda;

propriedade nr_matricula 793.487;

propriedade endereco rua xv de outubro 947;

propriedade nr_disciplinas 9;

propriedade vl_notas 9.2 6.2 1.9;

considera Material universidade usp;

localiza universidade usp;

considera Pessoal curso medicina;

objeto aluno joao laura marisa;

localiza aluno joao;

propriedade nr_matricula 793.123;

propriedade endereco rua xv de novembro 437;

propriedade nr_disciplinas 5;

propriedade vl_notas 9.0 6.1 2.9;

localiza aluno laura;

propriedade nr_matricula 793.156;

propriedade endereco rua xv de dezembro 127;

propriedade nr_disciplinas 4;

propriedade vl_notas 8.0 6.3 2.9;

localiza aluno marisa;

propriedade nr_matricula 793.000;

propriedade endereco rua xv de janeiro 907;

propriedade nr_disciplinas 6;
propriedade vl_notas 8.0 6.0 9.9;
considera Material universidade ufscar;
localiza universidade ufscar;
considera Pessoal curso direito
objeto aluno nedda osvaldo patricia
localiza aluno nedda;
propriedade nr_matricula 342.586;
propriedade endereco rua sete de setembro 831;
propriedade nr_disciplinas 7;
propriedade vl_notas 7.3 3.9 4.7;
localiza aluno osvaldo;
propriedade nr_matricula 265.713;
propriedade endereco rua duque de caxias 810;
propriedade nr_disciplinas 5;
propriedade vl_notas 1.2 7.3 8.5;
localiza aluno patricia;
propriedade nr_matricula 310.437;
propriedade endereco rua das alvoradas 908;
propriedade nr_disciplinas 4;
propriedade vl_notas 7.2 9.1 4.6;
considera Pessoal curso computacao
objeto aluno roberta samanta teobaldo;
localiza aluno roberta;
propriedade nr_matricula 310.123;
propriedade endereco rua das camélias 208;
propriedade nr_disciplinas 6;
propriedade vl_notas 7.2 9.0 8.6;
localiza aluno samanta;
propriedade nr_matricula 510.123;
propriedade endereco rua das orquideas 508;
propriedade nr_disciplinas 9;
propriedade vl_notas 7.4 9.6 8.6;
localiza aluno teobaldo;
propriedade nr_matricula 510.413;

propriedade endereco rua das flores 808;
propriedade nr_disciplinas 2;
propriedade vl_notas 7.4 9.8 8.6;
considera Pessoal curso fisioterapia;
objeto aluno adriana barbara camila;
localiza aluno adriana;
propriedade nr_matricula 510.456;
propriedade endereco rua das rosas 608;
propriedade nr_disciplinas 3;
propriedade vl_notas 7.4 9.9 1.6;
localiza aluno barbara;
propriedade nr_matricula 545.780;
propriedade endereco rua das pupilas 508;
propriedade nr_disciplinas 8;
propriedade vl_notas 7.4 9.1 1.6;
localiza aluno camila;
propriedade nr_matricula 545.320;
propriedade endereco rua das parreiras 808;
propriedade nr_disciplinas 6;
propriedade vl_notas 7.0 1.1 1.6;
considera Material universidade unaerp;
localiza universidade unaerp;
considera Pessoal curso farmacia;
objeto aluno dener everton fabiana;
localiza aluno dener;
propriedade nr_matricula 545.123;
propriedade endereco rua das pilastras 124;
propriedade nr_disciplinas 8;
propriedade vl_notas 7.0 1.5 1.6;
localiza aluno everton;
propriedade nr_matricula 545.125;
propriedade endereco rua das luzes 624;
propriedade nr_disciplinas 4;
propriedade vl_notas 7.0 1.7 9.6;
localiza aluno fabiana;

propriedade nr_matricula 549.908;
propriedade endereco rua das madeiras 424;
propriedade nr_disciplinas 6;
propriedade vl_notas 7.9 1.7 9.6;
considera Pessoal curso matematica;
objeto aluno gabriela helen igor;
localiza aluno gabriela;
propriedade nr_matricula 549.120;
propriedade endereco rua II de janeiro 924;
propriedade nr_disciplinas 8;
propriedade vl_notas 7.9 1.0 8.6;
localiza aluno helen;
propriedade nr_matricula 549.156;
propriedade endereco rua II de fevereiro 624;
propriedade nr_disciplinas 9;
propriedade vl_notas 7.9 3.0 8.6;
localiza aluno igor;
propriedade nr_matricula 549.845;
propriedade endereco rua II de marco 434;
propriedade nr_disciplinas 3;
propriedade vl_notas 7.9 9.0 8.6;
considera Pessoal curso jornalismo;
objeto aluno jorge lelis mauro;
localiza aluno jorge;
propriedade nr_matricula 549.545;
propriedade endereco rua II de abril 734;
propriedade nr_disciplinas 3;
propriedade vl_notas 7.9 9.1 8.6;
localiza aluno lelis;
propriedade nr_matricula 349.545;
propriedade endereco rua II de maio 834;
propriedade nr_disciplinas 4;
propriedade vl_notas 7.9 9.9 8.6;
localiza aluno mauro;
propriedade nr_matricula 349.125;

propriedade endereco rua II de junho 334;
propriedade nr_disciplinas 4;
propriedade vl_notas 7.9 9.5 8.6;
considera Material universidade unimep;
localiza universidade unimep;
considera Pessoal curso odontologia;
objeto aluno nair oto paulo;
localiza aluno nair;
propriedade nr_matricula 149.125;
propriedade endereco rua II de julho 734;
propriedade nr_disciplinas 5;
propriedade vl_notas 7.9 9.9 8.6;
localiza aluno oto;
propriedade nr_matricula 249.125;
propriedade endereco rua II de agosto 534;
propriedade nr_disciplinas 3;
propriedade vl_notas 7.9 1.2 8.6;
localiza aluno paulo;
propriedade nr_matricula 149.125;
propriedade endereco rua II de setembro 734;
propriedade nr_disciplinas 4;
propriedade vl_notas 7.9 2.3 8.6;
considera Pessoal curso fisica;
objeto aluno queli renata silmara;
localiza aluno queli;
propriedade nr_matricula 129.125;
propriedade endereco rua II de outubro 634;
propriedade nr_disciplinas 4;
propriedade vl_notas 7.9 2.5 8.6;
localiza aluno renata;
propriedade nr_matricula 129.455;
propriedade endereco rua II de novembro 934;
propriedade nr_disciplinas 4;
propriedade vl_notas 7.9 2.5 7.6;
localiza aluno silmara;

propriedade nr_matricula 429.455;
propriedade endereco rua II de dezembro 634;
propriedade nr_disciplinas 4;
propriedade vl_notas 7.1 2.5 7.6;
considera Pessoal curso química;
objeto aluno tatiana valeria zanetti;
localiza aluno tatiana;
propriedade nr_matricula 429.433;
propriedade endereco rua VII de janeiro 634;
propriedade nr_disciplinas 4;
propriedade vl_notas 7.1 2.5 7.6;
localiza aluno valeria;
propriedade nr_matricula 429.433;
propriedade endereco rua VII de janeiro 634;
propriedade nr_disciplinas 4;
propriedade vl_notas 7.1 2.5 7.6;
localiza aluno zanetti;
propriedade nr_matricula 329.433;
propriedade endereco rua VII de fevereiro 734;
propriedade nr_disciplinas 3
propriedade vl_notas 7.1 2.7 7.6;
considera Material universidade usp;
considera Pessoal curso administracao;
objeto professor renata ericsson;
localiza professor renata;
propriedade ds_disciplina administracao I;
propriedade carga_horaria 40;
localiza professo ericsson;
propriedade ds_disciplina introducao;
propriedade carga_horaria 20;
considera Pessoal curso analise;
objeto professor ricardo spencer;
localiza professor ricardo;
propriedade ds_disciplina algebra;
propriedade carga_horaria 24;

