UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA DE CAMPINAS

DEPARTAMENTO DE ENGENHARIA ELÉTRICA

PROJETO DE UM BANCO DE DADOS TOPOGRÁFICOS EM DATAS

LUIZ ARY MESSINA

Orientador: Prof. MÁRIO JINO

39/80

Tese de Mestrado apresentada à Faculdade de Engenharia da Universidade Estadual de Campinas

DEZEMBRO - 1980

" A LIBERDADE CONSISTE EM SER RAIZ DE AÇÕES" ~ GÉRARD LEBRUN

AGRADECIMENTOS

Aos meus pais CARLOS e THEREZINHA

À minha mulher FLÁVIA

Ao meu orientador JINO

Aos meus colegas da FEC

pelo carinho e participação
sem os quais este trabalho
dificilmente seria realizado.

SUMÁRIO

Nesta tese apresenta-se a descrição do banco de dados DATAS, um exemplo de aplicação para manipulação de dados topográficos e, uma análise do DATAS.

A estruturação dos dados topográficos segundo o modelo de dados do DATAS é discutida e sugere-se uma estrutura adequada aos requisitos de manipulação e manutenção da informação topográfica da aplicação escolhida.

A linguagem de organização e manipulação de dados do DATAS é o assembler associativo. O DATAS foi projetado para o tratamento de informações interativamente, passo a passo, ou por programas de aplicação em batch. Um conjunto de programas em batch foi desenvolvido para simplificar as operações com os dados.

Inclui-se ainda um apêndice onde se mostra que uma linguagem de alto nível pode ser implementada no DATAS.

ÍNDICE

CAPÍTULO 1

INTRODUÇÃO

A área de banco de dados vem despertando um interesse crescente tanto por parte das empresas como dos especialistas em computação. Para as empresas, representa uma forma eficaz de gerir a informação, que hoje é reconhecida como um dos recursos empresariais mais importantes. Para os especialistas em computação, surge como um campo de aplicação útil de diferentes disciplinas (5)

Mas o que é um banco de dados? Mesmo esta questão básica tem uma variedade de respostas na literatura. Visto idealisticamente, é como uma mesa de frios num caríssimo hotel nas montanhas: existem todos os tipos de comida e voce pode dirigir-se à mesa (se puder pagar), pegar apenas o que quiser e não se preocupar com o resto. Se voce observar os pratos dos diferentes hóspedes, descobrirá que cada um compôs seu prato próprio a partir da mesma mesa (6).

A intenção deste trabalho é esclarecer a utilização de um banco de dados (DATAS) fornecendo noções básicas de bancos de dados em geral. As características do banco de dados DATAS são analisadas e um estudo sobre uma linguagem de alto nível de organização e manipulação de dados é apresentada.

Os capítulos seguintes estão dispostos na seguinte estrutura:

No CAP. 2 apresenta-se a teoria de banco de dados em geral, dando-se ênfase a uma arquitetura que permita uma maior independência dos dados à alterações nas estruturas de armazenagem e estratégias de acesso. Os modelos de dados, relacional, hierárquico e de rede são apresentados e uma linguagem relacional de alto nível baseada na álgebra relacional é discutida.

No CAP. 3 apresenta-se uma descrição resumida do banco de dados DATAS, abordando-se detalhadamente sua estrutura de dados. A linguagem de organização e manipulação de dados na estrutura é o assembler associativo.

No CAP. 4 apresenta-se um exemplo de aplicação, onde o banco de dados é utilizado para criar um arquivo de estruturas associativas topográficas. Um conjunto de programas, que reflete as operações necessárias a serem efetuadas na estrutura de dados, é desenvolvido facilitando a utilização do sistema de informações topográficas. O exemplo de uma situação real é mostrado.

No CAP. 5, com base no exemplo desenvolvido no capítulo anterior, analisam-se algumas características do banco de dados DATAS e propõem-se implementações dando prosseguimento ao estudo iniciado neste trabalho.

O Apêndice mostra a possibilidade de criação de uma linguagem de alto nível de organização e manipulação de dados, baseada na equivalência semântica de operadores.

CAPÍTULO 2

INTRODUÇÃO A SISTEMAS DE BANCO DE DADOS

2.1. Conceitos Básicos

2.1.1. O que é um banco de dados?

Existem várias definições de banco de dados:

- Uma coleção de dados operacionais armazenados utilizados pelo sistema de aplicação de alguma empresa (1).
- Uma coleção de dados interrelacionados armazenados juntos, sem redundância prejudicial ou desnecessária para servir uma ou mais aplicações de uma maneira otimizada (2).

Empresa é um termo genérico conveniente para qualquer operação comercial, científica, técnica ou outras de larga escala. Qualquer empresa deve necessariamente manter uma grande quantidade de dados sobre suas operações. Estes são os dados operacionais.

Podemos ilustrar a conceituação através de uma empresa que mantém dados sobre fornecedores, partes e projetos. Existem dados de fornecedores, de partes e ainda informação de que os fornecedores fornecem algumas partes para alguns projetos. Fornecedores, partes e projetos são o que podemos chamar comumente de entidades, e as associações entre as entidades são os relacionamentos. Muitos bancos de dados consideram entidades e relacionamentos como dois tipos distintos de objetos. Para nossa exposição, entretanto, consideramos uma associação entre entidades também como uma entidade.

2.1.2. Porque utilizar-se banco de dados?

Porque deve uma empresa escolher a armazenagem de seus dados opera-

cionais num banco de dados? Uma resposta geral é que o banco de dados fornece um controle centralizado dos dados operacionais.

Consideremos as vantagens de obter um controle centralizado dos dados:

- O grau de redundância de dados armazenados pode ser reduzido.

Com controle central, o administrador do banco de dados pode identificar o fato de que aplicações usam essencialmente os mesmos dados e integrá-los armazenando-os apenas uma vez e tornando-os disponíveis às várias aplicações.

- Problemas de inconsistência nos dados armazenados podem ser evitados (até certo ponto).

Se não há redundância, não ocorrem inconsistências. Por inconsistência entende-se a existência de informações discordantes em pelo menos dois lugares.

- Pode-se forçar o cumprimento de normas e padrões.

Representação padronizada dos dados simplifica problemas de manutenção e intercâmbio de dados entre instalações.

- Restrições de segurança podem ser aplicadas.

Tendo completa jurisdição sobre os dados operacionais, o administrador de banco de dados pode assegurar que o acesso aos dados é feito através de canais próprios e então definir verificações de autorização a serem aplicados quando o acesso à dados sensíveis é requerido. Diferentes procedimentos podem ser estabelecidos para cada tipo de acesso (recuperação, atualização, remoção, etc.). Deve ser ressaltado que sem tais procedimentos, a segurança dos dados num banco de dados corre mais risco que em sistemas tradicionais de arquivos.

- Requisitos conflitantes podem ser balanceados

Conhecendo os requisitos gerais da empresa, o administrador do banco de dados pode optar por estruturas de armazenagem que possibilitem um acesso rápido para as aplicações mais importantes ao custo de baixo desempenho de outras aplicações.

- Integridade dos dados pode ser mantida

O problema de integridade é o de assegurar que o banco de dados contém apenas dados corretos. Controle centralizado do banco de dados pode evitar a existência de dados incorretos, permitindo que o administrador do banco de dados defina procedimentos de validação a serem aplicados sempre que uma operação de armazenagem é requerida.

2.1.3. Independência de dados

A maioria das aplicações hoje são dependentes dos dados. Ou seja, a maneira pela qual os dados são organizados na memória secundária e o modo de acesso são ambos ditados pelos requisitos da aplicação; a organização dos dados e as técnicas de acesso devem ser consideradas na lógica da aplicação.

Num sistema de banco de dados, cujo esquema é mostrado na Fig. 1, seria indesejável permitir que as aplicações fossem dependentes dos dados.

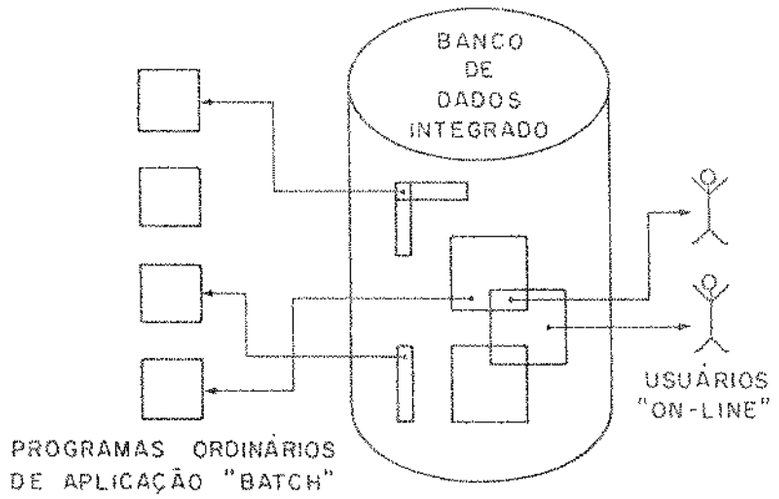


Fig. 1 - Visão simplificada de um sistema de banco de dados

O banco de dados em si é uma coleção de dados armazenados em discos, fitas ou outros meios de armazenagem secundária. Um conjunto de programas "batch" de aplicação operam sobre os dados da maneira usual (remoção, inserção, atualização, recuperação). Adicionalmente ou alternativamente, pode existir um

conjunto de usuários "on-line" que interagem com o banco de dados através de terminais interativos. Por fim, o banco de dados é integrado possibilitando o englobamento de vários conjuntos de dados e suas utilidades para a manipulação por vários grupos de usuários ("batch" ou "on-line").

A seguir, duas razões principais para almejar a independência dos dados:

a) Diferentes aplicações podem necessitar diferentes visões de um mesmo dado (e.g., 1 pé ou 30,48cm).
Isto gera a necessidade de interfaces para que o dado armazenado seja um só, e não dois representando a mesma informação.

b) O administrador do banco de dados, pessoa responsável pelos dados operacionais, deve ter a liberdade de modificar estruturas de armazenagens ou estratégias de acesso (ou ambas) em resposta à mudanças de requisitos, sem a necessidade de modificar as aplicações existentes (e.g., novos padrões são incorporados, prioridades de aplicações podem mudar, novos tipos de dispositivos de armazenagem podem ser instalados, etc.).

Todas essas alterações normalmente implicariam em um grande volume de modificações nos programas de aplicação, ocupando pessoal que poderia estar dedicado a criação de novas aplicações.

No próximo item, sugere-se uma arquitetura avançada baseada na independência dos dados vistos pelo usuário em relação aos dados físicos armazenados (compactados, codificados, etc.).

2.1.4. Uma arquitetura para um sistema de banco de dados

A arquitetura da Fig. 2 é uma representação bem próxima de alguns sistemas propostos e está de acordo com a arquitetura proposta pelo ANSI/SPARC Study Group on Data Base Management Systems (1).