localiza professor spencer;
 propriedade ds_disciplina estrutura de dados;
 propriedade carga_horaria 15;
considera Pessoal curso educacao;
objeto professor michele stephanie;
localiza professor michele;
 propriedade ds_disciplina atletismo;
 propriedade carga_horaria 40;
localiza professor stephanie;
 propriedade ds_disciplina voleibol;
 propriedade carga_horaria 40;
considera Pessoal curso medicina;
objeto professor aureo rene;
localiza professor aureo;
 propriedade ds_disciplina anatomia;
 propriedade carga_horaria 40;
localiza professor rene;
 propriedade ds_disciplina cirurgia;
 propriedade carga_horaria 50;
considera Pessoal curso direito;
objeto professor fernando rafael;
localiza professor fernando;
 propriedade ds_disciplina tributário;
 propriedade carga_horaria 40;
localiza professor rafael;
 propriedade ds_disciplina civil;
 propriedade carga_horaria 50;
considera Pessoal curso computacao;
objeto professor catarina gabriel;
localiza professor catarina;
 propriedade ds_disciplina fundamentos I;
 propriedade carga_horaria 40;
localiza professor rafael;
 propriedade ds_disciplina computacao;
 propriedade carga_horaria 50;

considera Pessoal curso fisioterapia;
objeto professor andréa milton;
localiza professor andréa;
 propriedade ds_disciplina deficiências I;
 propriedade carga_horaria 40;
localiza professor milton;
 propriedade ds_disciplina articulação;
 propriedade carga_horaria 50;
considera Pessoal curso farmácia;
objeto professor weber ivan;
localiza professor weber;
 propriedade ds_disciplina remédios;
 propriedade carga_horaria 40;
localiza professor ivan;
 propriedade ds_disciplina farmácia;
 propriedade carga_horaria 50;
considera Pessoal curso matemática;
objeto professor ricardo rodrigo;
localiza professor ricardo;
 propriedade ds_disciplina algebra;
 propriedade carga_horaria 40;
localiza professor rodrigo;
 propriedade ds_disciplina topologia;
 propriedade carga_horaria 50;
considera Pessoal curso jornalismo;
objeto professor denise lucimara;
localiza professor denise;
 propriedade ds_disciplina fotografia;
 propriedade carga_horaria 40;
localiza professor lucimara;
 propriedade ds_disciplina jornal I;
 propriedade carga_horaria 50;
considera Pessoal curso odontologia;
objeto professor milene mônica;
localiza professor milene;

propriedade ds_disciplina cirurgia;
propriedade carga_horaria 40;
localiza professor mônica;
propriedade ds_disciplina ortodontia;
propriedade carga_horaria 50;
considera Pessoal curso fisica;
objeto professor carlos alberto;
localiza professor carlos;
propriedade ds_disciplina eletricidade;
propriedade carga_horaria 40;
localiza professor alberto;
propriedade ds_disciplina fisica I;
propriedade carga_horaria 50;
considera Pessoal curso química;
objeto professor roberto macal;
localiza professor roberto;
propriedade ds_disciplina orgânica;
propriedade carga_horaria 40;
localiza professor macal;
propriedade ds_disciplina química;
propriedade carga_horaria 50;

Capítulo 4

Rotinas de Consulta ao Esquema e Acesso a Objetos

Capítulo 4

4.1- Utilização das Rotinas

4.1.1- Obtenção do nome dos tipos pertencentes a uma colônia :

Existem rotinas que permitem o retorno de nomes de tipos de objetos, modalidades, atributos, sinônimos, colônias e relacionamentos triplos. Por exemplo, considerando que gostaríamos de saber :

NOMOBJ $\Rightarrow$ informa o nome do tipo do objeto, encontrado através do tipo informado.

NOMMOD $\Rightarrow$ retorna o nome da modalidade, que é localizada pelo código do tipo de modalidade.

NOMATR $\Rightarrow$ o nome do atributo é conseguido através da informação do código do tipo de atributo.

NOMSIN $\Rightarrow$ informa o nome do tipo do sinônimo, encontrado através do tipo informado.

NOMCOL $\Rightarrow$ retorna o nome da colônia, que é localizada pelo código do tipo de colônia.

NOMTRI $\Rightarrow$ o nome do relacionamentos triplo é conseguido através da informação do código do tipo de relacionamento triplo.

Considerando que há necessidade de se fazer uma seleção de objetos do mesmo tipo, então bastaria utilizar a rotina NOMOBJ. Porém, dentre os parâmetros desta rotina está o código do tipo de objeto, o qual é gerado internamente. Este por sua vez pode ser lido através da rotina CODOBJ (descrito a frente), que retorna este código, logo para que utilizemos a rotina NOMOBJ, é necessário chamarmos primeiramente a rotina CODOBJ.

Exemplo : Listar todos os nomes de tipos de objetos.

Como no início do programa houve o uso da rotina CODOBJ, retornando o código do tipo correspondente ao objeto, então para listar os nomes basta utilizar NOMOBJ na Listagem Individual de Equipes.

A seleção de nomes de tipos de modalidades é feita pela rotina NOMMOD, a qual tem que obrigatoriamente ter a mesma sequência descrita para a rotina NOMOBJ, ou seja, primeiro utilizar a rotina CODMOD para localização do código e posteriormente então utilizar NOMMOD.

Exemplo : Listar todos os nomes de tipos de modalidade

Para essa operação, basta fazer uma seleção através do nome retornado pela rotina NOMMOD.

A rotina NOMATR tem várias utilidades, porém considerando que o usuário queira saber somente determinados atributos de certos objetos, como por exemplo, selecionar todas as alturas dos atletas para verificar a média de altura do time, podemos utilizar esta rotina não se esquecendo que a rotina correspondente, CODATR, deve ser anteriormente utilizada.

No caso de querermos saber qual o nome principal e qual o sinônimo, podemos utilizar a rotina NOMSIN que retorna o seu tipo.

Exemplo : Supondo que uma equipe possua os seguintes nomes :

- 1) " Equipe Juvenil de Voleibol Feminino - 1994 - Ubatuba "
- 2) " Morada do Sol "
- 3) " As Meninas de Ouro "

Neste exemplo, o nome principal seria o sinônimo corrente, " Equipe Juvenil de Voleibol Feminino - 1994 - Ubatuba ", que é o primeiro nome, já em outro momento pode ser o terceiro.

4.1.2 - Obtenção do código dos tipos pertencentes a uma colônia :

A obtenção do código dos tipos é necessária, devido a este ser gerado internamente. Logo, para que este seja utilizado em outras rotinas é fundamental defini-lo. Isso pode ser feito com o auxílio de muitas rotinas, entre elas :

CODOBJ $\Rightarrow$ Obtém o código do tipo de objeto, através do nome deste.

CODMOD1 ou CODMOD2 $\Rightarrow$ Obtém o código do tipo da modalidade, através do nome desta.

CODATR $\Rightarrow$ Obtém o código do tipo de atributo, através do nome deste.

CODSIN $\Rightarrow$ Obtém o código do tipo de sinônimo, através do nome deste.

CODCOL $\Rightarrow$ Obtém o código do tipo de colônia, através do nome desta.

CODTRI $\Rightarrow$ Obtém o código do tipo de relacionamento triplo, através do nome deste.

Considerando que se faça uma seleção de todos os objetos que possuírem o mesmo código.