Existem três níveis na arquitetura: o externo ou nível de comunicação com o usuário, o nível conceitual de definição de modelos de dados e o nível interno de armazenagem física dos dados.

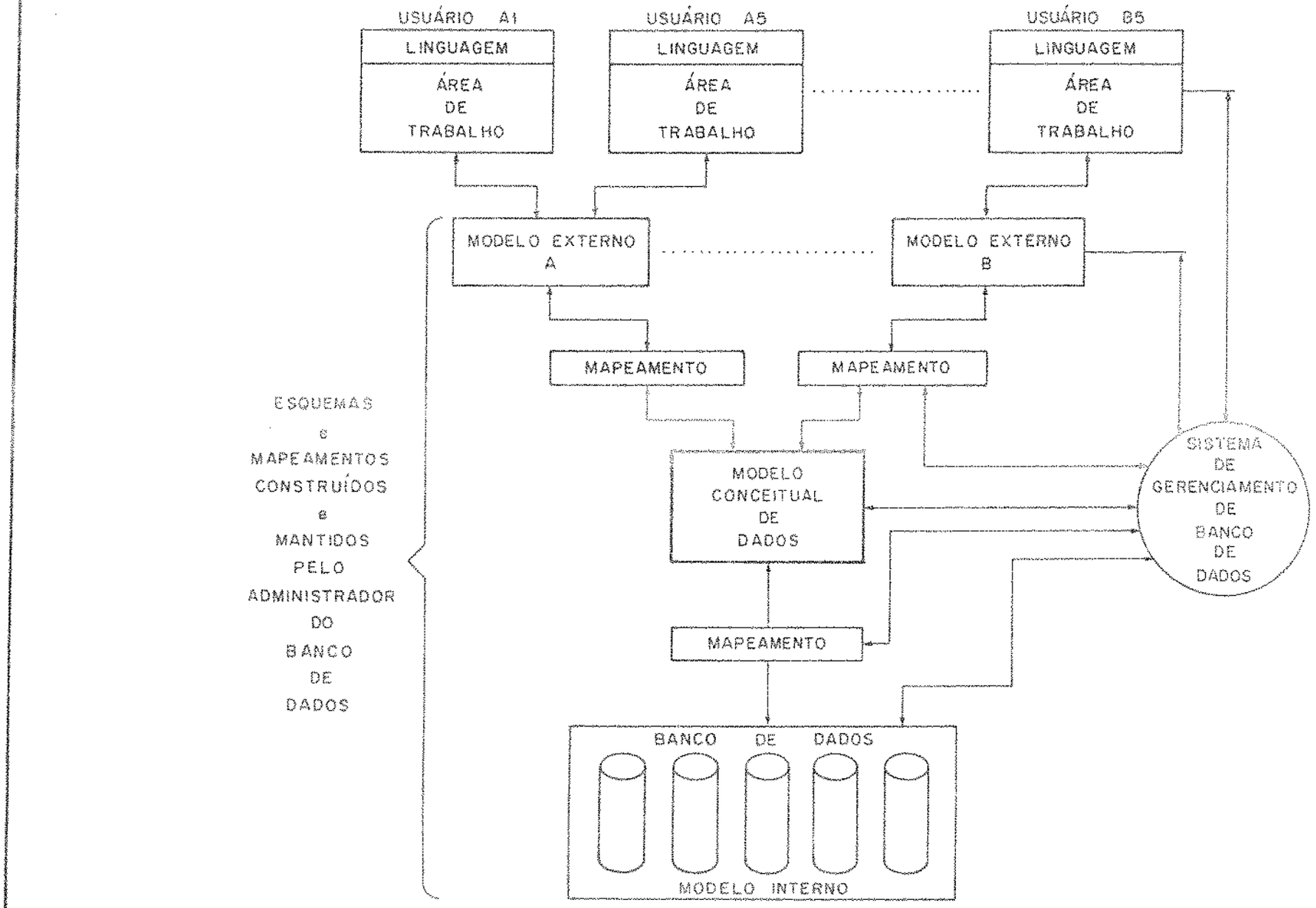


Fig. 2 - UMA ARQUITETURA PARA UM SISTEMA DE BANCO DE DADOS

O usuário "on-line" ou o programador de aplicação possui uma área de trabalho e através de uma linguagem de manipulação de dados determinada pelo modelo externo seleciona operações a serem efetuadas nos dados. O modelo externo é uma parte do modelo de dados, contendo características específicas à uma classe de usuários. O modelo conceitual é uma visão do conteúdo total do banco de dados, representado na forma dos vários modelos de dados conhecidos (relacional, hierárquico, rede, entidade-relação). O modelo interno é uma representação do dado armazenado fisicamente.

Uma alteração na estrutura armazenada ou na estratégia de acesso ao meio de armazenagem implica em modificações efetuadas no mapeamento interno/conceitual, mantendo o modelo conceitual e portanto os programas de aplicação independentes das modificações efetuadas a nível interno.

Abordaremos ainda três pontos: o sistema de gerenciamento de banco de dados, o administrador de banco de dados e a interface do usuário. O sistema de gerenciamento de banco de dados (SGBD) é o software encarregado de todo e qualquer acesso ao banco de dados. Conceitualmente ocorre o seguinte:

a) Um usuário formula uma requisição de acesso, usando algum tipo de linguagem natural (*);

b) O SGBD recebe a requisição e interpreta-a;

c) O SGBD inspeciona o esquema externo, o mapeamento externo/conceitual, o esquema conceitual, o mapeamento conceitual/interno e a definição da estrutura de armazenagem; e

d) O SGBD executa as operações necessárias no banco de dados armazenado.

O SGBD é também responsável pela aplicação das verificações de autorização e procedimentos de validação mencionados anteriormente.

O Administrador de banco de dados já mencionado é a pessoa ou grupo de pessoas encarregada do controle geral do sistema de banco de dados. Suas

---

(*) Analisaremos mais tarde um tipo de linguagem considerada quase natural (a nível conversacional).

obrigações incluem as seguintes:

- Decidir o conteúdo de informação do banco de dados
- Decidir a estrutura de armazenagem e estratégia de acesso
- Comunicação com usuários
- Definir verificações de autorização e procedimentos de validação
- Definir uma estratégia para "back-up" e recuperação de arquivos
- Monitorar desempenho e responder à mudanças em requisitos.

Finalmente a interface do usuário, que pode ser definida como uma linha divisória no sistema abaixo da qual tudo se passa de maneira invisível ao usuário. Esta interface é representada na arquitetura pelo modelo externo, o qual pode muitas vezes ser apenas parte do modelo conceitual.

2.2. Modelos e Sublinguagens de Dados

Na arquitetura apresentada, observa-se nitidamente o papel central representado pelo modelo conceitual de dados. A variedade de estruturas de dados mantidas no modelo conceitual afeta criticamente todos os componentes do sistema; em particular, dita o projeto de sublinguagens de dados desde que cada operação da sublinguagem é definida baseada em seus efeitos sobre as estruturas armasenadas. Cabe então uma pergunta: Que formas devem o modelo de dados e respectivas sublinguagens tomar?

Para responder esta questão, apresentaremos as abordagens mais conhecidas, deixando de lado o modelo entidade-relacionamento (4) por se tratar de um modelo não muito conhecido e do qual o modelo relacional e o DBTG podem ser derivados. O modelo entidade-relacionamento foi criado para permitir ao usuário a visualização e manipulação direta de suas informações dentro dos conceitos próprios e comuns de entidades e relacionamentos. A sublinguagem para ele criada é a GLEAN, baseada no emprego de operadores de álgebra relacional.

As três abordagens mais conhecidas são: RELACIONAL, HIERÁRQUICA, REDE.

2.2.1. A abordagem relacional

Todas as abordagens serão baseadas numa amostra de um banco de dados referente a fornecedores, partes e remessas. A Fig. 3 mostra os dados em forma relacional; isto é, ela representa um modelo relacional dos dados.

FORNECEDORES (S)

Sn	S NOME	STATUS	CIDADE
S1	JOÃO	10	RIO
S2	JOSÉ	10	S. PAULO
S3	PEDRO	30	BRASÍLIA

REMESSA (SP)

Sn	Pn	QTY
S1	P1	300
S1	P2	100
S1	P3	200
S2	P1	200
S2	P2	400
S3	P2	500

PARTES (P)

Pn	P NOME	COR	PESO	CIDADE
P1	PINO	AZUL	20	RIO
P2	ROSCA	PRETA	33	BELO
P3	PREGO	CINZA	18	S. PAULO
P4	PORCA	LARANJA	40	BRASÍLIA

Fig. 3 - Modelo Relacional de dados

Pode-se notar que, cada uma dessas tabelas (ou relação, como definiremos) se assemelha à um arquivo sequencial tradicional, com suas linhas correspondentes a registros e colunas à campos de registros. No entanto, existem certas diferenças significativas, tais como:

a - um "arquivo" (tabela) contém apenas um tipo de registro

b - cada tipo de registro tem um número fixo de tipos de campos

c - cada ocorrência de registro tem um identificador único

d - dentro de um "arquivo", ocorrências de um registro contêm uma ordem desconhecida, ou são ordenados por valores contidos nas ocorrências.

Cada uma dessas tabelas é realmente um caso especial do construto conhecido em matemática como uma relação, definida como segue: Dada uma coleção de conjuntos D_1, D_2, ..., D_n (não necessariamente distintos), R é uma relação nestes n conjuntos se R é um conjunto de enuplas (ou linhas) ordenadas $<d_1, d_2, \ldots, d_n>$ tais que $d_1 \in D_1$, $d_2 \in D_2$, ... $d_n \in D_n$. Os conjuntos D_1, D_2, ..., D_n são os domínios da relação R.

No caso da relação Fornecedor, os domínios da relação são Sn, SNOME, STATUS e CIDADE, e as tuplas são <S1, João, 20, Rio>, <S2, José, 10. S.Paulo>, e <S3, Pedro, 30, Brasília>, tais que Sn é um conjunto que contém S1, S2 e S3, SNOME contém João e Pedro etc. A relação Partes forma suas tuplas ordenadas dos domínios Pn, PNOME, COR, PESO, CIDADE.

Observa-se que em ambas relações Fornecedores e Partes existe um domínio comum CIDADE. Por exemplo, o fornecedor S3 e a parte P4 estão localizados em BRASÍLIA. Isto faz com que tuplas sejam associadas através de valores em colunas pertencentes ao mesmo domínio ($d_i \in D_i$).

Esta é uma característica da abordagem relacional: toda informação no banco de dados seja ela conceitualmente uma entidade ou uma associação, é representada da mesma maneira.

Pela própria disposição tabular, notamos a necessidade de nos servir de estruturas que facilitem o entendimento da semântica dos dados. Porém, simplicidade na representação dos dados não é tudo. Do ponto de vista do usuário, a sublinguagem de dados, i.e., o conjunto de operadores fornecido para manipular dados na forma relacional, é pelo menos tão importante. E desde que a informação é representada de uma só maneira, precisamos apenas um operador para cada tipo de função (inserção, remoção, etc.) que desejamos executar. Já em estruturas mais complexas, onde a representação da informação é feita de várias maneiras, vários conjuntos de operadores são necessários para operar sobre entidades conceitualmente distintas.