Exemplo : Equipe - Selecionar atletas da Equipe definida pelo usuário :

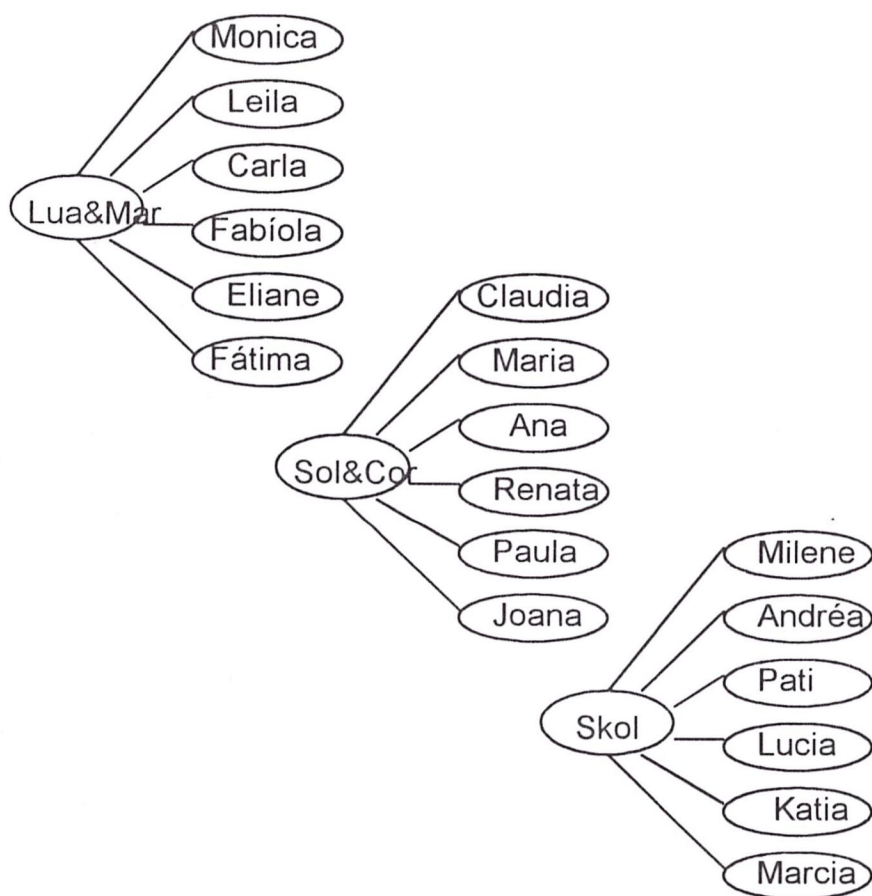


Figura 4.1 - Seleção de Vólibol

Se o usuário quiser a listagem apenas da equipe " Skol ", basta selecionar o código do tipo de objeto Equipe, ou seja :

Utilizando CODOBJ :

1) Informação dos parâmetros :

cadeia nome = skol

* t_obj = retorna o código referente a equipe.

Podemos além disto criar um objeto, com o auxílio desta rotina.

Obs.: Verificar exemplo com utilização desta rotina no tópico 4.2.1.1 ("Como criar um objeto").

A rotina CODMOD1 é utilizada quando por exemplo, desejamos incluir um relacionamento binário em um objeto; já no caso de inclusão de um relacionamento triplo, utilizamos CODMOD2. A inclusão é feita através da rotina CTREL, que pede como parâmetro o tipo de relacionamento, daí a utilidade de CODMOD1.

Utilizando CODMOD1 :

1) Informação dos parâmetros :

cadeia nome = composto_por

* t_mod = retorna o tipo de relacionamento, utilizado em CTREL
(inclusão de relacionamento binário).

Obs.: A utilização desta rotina pode ser vista no tópico 4.2.1.2 ("Como criar relacionamento binário ou triplo entre dois objetos").

Este pode ser utilizado também como mostrado no exemplo de Equipes, onde usamos a rotina TAREL (posiciona no primeiro relacionamento desse tipo) para selecionar o relacionamento desejado.

Obs.: O exemplo da utilização desta rotina está no tópico 4.2.5.2 ("Como posicionar em um relacionamento").

Neste exemplo, poderia ser utilizada a rotina PDREL (posiciona no relacionamento binário indicado), porém o relacionamento posicionado não é necessariamente o primeiro, o que o torna dispensável nessa aplicação.

A rotina CODATR permite que seja feita a inclusão de atributos em um determinado objeto. No exemplo referente a EQUIPE, foi utilizada na inclusão dos seguintes atributos :

- posição
- função
- altura
- idade

no objeto ATLETA.

Isso foi possível através das rotinas CTPRO, utilizada para inclusão de atributos do tipo propriedade. Como é necessária a informação do código do tipo de atributo, então utilizamos CODATR que faz seu retorno.

Utilizando CODATR :

1) Informações dos parâmetros :

cadeia nome = função

*t_atr = retorna o código referente ao atributo função.

O uso da rotina CODSIN é feito quando necessitamos do código referente ao sinônimo de um objeto. Por exemplo, supondo que Equipe possua :

- Nome Principal = Equipe Juvenil de Voleibol Feminino - 1994 - Ubatuba
- Sinônimo = Morada do Sol

Para inclusão do sinônimo através da rotina CTSIN, é necessário termos o código do tipo deste. Logo, utilizamos a rotina CODSIN para que haja o seu retorno.

Utilizando CODSIN :

1) Informação dos parâmetros :

cadeia nome = cognome

* t_sin = retorna o código do tipo do sinônimo.

A utilização deste pode ser comprovada no exemplo de Equipes, através da rotina CTSIN para inclusão do sinônimo referente a equipe.

Obs.: A utilização destas rotinas é mostrada no tópico 4.2.1.4 ("Como colocar um atributo em um objeto").

A utilização das rotinas CODCOL e CODTRI, podem ser úteis na operação com rotinas como : ANCOL, ZNCOL, PNCOL, XNCOL, CNCOL, BNCOL, B_BFCOL usadas na manipulação de operações com colônias, e CTTRI, BDTRI, VNTRI, PDTRI, RNTRI usadas na manipulação de operações com relacionamentos triplos.

4.1.3 -Parâmetros Globais :

Essas rotinas são utilizadas para operações relacionadas a parâmetros globais.

B_GLOBAL ⇒ Permite que seja definido um parâmetro global.

B_LEPGLOBAL ⇒ Faz a leitura de um parâmetro global.

B_RPGLOBAL ⇒ Esta rotina faz a restauração de todos os parâmetros globais.

B_XPGLOBAL ⇒ A rotina B_xpglobal é utilizada para permutar parâmetros globais.

B_INICPGLOBAL ⇒ Realiza o salvamento dos parâmetros globais atuais e faz sua inicialização.

A rotina que permite estabelecer parâmetros globais, possui três parâmetros. Entre eles existe :

1o.) Este parâmetro pode ser :

- **Colônia Ativa**

Muitas vezes a colônia a ser acessada pode não estar ativa, logo para isso é necessário que se defina no parâmetro global se esta ficará ou não ativa. Quando determinamos que a colônia ficará ativa, é necessário definir qual será a colônia (nome). Caso contrário, ou seja, quando não está ativa, não é necessário definir o parâmetro Valor.

Utilizando B_Global :

1) Informação dos parâmetros :

pal2 param = colativa

pal2 estado = on

long valor = co (nome da colônia)

ou

pal2 param = colativa

pal2 estado = off

long valor = não incluir

- Nível Semântico

Este parâmetro define o nível semântico que será analisado.

Utilizando B_Global :

1) Informação dos parâmetros :

pal2 param = nivseman

pal2 estado = on

long valor = sintaxe

ou

pal2 param = nivseman

pal2 estado = on

long valor = básico

ou

pal2 param = nivseman

pal2 estado = on

long valor = pleno
ou
pal2 param = nivseman
pal2 estado = on
long valor = multi_min
ou
pal2 param = nivseman
pal2 estado = off
long valor = não é incluído

- Ordem

Podemos através deste parâmetro definir em qual ordem será inserido o objeto :

Início - o objeto é inserido no início.

Posição - indica a posição onde o objeto será inserido.

Final - o objeto é inserido no final.