Como veremos no próximo capítulo e nos seguintes, o banco de dados que vamos utilizar possui uma estrutura de dados baseada no conceito matemático de conjuntos e em relações binárias. O Programa Supervisor, que seria um pequeno sistema de gerência de banco de dados permite manipulação interativa de dados em forma de diálogo e em "batch" usando uma linguagem tipo assembler de dados ou chamando as subrotinas diretamente por programas de aplicação que aceitem chamadas a subrotinas em Fortran IV. A linguagem assembler de manipulação de dados constituída de um conjunto de instruções, opera sobre conjuntos, relações e entidades simples (sobre as quais armazenam-se dados que têm significado

apenas para o usuário).

Consideremos agora uma linguagem de alto nível de manipulação de dados estruturados relacionalmente. Uma tal linguagem é uma facilidade muito útil em qualquer SGBD pois permite a sua utilização por usuários que entendam muito pouco ou quase nada de computadores e programação.

2.2.1.1. Uma sublinguagem de dados baseada na álgebra relacional

Álgebra relacional é uma coleção de operações de alto nível sobre relações. Neste item descrevemos um conjunto completo de operadores e indicamos uma possível sintaxe para a linguagem constituída desses operadores.

A álgebra completa compreende dois grupos de operações: operações tradicionais de conjunto (UNIÃO, DIFERENÇA, INTERSEÇÃO, e uma forma de PRODUTO CARTESIANO) e operações relacionais especiais (SELEÇÃO, PROJEÇÃO, JUNÇÃO e DIVISÃO). Todos os operadores são usados na recuperação de informação. Operações de armazenagens envolvem apenas união e diferença.

Todos os exemplos são baseados na Fig. 3. Assumiremos, para simplificar a definição das operações, que a ordem dos atributos numa relação é significativa.

OPERAÇÕES TRADICIONAIS SOBRE CONJUNTOS

Para efetuar as operações de união, interseção e diferença, as duas relações em questão devem ser compatíveis para união, i.e. as relações devem ter o mesmo grau (nº de atributos ou de colunas) e o i-ésimo atributo de uma relação deve pertencer ao mesmo domínio que o i-ésimo atributo da outra.

UNIÃO

A união de duas relações (compatíveis para união) A e B, A UNION B, é o conjunto de todas as tuplas t pertencentes a A ou a B (ou ambas).

Exemplo: Seja A o conjunto de tuplas de fornecedores no Rio e seja B o conjunto de tuplas de fornecedores que fornecem P1. Então A UNION B é o conjunto de tuplas de fornecedores localizados no Rio ou que fornecem P1, ou ambos.

INTERSEÇÃO

A INTERSECT B é o conjunto de todas as tuplas t pertencendo a ambas as relações A e B.

DIFERENÇA

A MINUS B é o conjunto de todas as tuplas t que pertencem à A e não à B.

PRODUTO CARTESIANO EXTENDIDO

O Produto Cartesiano Extendido de duas relações A e B, A TIMES B, é o conjunto de todas as tuplas t tais que t é a concatenação de uma tupla a pertencendo a A e uma tupla b pertencendo a B. A concatenação de uma tupla $a = <a_1, \ldots, a_m>$ e uma tupla $b = <b_{m+1}, \ldots, b_{m+n}>$, nesta ordem, é a tupla $t = <a_1, \ldots, a_m, b_{m+1}, \ldots, b_{m+n}>$.

OPERAÇÕES RELACIONAIS ESPECIAIS

SELEÇÃO

SELECT é um operador para construir um subconjunto horizontal de uma relação i.e., o subconjunto de tuplas que satisfazem uma certa condição (um predicado).

Fig. 4 mostra alguns exemplos

SELECT S WHERE CIDADE = "RIO"

Sn	SNOME	STATUS	CIDADE
S1	JOÃO	10	RIO

SELECT SP WHERE Sn = "S1"
AND Pn = "P1"

Sn	Pn	QTY
S1	P1	300

Fig. 4 - Exemplos de seleção

PROJEÇÃO

PROJECTION é um operador para construção de um subconjunto vertical de uma relação - i.e., um subconjunto obtido pela seleção de atributos especificados e eliminação de outros.

Na Fig. 5 mostra-se um exemplo.

PROJECT P OVER Pn

Pn
P1
P2
P3
P4

Fig. 5 - Exemplo de projeção

JUNÇÃO

O operador JOIN, do qual se apresenta um exemplo na Fig. 6 é, de fato, um equi-join, ou seja uma junção baseada na igualdade de valores no domínio comum. Podemos definir um greater-than-join como segue. O greater-than-join da relação A sobre o atributo X com a relação B sobre o atributo Y - JOIN A AND B WHERE X > Y - é o conjunto de todas as tuplas t tal que t é uma concatenação de uma tupla a pertencente a A com uma tupla b pertencente a B, onde X > Y (sendo X e Y valores de atributo comum às relações. Similarmente podemos definir joins para cada um dos operadores comparativos $\neq$, $\geqslant$, $<$, $\leqslant$ (em todos os casos X e Y devem pertencer ao mesmo domínio).

JOINS S AND SP OVER Sn WHERE Sn = "S2"

Sn	SNOME	STATUS	CIDADE	Sn	Pn	QTY
S2	JOSÉ	10	S.PAULO	S2	P1	200
S2	JOSÉ	10	S. PAULO	S2	P2	400

Fig. 6 - Exemplo de junção (equi-join)

DIVISÃO

A divisão é definida, em sua forma mais simples, como uma operação entre uma relação binária (o dividendo) e uma relação unária (o divisor) que produz uma relação unária (o quociente) como resultado. Seja A uma relação binária com atributos X e Y e seja o divisor B uma relação unária com atributo Z, e sejam Y e Z definidos pelo mesmo domínio subjacente. Assim a operação de dividir - DIVIDE A BY B OVER Y AND Z - produz um quociente definido no mesmo domínio de X: um valor x aparecerá no quociente se e somente se o par <x, y> aparece em A para todo valor de y que aparece em B.

No exemplo que se segue utilizaremos uma relação binária como dividendo e uma relação unária como divisor.

19) PROJECT SP OVER Sn, Pn
GIVING DEND

DEND

Sn	Pn
S1	P1
S1	P2
S1	P3
S2	P1
S2	P2
S3	P2

29) PROJECT (SELECT P WHERE PNOME = "ROSCA" OR
PNOME = "PREGO") OVER Pn GIVING DOR

DOR

Pn
P2
P3

39) DIVIDE DEND BY DOR OVER Pn GIVING RESULT

RESULT

Sn
S1

Para inserir, remover e/ou atualizar tuplas de uma relação, fazemos uso dos operadores de união e diferença de conjuntos.

Por exemplo para inserir uma nova tupla à tabela S, escrevemos:

S UNION {(S4, Carlos, 20, Paris)} GIVING S

Para deletar a tupla referente a parte P4 da tabela P, escrevemos:

P MINUS {(P4, Porca, Laranja, 40, Brasília)} GIVING P

Para as operações de atualização, fazemos uso de uma seqüência apropriada de operações de união e diferença.

Por exemplo: mudar o peso de P2 para 30

P MINUS {(P2, ?, ?, ?, ?)} GIVING P

P UNION {(P2, Rosca, Preta, 30, Belo)} GIVING P

Para finalizar listamos algumas características gerais de uma linguagem relacional algébrica:

. Simplicidade

A álgebra relacional é razoavelmente simples e potente.

. Completude

A álgebra é relacionalmente completa. Um corolário deste fato é que para mostrar que qualquer linguagem L é relacionalmente completa, é suficiente mostrar que L inclui análogos de cada uma das operações algébricas.

. Não proceduralidade

A álgebra é procedural, i.e., sua utilização envolve o encadeamento de operações como se faz num programa em uma linguagem comum de programação. Para o SGBD esta proceduralidade pode ser uma vantagem, na medida em que pelo menos a implementação pode ser razoavelmente direta. O SGBD pode simplesmente

executar todas as junções, projeções e outras operações tal como são expressados pelo usuário. Por outro lado, isto poderia conduzir a processamento pouco eficiente, obrigando o usuário a escolher o formato mais eficiente para a consulta (por exemplo, analisar se um SELECT aumentaria mais a eficiência, de uma consulta se feito antes ou depois de outra operação), o que é uma tarefa claramente indesejável. Não obstante, trabalhos tem sido desenvolvidos no sentido de otimizar a implementação de expressões algébricas.

. Facilidade de extensão

A álgebra não se presta facilmente à inclusão de operadores tais como COUNT e TOTAL.

2.2.2. A abordagem hierárquica

A figura abaixo mostra um possível modelo hierárquico para o banco de dados de fornecedores e partes. Neste modelo o dado é representado por uma única estrutura de árvore com partes superiores a fornecedores.

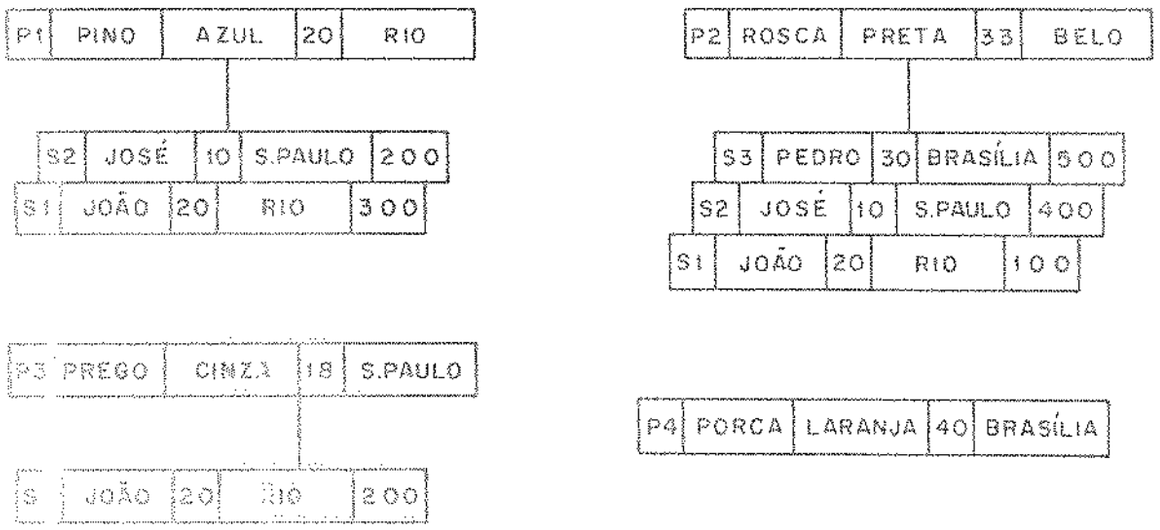


Fig. 7 - Amostra de dados em forma hierárquica

Na abordagem relacional, consideramos três arquivos dispostos em forma de tabela, cada arquivo com um tipo de registro que são as tuplas. Na abordagem hierárquica para o sistema de informações representado pelo conjunto de informações sobre fornecedores e partes, temos apenas um arquivo que contém dois tipos de registros (partes e fornecedores) e ainda uma ligação entre partes e fornecedores representando as remessas associadas. Nitidamente, estamos defronte a um problema de várias entidades representando a informação e portanto necessitaremos de vários operadores para a construção da sublinguagem de dados.

Devido à assimetria introduzida pelo próprio modelo de dados, muitas vezes o usuário vai se deparar com problemas de construção de perguntas inversas, problemas esses criados pelo modelo e não intrínsecos à questão. Significa que muitos programas são mais complicados do que precisam ser com a conseqüência de que escrita, depuração e manutenção de programas irão requerer mais tempo de programação do que deveriam.

Por outro lado, o modelo torna-se apropriado para muitas situações de necessidade no mundo real; como é o caso de empregado e patrão. O exemplo de fornecedores e partes não se aplica por ser uma correspondência m:m (muitos-à-muitos). Porém mesmo um caso genuíno de hierarquia ocorre o problema da assimetria na recuperação, por exemplo: "Encontre os empregados do patrão" e "Encontre o patrão dos empregados". Em termos práticos compreensíveis à nível de estruturas armazenadas de dados, uma solução seria haver um apontador de patrão para empregado e de empregado para patrão; aproximando-se de um modelo de redes. Uma outra solução seria armazenar outra hierarquia de empregado para patrão, duplicando os registros armazenados.

Voltando agora para operações de armazenagem, vamos observar claramente que, quando o sistema de informações não se ajusta perfeitamente a um modelo hierarquico, anomalias irão ocorrer na execução das operações básicas de armazenagem (inserção, remoção e atualização). Referindo-se ao exemplo, temos:

Inserção

Não é possível, sem introduzir uma parte fictícia, inserir dados referentes a um novo fornecedor, até que este forneça alguma parte.

Remoção

Desde que a informação sobre remessas é incorporada no tipo de re-

gistro do fornecedor, a única maneira de remove-la é remover o registro do fornecedor perdendo toda a informação sobre o fornecedor caso ele não forneça outra parte.

Atualização

Se necessitamos modificar a descrição do fornecedor - e.g. mudar a cidade de S1 para VITÓRIA - nos deparamos com o problema de termos de pesquisar toda a estrutura para achar todas as ocorrências de S1 e modificá-las ou a possibilidade de introdução de inconsistência no banco de dados, por omissão de alguma ocorrência.

2.2.3. A abordagem por redes

O modelo de redes para representação de dados não traz a desvantagem de produzir correspondência 1:n, como no M.H. As ligações num modelo de redes tendem a ser cíclicas, e os relacionamentos n para m. Neste modelo, os dados são representados por registros e "links" (apontadores).

A figura abaixo exemplifica através de uma amostra de dados, como seria uma estrutura de dados baseada na abordagem por redes.

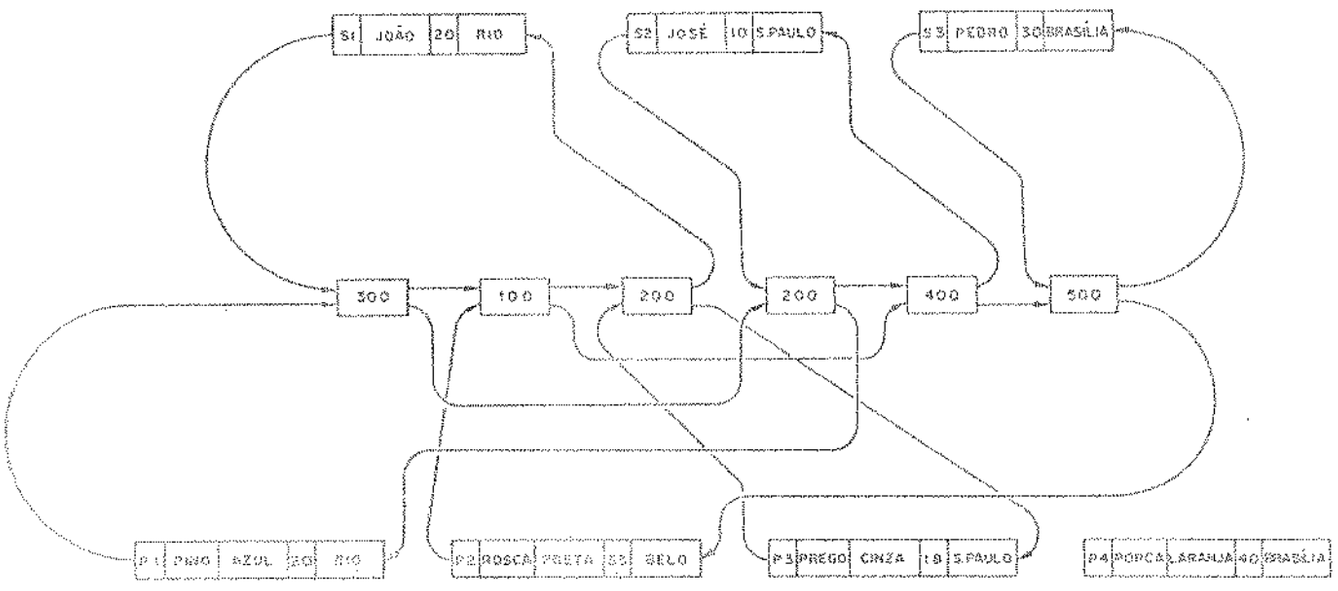


Fig. 8 - Amostra de dados em estrutura de rede

Para recuperar os números dos fornecedores que fornecem parte P2, por exemplo, faríamos basicamente a seguinte pergunta:

Obtenha a parte P2

NEXT : Encontre o próximo conector (remessa de partes fornecida pelos fornecedores) para esta parte

Conector encontrado? Se não, saída

Obtenha o fornecedor superior para esta remessa (conector)

Imprima Sn

GO TO NEXT

Como pode-se ver é bem mais complicado do que uma manipulação baseada na estrutura relacional; e o conteúdo de informação é o mesmo quer seja armazenado através de um modelo de rede ou relacional, ou ainda hierárquico. Contudo, no modelo de rede a simetria não é tudo e a complicação criada justifica-se pela variedade de tipos de registros e ligações entre os mesmos.

As operações de armazenagem não apresentam as anomalias discutidas para o modelo hierárquico. Apenas a remoção apresenta um problema a ser levantado a seguir

Inserção

Para inserir, por exemplo, dados referentes a um novo fornecedor, digamos S4, necessitamos apenas criar nova ocorrência de registro de fornecedor, mesmo que ele não esteja ainda conectado ao fornecimento de alguma remessa de uma parte.

Remoção

Ao remover a remessa, por exemplo, de P2 fornecida por S3, ambas as correntes de apontadores devem ser reestruturadas, problema este que deve ser resolvido automaticamente.

Atualização

Desde que uma informação apareça apenas uma vez na estrutura, a

atualização é efetuada em apenas um registro, sem conseqüências problemáticas.

Concordamos, portanto, que a desvantagem primeira da abordagem de rede é de fato a sua complexidade, tanto no modelo em si como na sublinguagem de dados associada. A variedade de estruturas de dados suportada reflete o fato de que o modelo de rede é realmente mais próximo de uma estrutura de armazenagem e não de um modelo conceitual de dados.

2.2.4. Escolha do modelo

O modelo hierárquico é o mais restrito porque só se aplica, de modo natural e adequado, quando os relacionamentos são hierárquicos. Para esses casos, tem a vantagem de ser simples e conduzir a implementações eficientes.

O modelo relacional e o de rede são igualmente de aplicação geral. O debate sobre a possível superioridade de um ou de outro não levou a uma conclusão que fosse aceita pela maioria dos autores (3).

Alguns sugerem que um SGBD completo deveria permitir que o usuário utilizasse o modelo que, em sua opinião, fosse mais adequado para cada banco de dados especificado. Indo mais além, sugerem que diferentes usuários de um mesmo banco de dados deveriam dispor de interfaces diferentes (modelos externos) que permitissem cada um visualizar e manipular o banco de dados conforme o modelo que escolhesse. Na arquitetura apresentada, essas interfaces são representadas pelos esquemas externos a serem definidos.

No próximo capítulo descreveremos um banco de dados associativo e mostraremos detalhadamente sua estrutura de dados.

CAPÍTULO 3

DESCRIÇÃO RESUMIDA DO DATAS

DATAS (DATa in Associative Storage) é um sistema para operação rápida e flexível com dados arbitrários conectados associativamente. O sistema permite entrada e saída de dados, armazenagem, estruturação, edição, remoção e recuperação de dados cujo nome, endereço ou associação a outros dados é conhecida.

As estruturas armazenadas no DATAS são estruturas associativas baseadas no conceito associativo de triplas "atributo-objeto-valor" chamadas tria das e no conceito matemático de conjunto. Os ítens básicos de informação são as entidades; cada entidade consiste de um bloco contíguo de dados e é endereçada por um endereço lógico imutável ou por nomes que devem se referir a uma única entidade. Os dados nestas entidades podem ser de qualquer tipo e seu tamanho pode variar até um máximo definido. Associações entre quaisquer destas entidades podem ser inseridas na estrutura dinamicamente e podem ser removidas em qualquer momento, assim como as próprias entidades podem ser armazenadas e removidas dinamicamente. Entidades especiais podem representar conjuntos (ordenados) de quaisquer tipos de entidades, mesmo de outros conjuntos ou de entidades representando atributos em associações. Estes conjuntos podem ser definidos explicitamente ou gerados como resultados de operações de recuperação na estrutura associativa; para operar com esses conjuntos as operações matemáticas de união, diferença e interseção podem ser utilizadas. A principal característica do DATAS é a sua total dinamicidade: entidades e associações podem ser introduzidas, mudadas e eliminadas em qualquer momento sem a necessidade de se reorganizar a estrutura. O tamanho total da estrutura pode ser aumentado dinamicamente no caso de não ser suficiente para uma certa aplicação.

As relações matemáticas definidas pelas associações "atributo-objeto-valor" são todas relações binárias cujos conteúdos podem também variar dinamicamente sem qualquer necessidade de uma especificação explícita de seus domínios, em contraste com a maioria dos bancos de dados relacionais. Para o usuário, o DATAS é um meio de armazenar e manipular seu modelo relacional de dados num computador de uma maneira simples e conveniente. Além disso, é possível

construir estruturas hierárquicas através do conceito de conjuntos.

A comunicação com o banco de dados processa-se através de uma linguagem assembler associativa simples, mas potente; e o usuário pode escolher a forma de utilização do banco de dados, seja através de um diálogo interativo ou através de um controlador batch.