Utilizando B_global :

1) Informação dos parâmetros :

pal2 param = ordem
pal2 estado = on
long valor = início
ou
pal2 param = ordem
pal2 estado = on
long valor = posição
ou
pal2 param = ordem
pal2 estado = on
long valor = final
ou
pal2 param = ordem

pal2 estado = off

long valor = não é informado

- Tripla

No parâmetro Tripla, é definido a manipulação de relacionamentos triplos ou binários dependendo da necessidade.

Utilizando B_Global :

1) Informação dos parâmetros :

pal2 param = tripla

pal2 estado = on

ou

pal2 param = tripla

pal2 estado = off

Para manipulação do relacionamento triplo, utilizamos o parâmetro TRIPLA ON.

Considerando a inclusão de um relacionamento por exemplo :

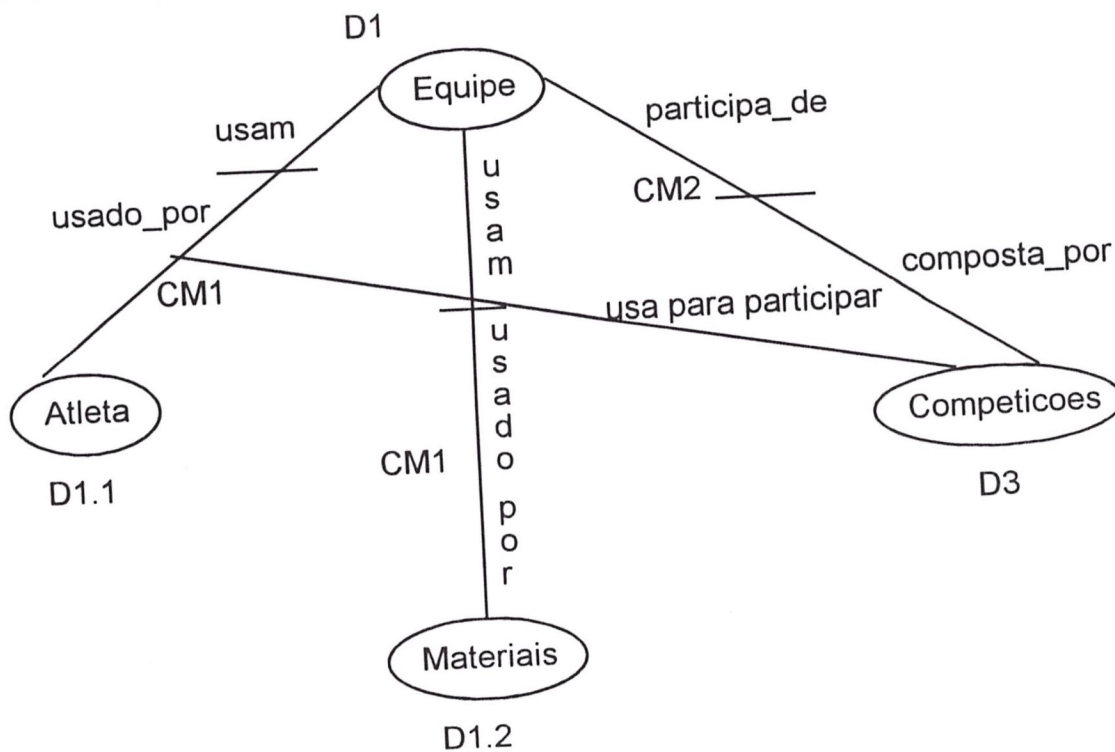


Figura 4.2 - Inclusão de Relacionamentos

`pnobj (d1); /* Posiciona no objeto da divisão 1, ou seja, o objeto origem */`

`cttri (cm1, d3);`

`B_Global (tripla, on);`

`ctrel (cm1, d11);`

`ctrel (cm1, d12);`

Ao incluirmos um relacionamento triplo, é necessário inicialmente definir a existência do relacionamento triplo, para posteriormente definir os relacionamentos binários que fazem parte deste. Isso é possível através do parâmetro tripla da rotina `B_Global`, onde podemos acessar os relacionamentos.

- Meta

Acessa o esquema ou o meta-esquema. Em determinados casos, por exemplo, há utilização das rotinas de acesso a base de dados no esquema. Logo, utilizamos o parâmetro `Meta`, que permite o acesso a uma determinada colônia do esquema.

Utilizando B_global :

1) Informação dos parâmetros :

pal2 param = meta
pal2 estado = on
long valor = co (nome da colônia)
ou
pal2 param = meta
pal2 estado = off
long valor = não é informado

- Gerente

Determina o privilégio de acesso aos usuários. O parâmetro referente ao gerente, permite que haja determinação de prioridades a determinados usuários.

Utilizando B_global :

1) Informação dos parâmetros :

pal2 param = gerente
pal2 estado = on
long valor = normal
ou
pal2 param = gerente
pal2 estado = on
long valor = geral
ou
pal2 param = gerente
pal2 estado = off
long valor = não é informado

Utilização das Rotinas referentes aos parâmetros globais :

Considerando a existência do padrão dos parâmetros globais. Supondo que necessitemos alterar alguns deles, para uso exclusivo de alguma rotina, então utilizaremos as rotinas referente aos parâmetros globais.

Exemplo :

```
main ( ) {  
    ...  
    b_global (colativa, on, equipe);  
    b_global (ordem, on, inicio);  
    b_global (meta, on, equipe);  
    ...  
    chamada de rotina
```

```
B_inicpglobal  
B_global (meta,off)  
    ...  
B_xpglobal  
    ...
```

Figura 4.3 - Parâmetros Globais

Supondo que exista uma função principal e nela existam parâmetros globais estabelecidos. Porém, no meio da função principal há a chamada de uma rotina que necessita de alteração em alguns parâmetros. Logo, para isso é necessário o uso das seguintes rotinas :

B_inicpglobal, que faz a inicialização dos parâmetros no endereço informado. Então, estabelecemos os novos parâmetros a serem utilizados por esta rotina, através de B_global :

```
B_global (meta, off);  
B_global (ordem, on, final);  
    ...
```

Lemos o valor através da rotina B_lepglobal, e finalmente utilizamos B_xpglobal, para que esta ignore os padrões estabelecidos após seu uso pela rotina, e retorne ao valor default.

4.2 - Operação com Rotinas

4.2.1 - Criação

As rotinas de criação possuem operações de inclusão de um relacionamento ou atributo em um objeto.

4.2.1.1- Como criar um objeto

A aplicação desta pode ser vista no programa referente a Equipes, na rotina referente a Inclusão de Atletas, onde é utilizada CTOBJ (cria um objeto) para incluir o nome do atleta, que é considerado um objeto.

Definição dos parâmetros :

Para a rotina CODOBJ

cadeia nome = "nome"

*t_obj = &nome

Para a rotina CTOBJ

cadeia nome = nom_atl

t_obj = atleta

Baseado no exemplo " Equipes " :

```
inc_atl ( ) { /* Inclusão de Atletas */
```

definição de variáveis

```
while (nome != " ") {  
    printf(" Nome : ");  
    getk (nom_atl);
```

```

erro = ctobj (nom_atl, atleta);
if (erro == 0) {
    printf ("Funcao : ");
    getk (fun_atl);
    erro = ctobj (fun_atl, atleta);
    ...

```

4.2.1.2 - Como criar relacionamento binário ou triplo entre dois objetos :

A criação de um relacionamento binário ou triplo, é feita pelas rotinas CTREL e CTTRI respectivamente. Porém o seu uso é possível através das rotinas CODMOD1 e CODMOD2, pois elas retornam o código interno o qual deve ser passado as rotinas CTREL e CTTRI. No exemplo foi incluído um relacionamento do tipo " participa_de", no objeto Atleta.

Definição dos parâmetros :

Relacionamento Binário

Para a rotina CODMOD1

```

cadeia nome = "participa_de"
*t_mod = &modal

```

Para a rotina CTREL

```

t_rel = modal
codest = equipe

```

Baseado no exemplo referente a Equipes :

```

inc_atl ( ) { /* Inclusão de Atletas */

```

definição de variáveis

```

while (nome != " ") {
    if (erro == 0) {
        ctpro (pos_atl, posicao);
    }
}

```

```
ctrel (participa_de, equipe); /*Inclui o relacionamento "Atleta
participa_de Equipe" */
```

...

Relacionamento Triplo

Para a rotina CODMOD2

```
cadeia nome = "ligada_a"
*t_mod = &modal
```

Para a rotina CTTRI

```
t_rel = modal
codest = federação
```

Considerando por exemplo que seja incluído um relacionamento "ligada_por":

```
inc_atl ( ) { /* Inclusão de Atletas */
...
definição de variáveis
codmod2 (ligada_a, &moda_2);
...
while (nome != "    ") {
    if (erro == 0) {
        ctrel (ligada_a, federacao); /* Inclui o relacionamento "Equipe ligada
a Federacao" */
```

4.2.1.3- Como criar colônias :

A criação de colônias é feita pela rotina CNCOL. Porém é passado para essa função o código do tipo de colônia, o qual é obtido somente através da rotina CODCOL.

Exemplo : Supondo que no programa Equipe fosse incluído a colônia Federação.

Definição dos parâmetros :

Para a rotina CODCOL


```
cadeia nome = "Federação"  
    * tcol = &federa
```

Para a rotina CNCOL

```
tcol = federa
```

```
inc_col ( ) { /* Inclusão de Colônias */  
    ...  
    definição de variáveis  
    ...  
    cncol (materiais); /* Inclui a colônia especificada */
```

4.2.1.4 - Como colocar um atributo em um objeto :

Dentre os tipos de atributos existentes temos

- sinônimo
- comentário
- regras
- procedimento
- tempo
- estrutura de dados
- gráfico
- imagens
- som
- visualização
- propriedade
- partitura

Sendo que para cada tipo há uma rotina correspondente. No exemplo de Equipes, foi utilizado o tipo de atributo propriedade e sinônimo.

Para estes dois tipos de atributos são utilizadas as rotinas CTPRO e CTSIN respectivamente. Porém, na passagem de parâmetros para estas funções é informado o código do tipo, o qual é obtido pelas rotinas CODATR e CODSIN. O exemplo mostra a inclusão do atributo Posição no objeto Atleta, e para o atributo Cognome no objeto Equipe.

Definição dos parâmetros :

Para a rotina CODATR

cadeia nome = "posicao"

*t_atr = posicao

Para a rotina CTPRO

union B_valores valor = fun_atl

t_atr = atleta

Para a rotina CODSIN

cadeia nome = "cognome"

*t_sin = cognome

Para a rotina CTSIN

cadeia nome = &cognome

t_sin = &apelido

Baseado no exemplo referente a Equipes :

inc_atl () { /* Inclusão de Atletas */

definição de variáveis

...

while (nome != " ") {

printk ("Funcao :");

getk (fun_atl);

erro = ctobj(fun_atl, atleta);

if (erro == 0) {

ctpro (fun_atl, funcao); /* Inclusão da propriedade referente a

função atleta */

4.2.1.5 - Como colocar um atributo em um relacionamento :

Dentre os tipos de atributos existentes temos

- sinônimo
- comentário
- regras
- procedimento
- tempo
- estrutura de dados
- gráfico
- imagens
- som
- visualização
- propriedade
- partitura

sendo que para cada tipo há uma rotina correspondente. No exemplo de Equipes, foi utilizado o tipo de atributo propriedade e sinônimo. O exemplo a ser mostrado, utilizará um atributo do tipo propriedade.

Neste caso a inclusão de um atributo em um relacionamento, é feita através da definição do código do tipo de atributo, que é obtido com a rotina CODATR. Depois, este código é passado para a rotina CTPRO.

No exemplo utilizado foi incluído um atributo `Dur_Treino` que possui o conteúdo referente a horas de duração de treino.

Definição dos parâmetros :

Para a rotina CODATR

cadeia nome = " joga 3 horas diárias "

* t_atr = & relbin

Para a rotina CTPRO

union b_valores valor = tip_rel

t_atr = relbin

Considerando que seja incluído um atributo "Duração_do_Treino" no relacionamento treinada_por.

```
inc_equ( ) { /* Inclusão de Equipes */  
    definição de variáveis  
    ...  
    codatr ("joga 3 horas diárias", &relbin);  
    ...  
    ctrel ("treinada_por", atleta);  
    ctpro ("treinada_por", relbin); /* Inclui o atributo no relacionamento  
                                   indicado*/
```

4.2.2 - Exclusão

As rotinas de exclusão de objetos são utilizadas para retirar objetos, atributos de objeto e/ou relacionamento.

4.2.2.1 - Como retirar um objeto :

A exclusão de um objeto é feita pela rotina RNOBJ.

Definição de parâmetros :

Nesta rotina não há definição de parâmetros.

4.2.2.2 - Como retirar um atributo de um objeto :

Para a retirada de um atributo de um objeto, é necessário o posicionamento no objeto. Para isso, seria necessário a utilização de várias rotinas :

```
codobj (cognome, &cogno)
```

...
ptobj (cognome, cogno)

...
rnsin(); /* Retira o sinônimo referente ao objeto posicionado */

O exemplo mostrado se refere a retirada de atributo do tipo sinônimo. Caso seja feita retirada de outro tipo de atributo, como por exemplo Propriedade, então utilizar

rnpro(); /* Retira a propriedade referente ao objeto posicionado */

O mesmo é feito para o comentário e outros tipos de atributos, cada qual com sua rotina respectiva.

4.2.2.3 - Como retirar um relacionamento binário :

Para a retirada de um relacionamento binário de um objeto, é necessário o posicionamento no objeto. Para isso, seria necessário a utilização de várias rotinas

codmod1 (descrição, &modal)

...
tarel (composto_por)

...
rnrel(); /* Retira o Relacionamento Binário */

O exemplo mostrado se refere a retirada de relacionamento do tipo binário.

4.2.2.4 - Como retirar um relacionamento triplo :

Para a retirada de um relacionamento triplo de um objeto, é necessário o posicionamento no objeto. Para isso, seria necessário a utilização de várias rotinas

codmod2 (descrição, &modal)

...
tarel (composto_por)

...

```
rntri (); /* Retira o Relacionamento Triplo */
```

O exemplo mostrado se refere a retirada de relacionamento do tipo triplo.

4.2.3 - Leitura

4.2.3.1 - Como ler um atributo :

Considerando como exemplo o programa referente a Equipes, ou seja :

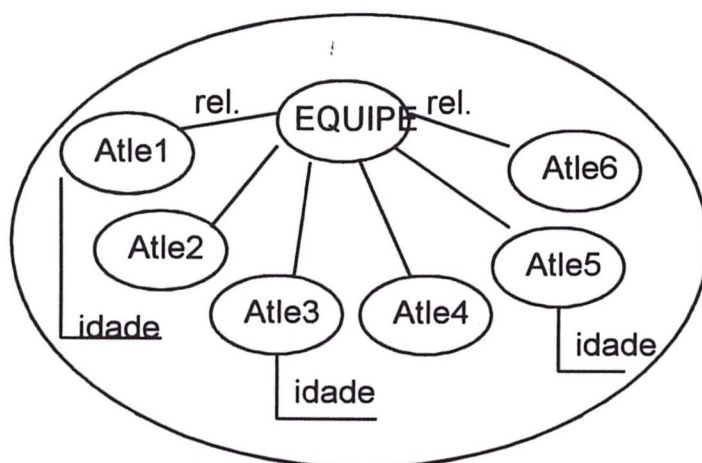


Figura 4.4 - Equipes

A colônia global tem que estar aberta para que seja feita a leitura do atributo IDADE com seu respectivo objeto ATLETA. Então devemos posicionar no objeto EQUIPE, para posteriormente relacioná-lo ao objeto ATLETA através do relacionamento desejado, e obter o código deste. Para a leitura do atributo de todos os atletas deve ser utilizado o comando que permite o avanço, permitindo percorrer todos os atletas daquela equipe. Finalmente depois de localizado e posicionado no objeto desejado, é possível a leitura do atributo IDADE.