3.1. Estrutura de Programas do Sistema

O sistema DATAS opera basicamente em três níveis. Os níveis de programa são:

- Um interpretador de linguagem Assembler Associativo usada em modo de diálogo ou em "batch"
- Subrotinas de implementação do Assembler Associativo que podem ser chamadas por programas de aplicação
- Rotinas auxiliares, não acessíveis ao usuário comum.

As subrotinas executam a manipulação de dados, nomes, conjuntos, relações e arquivos externos, assim como a recuperação desses itens, a inicialização e a manutenção da estrutura, e o transporte de páginas. Essas subrotinas são chamadas pelo programa interpretador da linguagem assembler ou por qualquer programa de aplicação desde que a linguagem do programa permita chamadas às subrotinas, que são escritas em FORTRAN X.

O interpretador permite ao usuário chamar as subrotinas diretamente usando instruções do tipo assembler através de um terminal, mantendo um diálogo orientado pelo sistema.

Neste diálogo, a comunicação do usuário com o sistema é feita instrução por instrução ou seja, a cada instrução executada o sistema aguarda pela próxima instrução a ser especificada pelo usuário. Mensagens de erro são fornecidas, facilitando a utilização do sistema.

Um conjunto de instruções de controle de fluxo e de manutenção de registros em software é anexado às instruções do assembler associativo, para possibilitar a utilização do banco de dados por programas assembler completos,

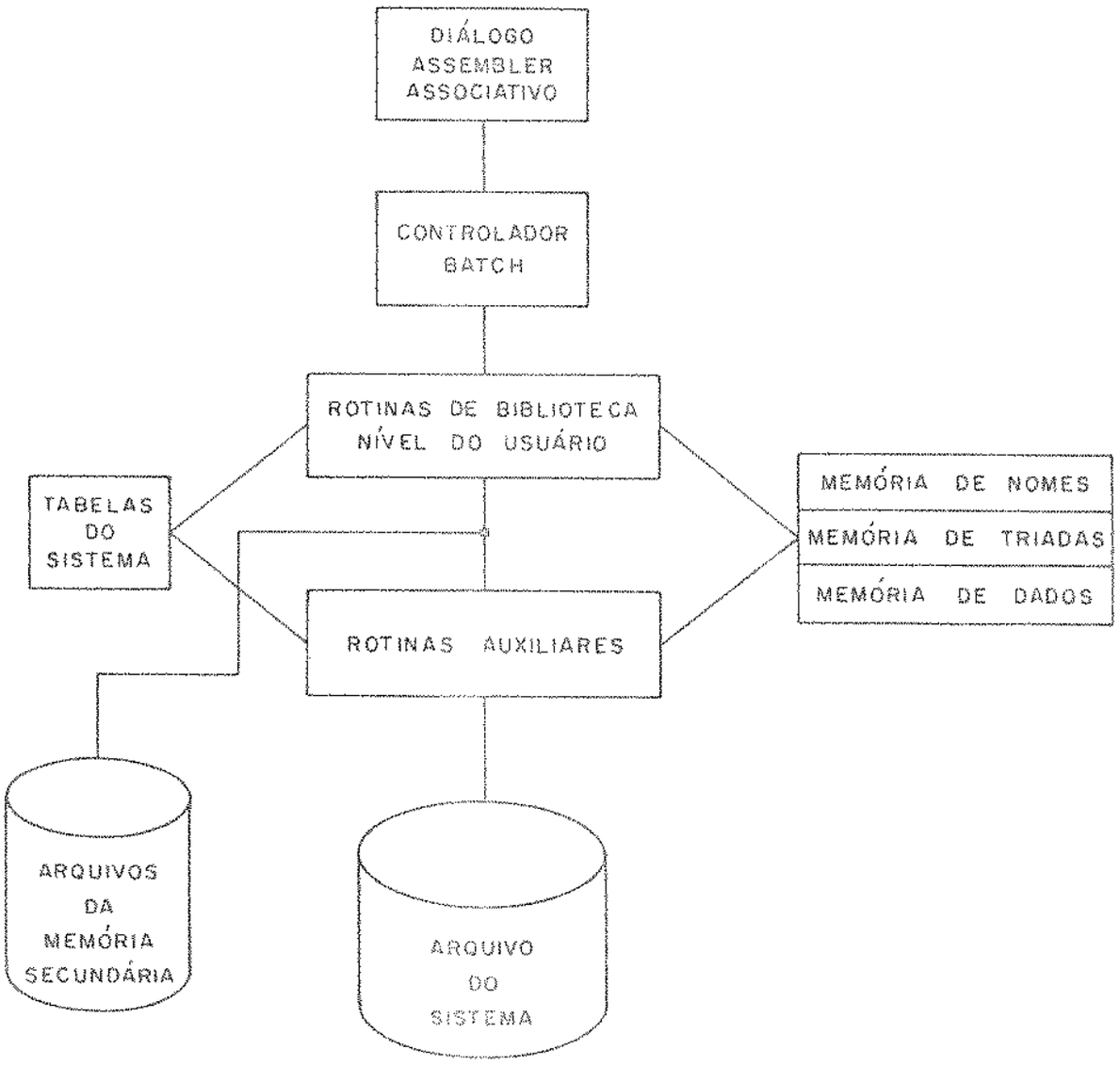


FIG. 1 - ESTRUTURA DO BANCO DE DADOS

armazenados em arquivos-texto no disco. Um controlador "batch" controla a execução destes programas. Os registros podem ser utilizados para armazenar informação intermediária durante a execução de um programa em "batch".

As rotinas auxiliares operam diretamente sobre nomes, triadas, entidades e seus blocos contíguos. Elas não são acessíveis ao usuário comum, pois seu uso indevido pode destruir o banco de dados.

O sistema DATAS está disponível em FORTRAN de acordo com as especificações do compilador FORTRAN/F1Ø do DEC1Ø da UNICAMP.

3.2. Componentes do Sistema

3.2.1. Elementos básicos

Os elementos básicos do DATAS são:

. nomes

. entidades

. triadas

Entidade é o menor ítem de informação endereçável no DATAS. Ela é endereçada através de um "endereço lógico", o identificador interno ID. O usuário pode também criar nomes ligados às entidades como uma outra possibilidade de endereçamento.

Um nome é uma cadeia de, no máximo, 10 caracteres ASCII, dos quais os cinco primeiros não podem ser todos nulos.

Qualquer entidade pode ter até 15 nomes, considerados sinônimos. Além disso, qualquer entidade pode conter dados arbitrários cujo significado é desconhecido e não interpretado pelo sistema. Triadas são triplas de ligação entre entidades, usadas para descrever associações entre entidades.

3.2.2. Entidades

Três classes de entidades são especificadas no DATAS:

Classe 1 - entidades relação

Classe 2 - entidades valor simples

Classe 3 - entidades valor múltiplo

Entidades relação representam associações entre entidades. Estas associações são armazenadas na estrutura através de triadas, descritas na próxima seção.

Entidades valor simples contêm dados especificados pelo usuário ou um dado nulo. Elas podem ser referenciadas, também, como componentes de relações binárias ou elementos de conjuntos.

Entidades valor múltiplo representam conjuntos matemáticos e contêm, em adição aos dados especificados pelo usuário apontadores para seus elementos.

Internamente, as entidades são referenciadas diretamente pelos identificadores internos ID's. Cada ID consiste de um par de números inteiros, o primeiro dos quais é o número da página de dados contendo a entidade, e o segundo é o número da entidade nesta página; pode ainda consistir de um único número inteiro composto destas duas partes.

3.2.3. Triadas

3.2.3.1. Entradas da relação

Associações lógicas entre entidades são consideradas no DATAS como sendo elementos de relações binárias. Essas associações são armazenadas como triplas ordenadas ABC de ID's, chamadas triadas.

$$A \subseteq B' \times C'$$

$$A = \{(B, C) \mid B \in B' , C \in C' , H(B, C)\} = \bigcup_{\substack{B \in B' \\ C \in C' \\ H(B,C)}}^{B}{}_{C} \{(B, C)\}$$

Portanto a relação A é um conjunto de pares ordenados (B,C) que satisfazem à condição H(B,C). B' e C' são domínios da relação. O par (B,C) é chamado uma entrada da relação A, se o mesmo está contido nesta relação. As relações no DATAS são binárias.

3.2.3.2. Tipos de permutação

A qualquer relação associa-se um parâmetro p (0, 1 ou 2) que determina quais as permutações de triadas desta relação que serão armazenadas no banco de dados. Os tipos de questões de recuperação que podem ser respondidas para uma determinada relação dependem das permutações armazenadas para a mesma.

Se p = 0 para uma relação A, apenas a permutação ABC de triadas é armazenada. Portanto, apenas as questões de recuperação:

F0(A,B,C): existe a associação ABC?

F1(A,B,?): que entidade(s) está (estão) associada(s) a B via relação A?

F4(A,?,?): existem entidades associadas entre si via relação A?

podem ser respondidas para esta relação A.

Para a relação A com p = 1, a permutação CAB é armazenada, além da permutação ABC. Assim, as seguintes questões adicionais de recuperação também podem ser respondidas:

F2(A,?,C): qual (quais) entidade(s) está (estão) associada(s) a C via relação A?

F3(?,?,C): quais as relações que associam C à quaisquer outras entidades?

Finalmente, para uma relação A com p = 2, todas as permutações cíclicas ABC, CAB e BCA de triadas são armazenadas, resultando na possibilidade de responder adicionalmente às outras duas questões de recuperação associativa restantes:

F3(?,B,C): que relação (relações) associa(m) B e C entre si?

F6(?,B,?): quais as relações que associam quaisquer entidades a B?

Os tipos de permutações permanecem fixos para todas as triadas de uma mesma relação. A mudança de tipos de permutações exige que se faça uma rearmazenagem de todas as triadas da relação.

3.2.3.3. Classes de questões de recuperação

Os sete tipos de questões de recuperação mostrados na seção anterior podem ser agrupados em três classes:

Classe 0: F0: uma associação é verdadeira?

Classe 1: F1, F2, F3: que entidade(s) está (estão) associada(s) a duas entidades especificadas?

Classe 2: F4, F5, F6: existem quaiquer entidades associadas a uma entidade especificada?

O DATAS permite responder às três classes de questões de recuperação.

3.2.3.4. Modo de recuperação

Às questões de recuperação no DATAS associa-se um parâmetro m, modo de recuperação, que controla a forma da resposta à questão e as operações a serem efetuadas durante a recuperação para questões de classe 1. Este modo tem diferentes significados dependendo se foi especificado ou não um conjunto para conter os resultados da recuperação. Deve ser observado que recuperação com re-

sultado-conjunto é definida somente para questões de recuperação de classe 1.

3.2.4. Capacidade

Os valores a seguir devem ser considerados absolutos pois acréscimos causariam alterações substanciais no sistema. Os limites máximos são:

Página de memória de nomes - 682 nomes

Página de memória de triadas - 750 triadas

Página de memória de dados - 255 entidades, cujos dados totalizados não ultrapassem 1789 palavras.