4.2.3.2 - Como ler um relacionamento :

A leitura de um relacionamento é possível se e somente se, primeiramente precisamos posicioná-lo no objeto desejado, por exemplo, Equipe Banespa, obter o código deste e relacioná-lo com o atleta, ou seja, posicionar no relacionamento para posteriormente obtê-lo.

A sequência definida acima pode ser vista no programa Equipes.

4.2.4 - Avanço

4.2.4.1 - Como avançar para o objeto seguinte :

O posicionamento a um objeto se torna indispensável no avanço para o objeto seguinte, afinal é necessário que o objeto esteja corrente. O avanço pode ser utilizado por exemplo, para seleção de objetos como Equipes.

4.2.4.2 - Como avançar um atributo :

O avanço para o objeto seguinte depende do posicionamento neste. Isso é necessário, pois temos que torná-lo corrente. O avanço pode ser utilizado para selecionar atributos de vários atletas de uma determinada Equipe. Para isso é necessário avançar até que finalizem os atletas do Banespa por exemplo.

4.2.4.3 - Como avançar um relacionamento binário :

Ao avançar um relacionamento binário temos que posicionar neste. Podemos utilizá-lo para selecionar vários atletas de uma determinada equipe, que possui um relacionamento específico com a Equipe.

4.2.4.4 - Como avançar um relacionamento triplo :

Ao avançar um relacionamento triplo temos que posicionar neste. Podemos utilizá-lo para selecionar vários objetos de uma determinada equipe, que possui um relacionamento específico com a Equipe.

4.2.5 - Posicionamento

4.2.5.1 - Como posicionar em um objeto :

A utilização da rotina PTOBJ (posiciona em um objeto pelo seu nome e tipo) é mostrado na Listagem Individual de Equipes de Voleibol, para posicionar no objeto do tipo Equipe.

Definição dos parâmetros :

Para a rotina CODOBJ

cadeia nome = " grupo "

*t_obj = &grupo

Para a rotina PTOBJ

nome = nome

t_obj = grupo

Baseado no Programa referente a Equipes :

```
equi_ind ( ); /* Rotina para listagem individual de equipes */
```

```
...
```

definição de variáveis

```
codobj (nomeequi, &nomeequi);
```

```
printf (\"Equipe :);
```

```
getk (nom_equ);
```

```
erro = ptobj (nom_equ, grupo); /* Posiciona no objeto indicado */
```

4.2.5.2- Como posicionar em um relacionamento :

O posicionamento em um relacionamento pode ser de duas formas

- 1) Posicionamento no primeiro relacionamento do tipo especificado.
- 2) Posicionamento no relacionamento do tipo especificado.

sendo estes feitos através das rotinas TAREL e PDREL respectivamente.

Porém, para que estas sejam usadas é necessário a passagem do código referente ao tipo de relacionamento, o qual é obtido através das rotinas CODMOD1 (para relacionamento binário) e CODMOD2 (para relacionamento triplo).

Definição dos parâmetros :

Para a rotina CODMOD1

cadeia nome = composto_por
*t_mod =

Para a rotina TAREL

t_rel = composto_por

Para a rotina PDREL

t_rel = composto_por
codest = atleta

Baseado no exemplo de Equipes :

definicao de variáveis

...

printf ("Equipe : ");

getk (nome);

erro = ptobj (nome, grupo);

if (erro == 0) {

 bnoobj (&codaux);

tarel (composto_por); /* Posiciona no relacionamento indicado */

...

4.2.5.3- Como posicionar em uma colônia :

O posicionamento em uma colônia depende de três operações :

- Requisitar acesso a colônia;
- Abrir a colônia;

Essas operações são possíveis através das rotinas ANCOL e PNCOL respectivamente. A rotina ANCOL possui 3 parâmetros os quais informam :

- Tipo de Colônia;
- Pedido;

Este parâmetro define o tipo de acesso requisitado pelo usuário, dentre eles temos:

- geo_le
- geo_le_escreve
- geo_cria_objeto
- geo_apaga_objeto
- geo_cria_colonia
- geo_apaga_colonia
- geo_modifica_colonia
- geo_bloqueia_colonia
- geo_cria_relacionamento
- geo_apaga_relacionamento
- geo_modifica_relacionamento
- geo_cria_atributo
- geo_apaga_atributo
- geo_modifica_atributo

que permite fazer as funções indicadas a frente da palavra GEO.

- Concedido : Este parâmetro retorna se a colônia está ou não acessível.

A rotina PNCOL possui os parâmetros referentes a:

- nome da colônia
- tipo da colônia

Consideramos que após ser manipulada, a colônia deve ser fechada e posteriormente liberada, o que é feito pelas rotinas XNCOL e ZNCOL respectivamente.

Os parâmetros são nome e tipo da colônia para ZNCOL, e o tipo da colônia para XNCOL.

Exemplo :

```
definição de variáveis  
codcol ("federacao", tipfed);  
...  
ancol (tipfed, geo_le, acesso);  
pncol ("federacao", tipfed);  
...  
xncol (tipfed);  
zncol ("federacao", tipfed);
```

Consideramos que após a abertura da colônia foi feita a operação desejada pelo usuário, e após isso esta foi fechada e liberada para acesso.

4.2.6 - Busca

As rotinas referente a busca, definem a sequência em uma colônia.

4.2.6.1 - Como definir sequência de busca pelo tipo :

A rotina TDOBJ (inicia busca pelo tipo), faz a seleção dos objetos que tiverem o tipo indicado, e considera para a sequência que irá montar, apenas o nome principal.

Exemplo : Considerando a seguinte colônia com seus respectivos objetos e sinônimos.

Colônia Global

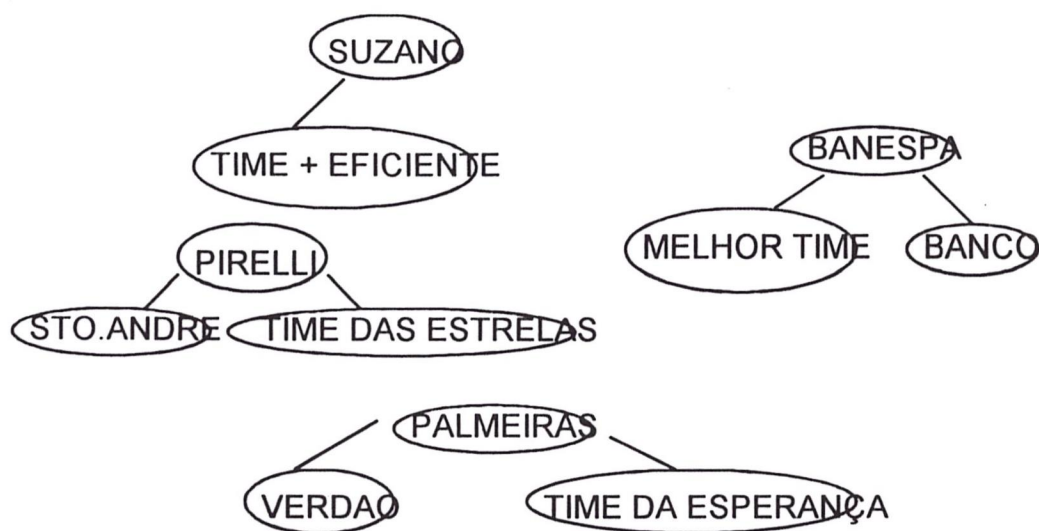


Figura 4.5 - Definição das Equipes na Colônia Global

Atletas :

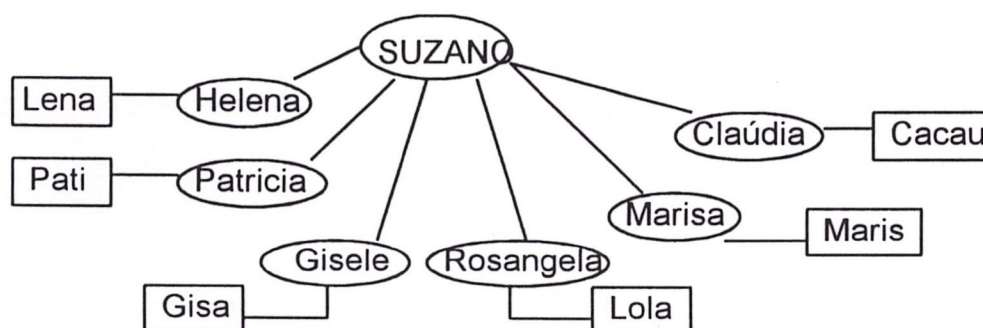


Figura 4.6 - Definição dos Atletas da Equipe Suzano

Suzano - contém as atletas :

Helena, Patricia, Gisele, Rosangela, Marisa, Claudia

com seus respectivos sinônimos :

Lena, Pati, Gisa, Lola, Maris, Cacau

Pirelli - contém as atletas :

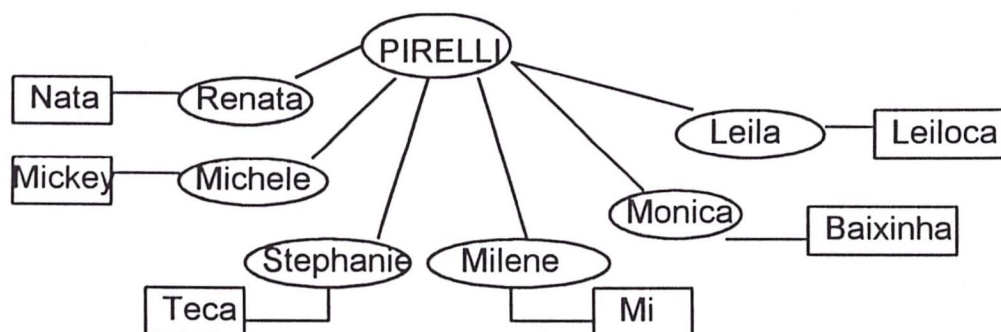


Figura 4.7 - Definição dos Atletas da Equipe Pirelli

Renata, Michele, Stephanie, Milene, Mônica, Leila

com seus respectivos sinônimos :

Nata, Mickey, Teca, Mi, Baixinha, Leiloca

Banespa - contém as atletas :

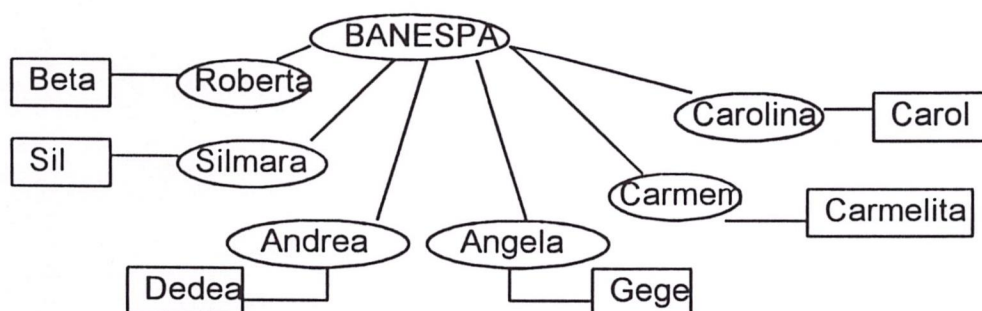


Figura 4.8 - Definição dos Atletas da Equipe Banespa

Roberta, Silmara, Andréa, Angela, Carmem, Carolina

com seus respectivos sinônimos :

Beta, Sil, Dedéa, Gegê, Carmelita, Carol

Palmeiras - contém as atletas :

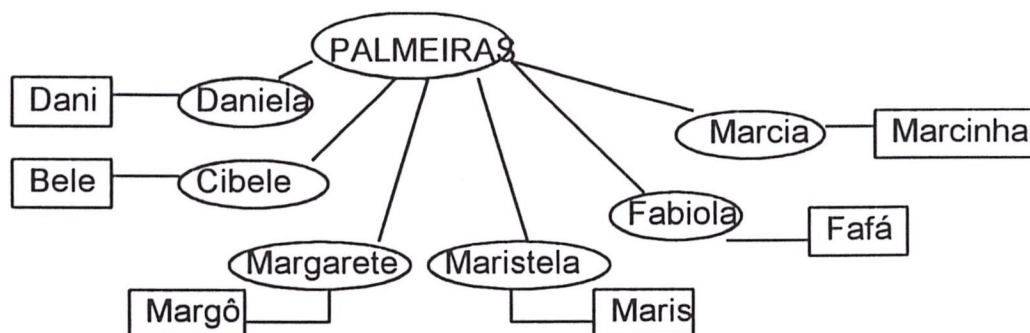


Figura 4.9 - Definição dos Atletas da Equipe Palmeiras

Daniela, Cibebe, Margarete, Maristela, Fabiola, Marcia

com seus respectivos sinônimos :

Dani, Bele, Margô, Tela, Fafá, Marcinha

Essa rotina fará o seguinte :

BANESPA	PALMEIRAS	PIRELLI	SUZAN
---------	-----------	---------	-------

Figura 4.10 - Nomes de Equipes

considera na busca apenas os nomes principais das Equipes.

4.2.6.2 - Como definir sequência de busca na colônia :

A rotina TEOBJ (inicia busca na colônia) considera para a sequência que irá montar, o nome principal de todas as Equipes e seus respectivos atletas, ou seja, a colônia.

BANESPA	PALMEIRAS	PIRELLI	SUZAN
---------	-----------	---------	-------

Figura 4.11 - Nomes Principais de Equipes

E todos os nomes principais dos atletas referentes a cada Equipe acima.

4.2.6.3 - Como definir sequência de busca pelos sinônimos :

A rotina TFOBJ inicia busca de nomes na colônia, ou seja, são considerados nomes principais e sinônimos de todas as Equipes e seus respectivos atletas.

PALMEIRAS	BANESPA	SUZANO	PIRELLI
TIME ESPERANCA	BANCO	GAROTOS ESPERTO	OSTO.ANDR

Figura 4.12 - Nomes Principais e Sinônimos de Equipes

E todos os nomes principais dos atletas :

Helena, Patricia, Gisele, Rosangela, Marisa, Claudia, Renata, Michele, Stephanie, Milene, Mônica, Leila, Roberta, Silmara, Andréa, Angela, Carmem, Carolina, Daniela, Cibebe, Margarete, Maristela, Fabíola, Marcia.

com seus respectivos sinônimos :

Lena, Pati, Gisa, Lola, Maris, Cacau, Nata, Mickey, Teca, Mi, Baixinha, Leiloca, Beta, Sil, Dedéa, Gegê, Carmelita, Carol, Dani, Bele, Margô, Tela, Fafá, Marcinha.

4.2.7 - Abertura

4.2.7.1 - Como abrir uma colônia :

A abertura de uma colônia é feita pela rotina PNCOL, que necessita da passagem do código do tipo da colônia. Logo é necessário a utilização da rotina CODCOL para obtê-lo. Porém inicialmente devemos requisitar acesso a esta, é feito através da rotina ANCOL, que possui os parâmetros :

- Tipo de Colônia;
- Pedido;

Este parâmetro define o tipo de acesso requisitado pelo usuário, dentre eles temos :

geo_le
geo_le_escreve
geo_cria_objeto
geo_apaga_objeto
geo_cria_colonia
geo_apaga_colonia
geo_modifica_colonia
geo_bloqueia_colonia
geo_cria_relacionamento
geo_apaga_relacionamento
geo_modifica_relacionamento
geo_cria_atributo
geo_apaga_atributo
geo_modifica_atributo

que permite executar as funções indicadas a frente da palavra GEO.