Entidade valor múltiplo - 1789 elementos

Entidade relação - 1789 páginas de memória de triadas

O número total de páginas que pode ser manipulado pelo sistema foi fixado em 2039 páginas. Este valor, entretanto, pode ser consideravelmente aumentado até um máximo de 65535 páginas de cada tipo de memória, modificando-se apenas algumas instruções nos programas fontes FORTRAN do DATAS.

3.2.5. Características adicionais

3.2.5.1. Entidades formatadas

Apesar do sistema utilizar os caracteres ASCII, é também possível especificar um formato FORTRAN de até 70 caracteres. Entidades possuindo dados deste tipo são chamadas formatadas.

3.2.5.2. Função archives

Para manipular dados de tamanho arbitrário e ilimitado (a não ser

pela quantidade de memória disponível fisicamente em disco) o sistema fornece meios para a armazenagem externa ao banco de dados, em arquivos de acesso aleatório.

3.2.5.3. Entidades valor múltiplo encadeadas

Para permitir a possibilidade de definir conjuntos de tamanho arbitrário e de expandir conjuntos mesmo quando os limites da página são atingidos, está sendo implementada a possibilidade de encadear quaisquer entidades valor múltiplo conectando-as através de apontadores. Nesta implementação, a primeira entidade da cadeia, a ancora, recebe o nome do conjunto.

3.2.6. O Assembler Associativo

O interpretador DATAS permite a construção e manipulação de estruturas associativas no DATAS assim como a recuperação de informação armazenada na estrutura, através de uma linguagem associativa que, por sua estrutura e sintaxe, pode ser considerada como uma espécie de linguagem assembly de dados: o Assembler Associativo.

As instruções da linguagem consistem de um código mnemônico composto de três caracteres que especifica a operação a ser executada seguido, se necessário, por parâmetros alfanuméricos, numéricos ou uma expressão (9).

3.2.6.1. Registros em "Software"

O assembler associativo permite o uso de um total de 28 registros em "software":

4 registros de saída para parâmetros de nome (A1-A4)

4 registros de saída para parâmetros numéricos (N1-N4)

10 registros de objetivo geral para parâmetros numéricos (%1-%10)

10 registros de entrada para parâmetros de nome e dados (@1-@10)

Os valores colocados nos registros de saída podem na execução de alguma instrução, ser guardados, por uma das instruções SVA ou SVN, em um dos registros de entrada ou de objetivo geral, respectivamente. Qualquer valor que se queira usar mais tarde deve ser guardado imediatamente após ter sido produzido. O maior uso destes registros ocorre na utilização do controlador batch.

3.3. A Estrutura de Dados no Datas

Tendo como objetivo principal uma recuperação eficiente de informação através da especificação parcial de seu conteúdo, os elementos do DATAS ficam armazenados numa memória implementada por software, subdividida em três partes:

. Memória de Nomes - contém os nomes das entidades e os seus ID's.

. Memória de Triadas - contém as triadas que representam as associações entre as entidades.

. Memória de Dados - contém informações sobre as entidades e seus dados.

Cada uma destas três áreas de memória é subdividida em páginas do mesmo tamanho, 2048 palavras de pelo menos 36 bits de comprimento cada uma. Todas as páginas residem num meio externo de armazenagem de acesso direto (disco ou tambor). Uma página aqui corresponde exatamente a um registro de 2K palavras de comprimento. Um arquivo consistindo destes registros é um arquivo do sistema de banco de dados. Antes da primeira interação com este arquivo do banco de dados, ele é aberto (normalmente para escrita). Após a última interação com o mesmo, o arquivo é fechado e fica disponível para outros usuários. Posteriormente, é possível selecionar e trabalhar com outro arquivo do sistema. Por causa da possibilidade de escolha dinâmica do tamanho do arquivo (número de registros) e de sua expansão posterior (se necessário), cada usuário pode criar arquivos específicos do sistema para cada aplicação podendo assim manter arquivos e tempos de recuperação pequenos. Por outro lado, o sistema é também capaz de manusear quantidades enormes de dados de muitas aplicações e usuários diferentes num mesmo arquivo do sistema. Aqui, entretanto, pode ser necessário que se estabeleçam restrições de acesso a partes dos dados, proibindo algumas operações do

DATAS para alguns usuários.

O primeiro registro de um arquivo do sistema contém informação geral sobre o estado do arquivo e sobre cada uma das primeiras 2039 páginas de memória de nomes, triadas e dados. Os registros seguintes contêm páginas de memória de nomes, triadas e dados, nesta ordem. Se o arquivo contém mais do que 2039 páginas, a informação sobre o estado das páginas restantes é armazenada nos últimos registros do arquivo. O número mínimo de páginas é um para cada tipo de memória, ou seja, um tamanho mínimo de arquivo de sistema de 8K palavras, incluindo-se o primeiro registro.

Quando se opera com o DATAS, uma página de cada tipo é mantida na memória do computador. O transporte de página é feito automaticamente por um sistema de paginação sem qualquer necessidade de intervenção do usuário. Normalmente, isto é feito de uma maneira transparente ao usuário. Por outro lado, o usuário pode carregar uma página que ele necessita num certo momento, fornecendo esta especificação ao sistema. Para acelerar o transporte de páginas, apenas as que forem modificadas durante sua permanencia na memória são reescritas de volta ao dispositivo externo, destruindo-se a versão antiga da página lá residente. Páginas não modificadas durante sua permanência na memória são simplesmente substituídas pela leitura de outras.

A seguir, descrevem-se detalhadamente os três tipos de memórias do DATAS: memória de nomes, memória de triadas e memória de dados.

3.3.1. Área de Memória de Nomes

Cada página da memória de nomes está dividida em duas partes: uma área de hash e uma área de conflitos (vide fig. 2).

A área de hash é endereçada através de um algoritmo de hash que calcula um endereço a partir dos primeiros 10 caracteres do nome a ser armazenado ou recuperado. Este algoritmo pode ser facilmente substituído por um outro diferente, sem grandes mudanças no sistema, de modo a permitir a adaptação do DATAS a aplicações específicas para as quais o algoritmo normalmente usado não é adequado. Cada célula desta área de hash consiste de três palavras: as duas primeiras contêm os caracteres do nome lá armazenado; a última palavra da célula contém o ID da entidade e, se necessário, o número da primeira célula de uma lista conflito, associada a este endereço hash, ou um apontador para uma área de página de overflow.

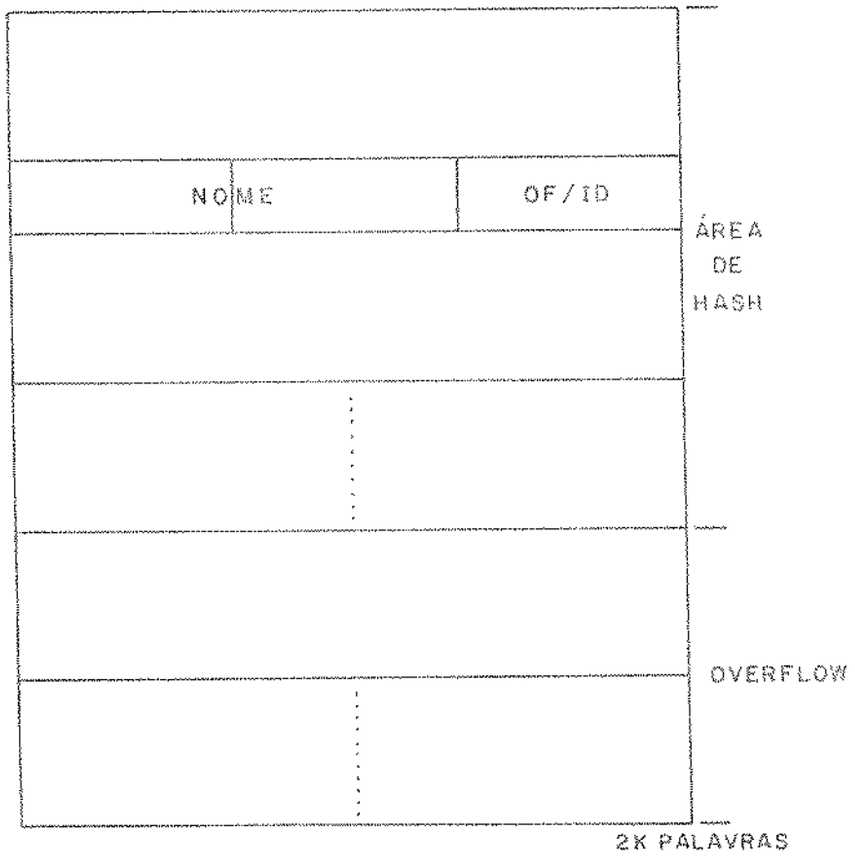


FIG. 2 - PÁGINA DE MEMÓRIA DE NOMES

ID ⟺ ID∅1 ⟺ Nº DA PÁGINA DE DADOS QUE CONTÉM A ENTIDADE
ID∅2 ⟺ Nº CORRENTE DA ENTIDADE NA PÁGINA DE DADOS

A área de conflito contém listas conflito e células vazias conectadas a uma lista livre. Se a lista livre está vazia e uma situação de conflito ocorre, o nome não pode ser armazenado nesta página. Para pequenos sistemas, de até três páginas de memória de nomes uma mensagem de erro é fornecida, e a tentativa de armazenar o nome é abortada. Para sistemas maiores, a última página é considerada como um conjunto comum de overflow para todas as outras páginas e a lista conflito a ser expandida tem seqüência nesta última página. Se também esta página estiver completa e ocorrer um conflito que não pode ser tratado em sua própria página, a tentativa de armazenar este nome é abortada. Neste caso, o usuário pode expandir a área de memória de nomes adicionando novas páginas de overflow. A capacidade da área de memória de nomes é aumentada, mas a recuperação pode se tornar mais demorada, pois poderá ser necessário procurar em todas as páginas deste conjunto de overflow em vez de obter diretamente a página certa a partir do algoritmo de hash. Por esta razão, existe um programa adicional que executa um rehash na área de memória de nomes quando há apenas uma página de overflow.

A subdivisão interna das páginas de memória de nome é arbitrária e pode ser escolhida diferentemente para cada página em particular. Em geral, entretanto, o usuário pode escolhê-la globalmente durante a inicialização do DATAS ou aceitar o valor "default" atribuído pelo sistema. A subdivisão é determinada pelo número de células da área de hash, Nc, para esta página. É recomendado que seja escolhido um número primo entre 300 e 600 para Nc; o valor default é 499. Valores >680 não são permitidos.

3.3.2. Área de memória de Triadas

Cada página da memória de Triadas é subdividida em três áreas:

- área de memória livre - F1
- área de endereçamento hash - HAS
- área de memória livre - F2

Considerando a armazenagem da permutação ABC apenas de uma Triada (p= 0), podemos dizer que F1 contém as células A na forma de um anel ocupado, enquanto que as células vazias de F1 são organizadas na forma de um anel vazio. (F1 : fig. 3). As duas primeiras palavras da área F1 contêm, respectivamente, o

endereço da última célula do anel ocupado e o endereço da primeira célula do anel vazio. As células A do anel ocupado consistem de duas palavras. A primeira palavra contém um identificador de controle de célula do anel ocupado e o identificador lógico da entidade (ID). A segunda palavra contém um apontador para um endereço na área HAS de uma célula B, topo do anel B correspondente à célula A em questão, e ainda o endereço da próxima célula A no anel ocupado em F1.

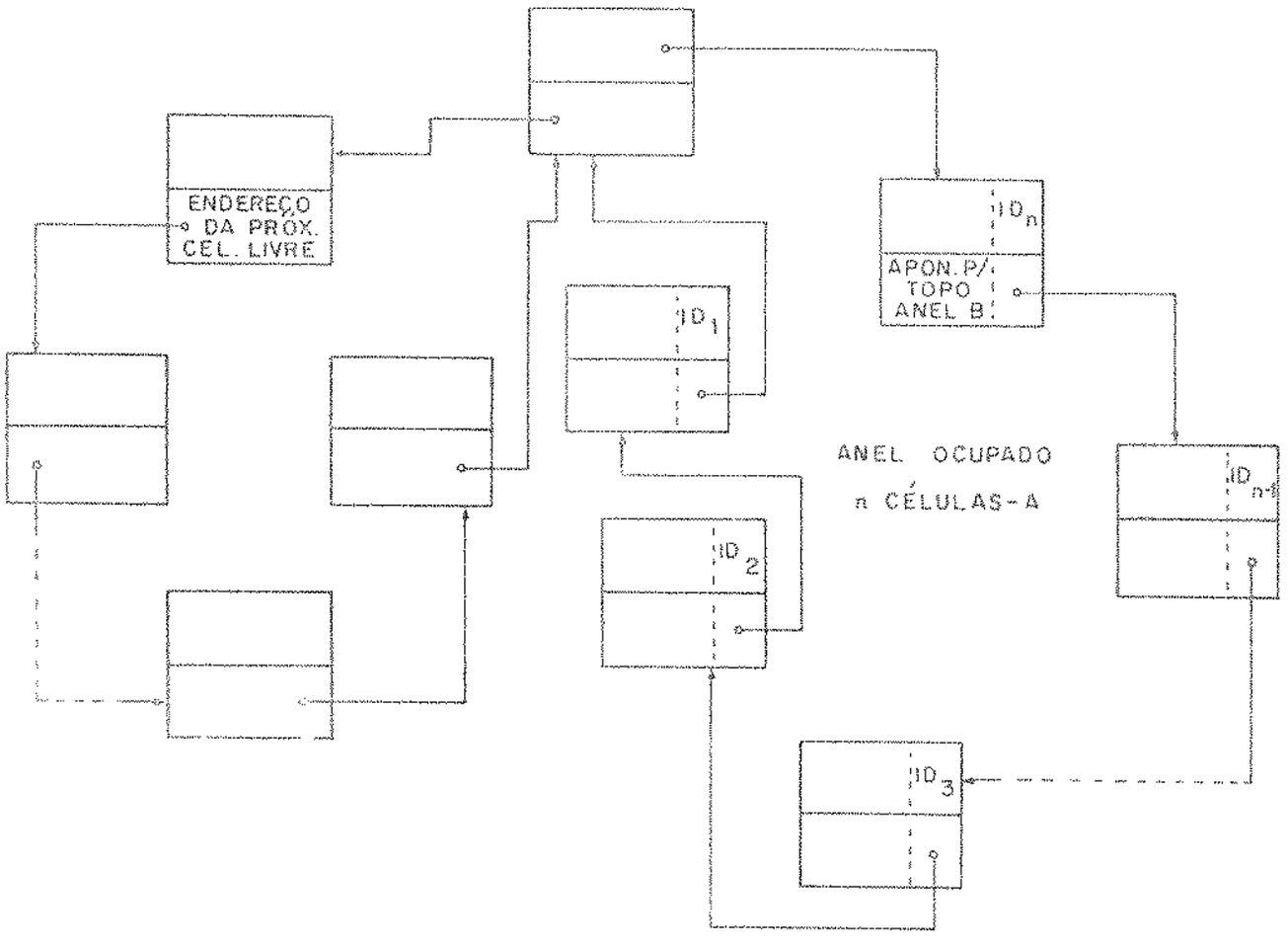


Fig. 3 - Estrutura de anéis de células-A na área F1

Na área de memória de Trindade um outro algoritmo de hash (que pode facilmente ser substituído) calcula um endereço em HAS a partir dos ID's de A e B. Se a célula endereçada está vazia, a célula B é armazenada neste endereço. Senão, esta célula é a cabeça de um anel conflito situado em F2 que contém todas as células B pertencentes ao endereço hash calculado. Além disso, todas as células B de uma dada célula A formam um anel B, com esta célula A (contida em

F1) na cabeça. Cada célula B, quer seja em HAS ou em F2, consiste de 4 (quatro) palavras. A primeira palavra contém um identificador de controle de célula ocupada e o identificador da entidade B; a segunda palavra contém o tipo de célula B do anel (4-topo, 1-continuação, 6-única e 2-fundo) e o endereço da próxima célula B associada à célula A ou o endereço de volta à célula A, caso o tipo da célula B seja 6 ou 2. A terceira palavra contém um bit de controle se existe apenas uma entidade C associada à célula B, e o identificador lógico desta entidade C; caso existam várias entidades C associadas à célula B, então a terceira palavra contém apenas o endereço, na área F2, da célula C topo do anel associado à célula B. A quarta palavra da célula B em HAS contém o endereço na área de conflito (em F2) do topo do anel conflito de células B pertencendo a este endereço hash. Se o endereço da célula B em HAS está ocupado, i.e., a célula B está em F2, a quarta palavra contém o tipo de célula B do anel conflito (4-topo, 2-fundo, 6-única, 1-continuação) e o endereço da próxima célula B do anel conflito ou o endereço de retorno à HAS. (Vide figs. 4 e 5).

A área F2, portanto, contém células B e células C; as células C constituem de duas palavras. A primeira palavra contém um identificador de controle de célula C que a distingue das células B em F2 e o identificador da entidade C. A segunda palavra contém o tipo de célula C do anel C (4-topo, 2-fundo e 1-continuação) e o endereço da próxima célula C do anel C associado à célula B ou o endereço de retorno à célula B associada, caso o tipo da célula C seja igual a 2. (Vide figs. 4 e 5).

O sistema é capaz de distinguir e manipular estas células através de suas posições na página e através de bits de controle armazenados adicionalmente nas células, indicando se uma certa célula está ocupada ou livre, se é uma célula B ou C, uma célula terminal (caso em que a célula B contém apenas uma entidade C associada) ou a cabeça de um anel C, ou se o apontador para a próxima célula num certo anel é a ligação de uma célula topo do anel ou fundo do anel.

A subdivisão das páginas em três áreas F1, HAS e F2 é novamente arbitrária e pode ser escolhida diferentemente para cada página. O usuário pode, entretanto, escolhe-la globalmente durante a inicialização do sistema especificando os números Nr e Nh de células em F1 e HAS, respectivamente, para uma página. Por outro lado, valores "default" são atribuídos pelo sistema. É recomendado escolher Nr < 150 e Nh como um número primo no intervalo (150, 300); os valores default são Nr = 60 e Nh = 149. Os valores escolhidos para Nr e Nh devem satisfazer a condição 2Nh + Nr<1020.

Para recuperação sem resultado-conjunto de modo m=1, a decisão se uma triada é exibida ou um conjunto é gerado depende do tipo de célula B encontrada: para células terminais, uma triada é exibida, enquanto que na presença de um anel C os elementos deste anel são colocados num conjunto.

Cabe ainda uma explicação mais detalhada sobre as estruturas de anel B e C, e do tipo de permutação de uma triada. Tentemos elucidar através de exemplos.