- Concedido : Este parâmetro retorna se a colônia está ou não acessível.

A rotina PNCOL possui os parâmetros referentes a :

- nome da colônia;
- tipo da colônia;

Consideramos que após ser manipulada, a colônia deve ser fechada e posteriormente liberada, o que é feito pelas rotinas XNCOL e ZNCOL respectivamente.

Os parâmetros são nome e tipo da colônia para ZNCOL, e o tipo da colônia para XNCOL.

Exemplo :

```
definição de variáveis
codcol ("federacao", tipfed);
...
ancol (tipfed, geo_le, acesso);
pncol ("federacao", tipfed);
...
xncol (tipfed);
zncol ("federacao", tipfed);
```

Consideramos que após a abertura da colônia foi feita a operação desejada pelo usuário, e após isso foi fechada e liberada para acesso.

4.2.8 - Listagem

4.2.8.1 - Como listar todos os objetos de uma colônia :

Considerando que exista uma colônia global com os objetos Equipe e Atleta.

```
...
printf (" Equipe : ");
getk (nome); /* Considerando que os parâmetros serão informados */
erro = ptobj(nome, grupo);
if (erro == 0)
```

```
    bnobj (&codaux);
while (cod_equi == codaux) { /* De acordo com este teste é feita a seleção dos
                               objetos */

    printk (" Nome: ");          /* Imprime os atributos, capturados pela
    pnobj (cod_atl);              rotina btobj, e posicionado através da
    btobj (nom_atl, atleta);      rotina ptobj */
    printk ("%a\n", nom_atl);

    printk (" Função : ");
    btobj (fun_atl, atleta);
    printk ("%a \n", fun_atl);

    printk (" Posição : ");
    btobj (pos_atl, atleta);
    printk ("%a \n", pos_atl);

    printk (" Idade : ");
    btobj (ida_atl, atleta);
    printk ("%a \n", ida_atl);

    printk (" Altura : ");
    btobj (alt_atl, atleta);
    printk ("%a \n", alt_atl);

    vnobj ( );
}
```

4.2.8.2 - Como listar todos os objetos de um determinado tipo, de uma colônia :

Considerando o exemplo acima, listar somente as Equipes.

```
...
printf ("Equipe :");
```

```

getk (nome);
erro = ptobj (nome, grupo);
if (erro == 0)
    bnobj (&codaux);
while (cod_equi == cod_aux) {

    printk ("Nome : ");          /* Impressão do nome e cognome ref. ao
    pnobj (cod_equi);            objeto Equipe */
    printk ("%a\n", nome);

    printk ("Cognome : ");
    btsin (cognome, grupo);
    printk ("%a\n", cognome);
}

```

```
printk ("Nome :");  
pnobj (cod_equi);  
printk ("%a\n", nome);
```

4.2.8.4 - Como listar todos os valores de todos atributos de um objeto :

Exemplo : Listagem do nome e cognome (todos os atributos) de todas as equipes.

```
...  
printf ("Equipe :");  
getk (nome);  
erro = ptobj (nome, grupo);  
if (erro == 0)  
    bnobj (&codaux);  
while (cod_equi == cod_aux) {  
  
    printk ("Nome : ");  
    pnobj (cod_equi);  
    printk ("%a\n", nome);    /* Impressão de todos os atributos referente  
    printk ("Cognome :");      ao objeto Equipe */  
    btsin (cognome, apelido);  
    printk ("%a\n", cognome);
```

4.2.8.5 - Como listar todos os valores de um tipo de atributo de um objeto :

Caso queira a listagem da idade de todos os atletas, utilizar a rotina de acordo com o seguinte exemplo.

```
...  
printf ("Equipe : ");  
getk (nome);  
erro = ptobj (nome, atleta);  
if (erro == 0)
```



```

bnoobj (&codaux);
tarel (composto_por);
erro = bdrel(&cod_equi, &rel_pad, &cod_atl);
while ((erro == 0) && (rel_pad == composto_por)) {

    printk ("Idade : ");
    tapro (idade);           /* Impressão somente do atributo
    bdpro (ida_atl,idade);    idade, referente ao objeto Atleta */
    printk ("%a\n", ida_atl);
}

```

4.2.8.6 - Como listar todos os valores de todos atributos de todos objetos :

Considere que listemos :

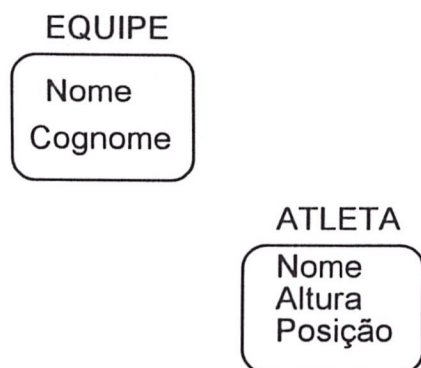


Figura 4.13 - Listagem de Equipe e Atleta

```

...
printf ("Equipe : ");
getk (nome);
erro = projb (nome, atleta);
if (erro == 0)
    bnoobj (&codaux);
    tarel (composto_por);
    erro = bdrel(&cod_equi, &rel_pad, &cod_atl);
    while ((erro == 0) && (rel_pad == composto_por)) {

```

```

printk ("Altura :");
tapro (altura);
bdpro (alt_atl, altura);
printk ("%a\n", alt_atl);

```

```

printk ("Posicao :");
tapro (posicao);
bdpro (pos_atl, posicao);
printk ("%a\n", pos_atl);

```

/* Impressão de todos os
atributos referente a
todos os objetos */

```

printk ("Idade:");
tapro (idade);
bdpro (ida_atl, idade);
printk ("%a\n", ida_atl);

```

4.2.8.7 - Como listar todos os valores de um tipo de atributo de todos objetos :

Exemplo : Considerando que dois ou mais objetos tenham um atributo em comum.
Listagem de todos os nomes de Equipes e Atletas.

```

...
printk ("Equipe :");
getk (nome);
erro = ptobj(nome, grupo);
if (erro == 0)
    bnobj (&codaux);
    printk ("%a\n", nome);
    tarel (composto_por);
    erro = bdrel (&cod_equi, &rel_pad, &cod_atl);
    while ((erro == 0) && (rel_pad == composto_por)) {

```

```

printk ("Nome : ");
pnobj (cod_atl);

```

/* Impressão do atributo Nome ref.aos

```
btobj (nom_atl, nome);           objetos Equipe e Atleta */
printk ("%a\n", nom_atl);
```

4.2.8.8 - Como listar todos os relacionamentos binários de um objeto :

Exemplo : Listagem de :

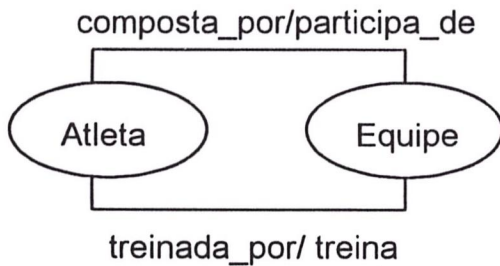


Figura 4.14 - Listagem de Relacionamento Binários

todos os relacionamentos binários acima mostrados.

```
...
printk ("Equipe :");
getk (nome);
erro = ptobj (nome, grupo);
if (erro == 0)
    bnobj (&codaux);
    erro = bdrel (&cod_equi. &rel_pad, &cod_atl);

if (erro == 0) {                               /* Seleção dos relacionamentos
    printk ("Relacionamento :);               binários a serem impressos */
    printk ("%a\n", rel_pad);
    vnrel ();
```