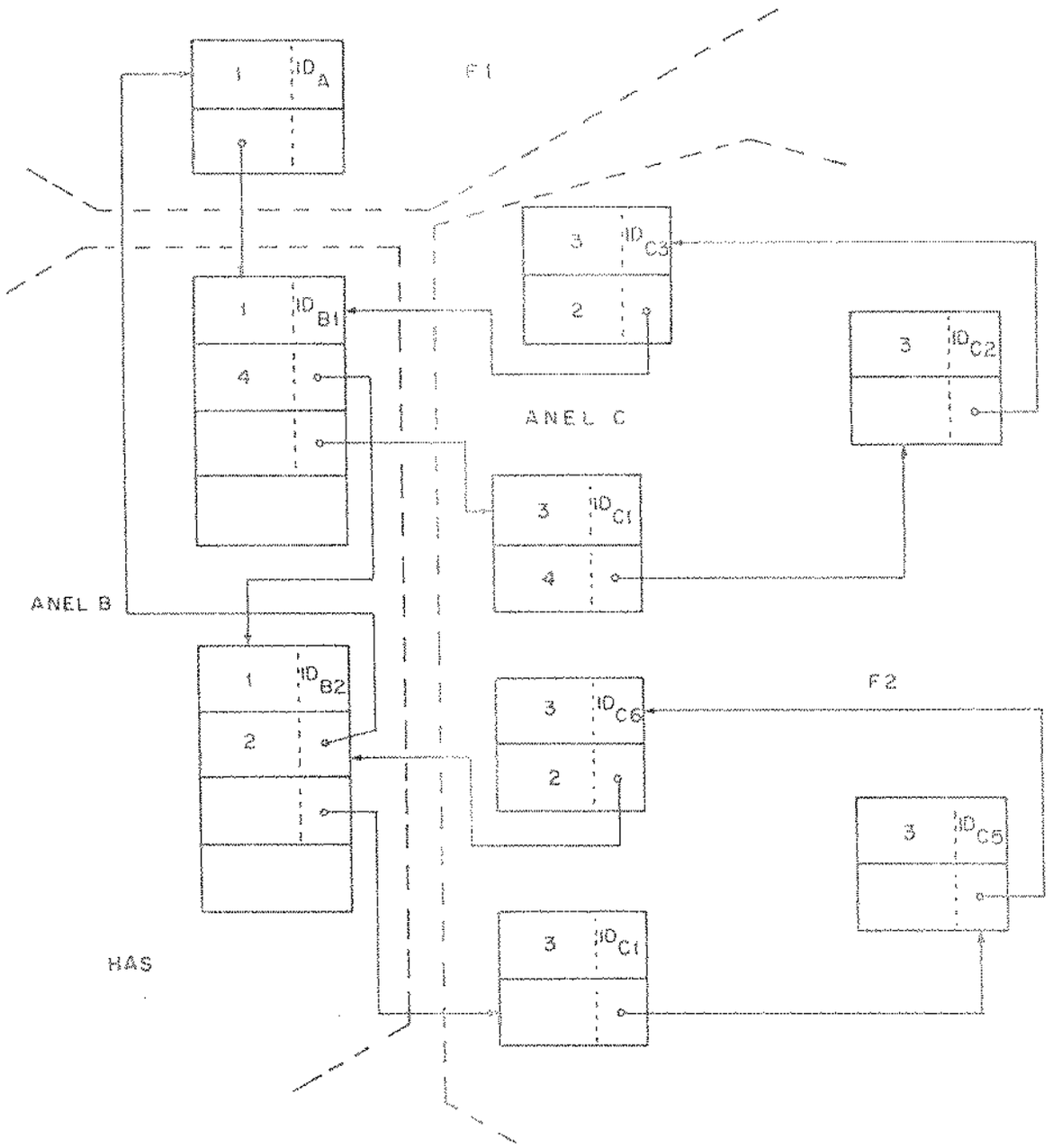


Fig. 4 - Estrutura de anéis na página de triadas

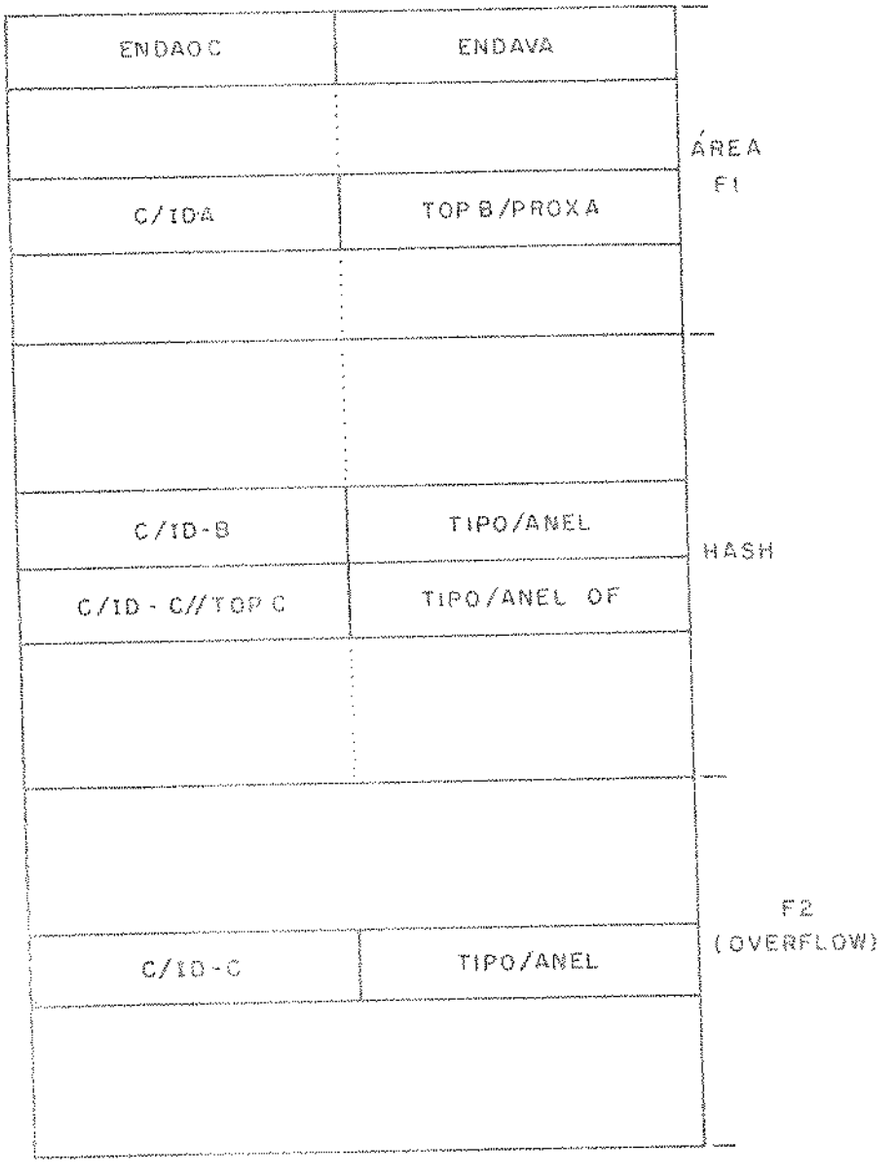


FIG. 5 - PÁGINA DE MEMÓRIA DE TRIADA

*

CÉLULAS T EM F2 SÃO DISTINTAS PELO IDENTIFICADOR DE CONTROLE C

Fazendo uso das seguintes instruções de armazenagem, com p = ∅

STM, A, B1, ∅, 3, C1, C2, C3
STM, A, B2, ∅, 3, C1, C5, C6

a estrutura da Fig. 4 é armazenada na memória de triadas.

Se no entanto, p fosse igual a 1, a triada CAB também seria armazenada,

STM, A, B1, 1, 3, C1, C2, C3
STM, A, B2, 1, 3, C1, C5, C6

e, além da estrutura anterior, a memória de triadas conteria a estrutura da Fig. 6.

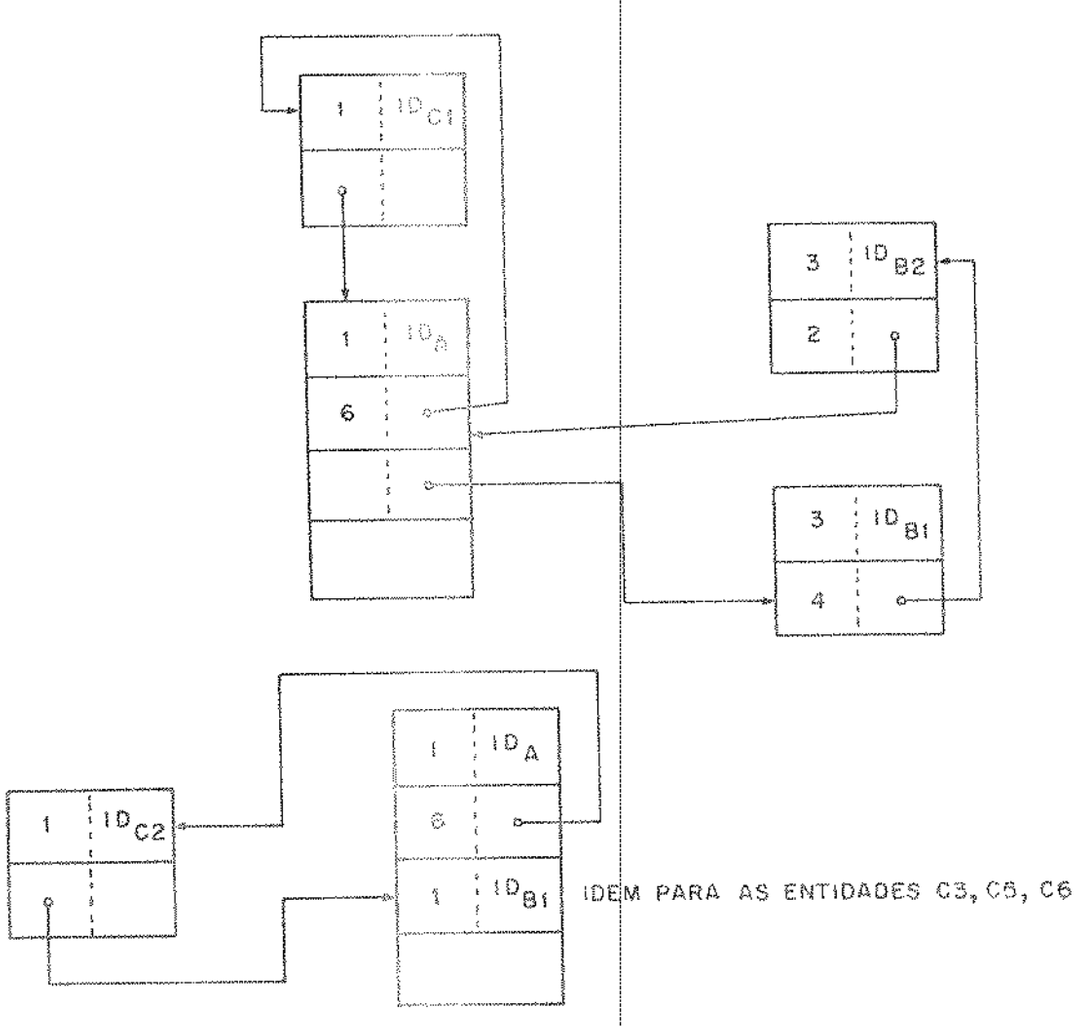


Fig. 6 - Estrutura de anéis para a permutação CAB

Para p = 1 a área F1 conteria também as entidades C1, C2, C3, C5 e C6. Observa-se, portanto, uma ocupação de memória muito maior para o mesmo número de entidades, com a vantagem de se permitir questões do tipo (?, ?, C).

Se p = 2

STM, A, B1, 2, 3, C1, C2, C3
STM, A, B2, 2, 3, C1, C5, C6

obteríamos, além das estruturas para p = 1, a estrutura, da Fig. 7, baseada na permutação BCA da triada ABC;

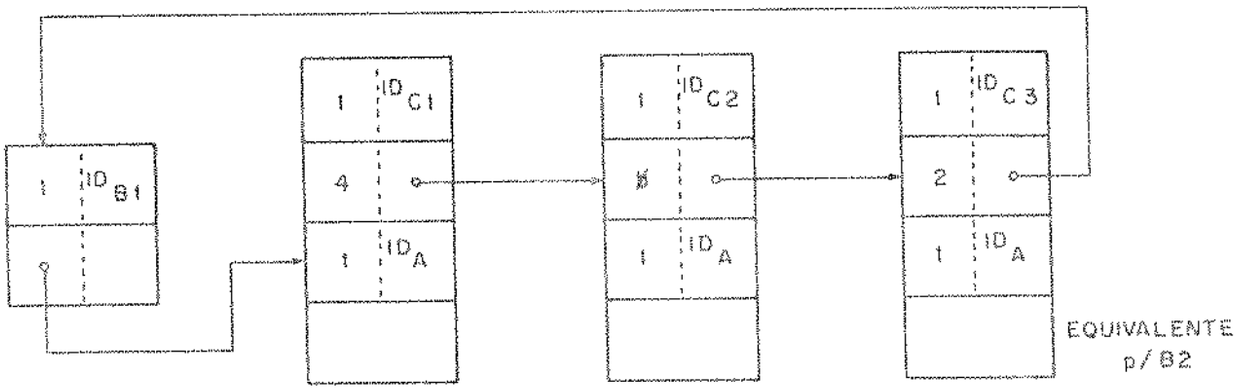


Fig. 7 - Estrutura de anéis para a permutação BCA da triada ABC

Vamos supor agora que os três pares (A1, B1), (A2, B2) e (A3, B3) possuem o mesmo endereço em HAS, causando portanto conflito. Assim, a partir das seguintes instruções de armazenagem

STM, A1, B1, ∅, 2, C1, C2
STM, A2, B2, ∅, 2, C3, C4
STM, A3, B3, C5, ∅

obtemos a estrutura esquemática da Fig. 8.

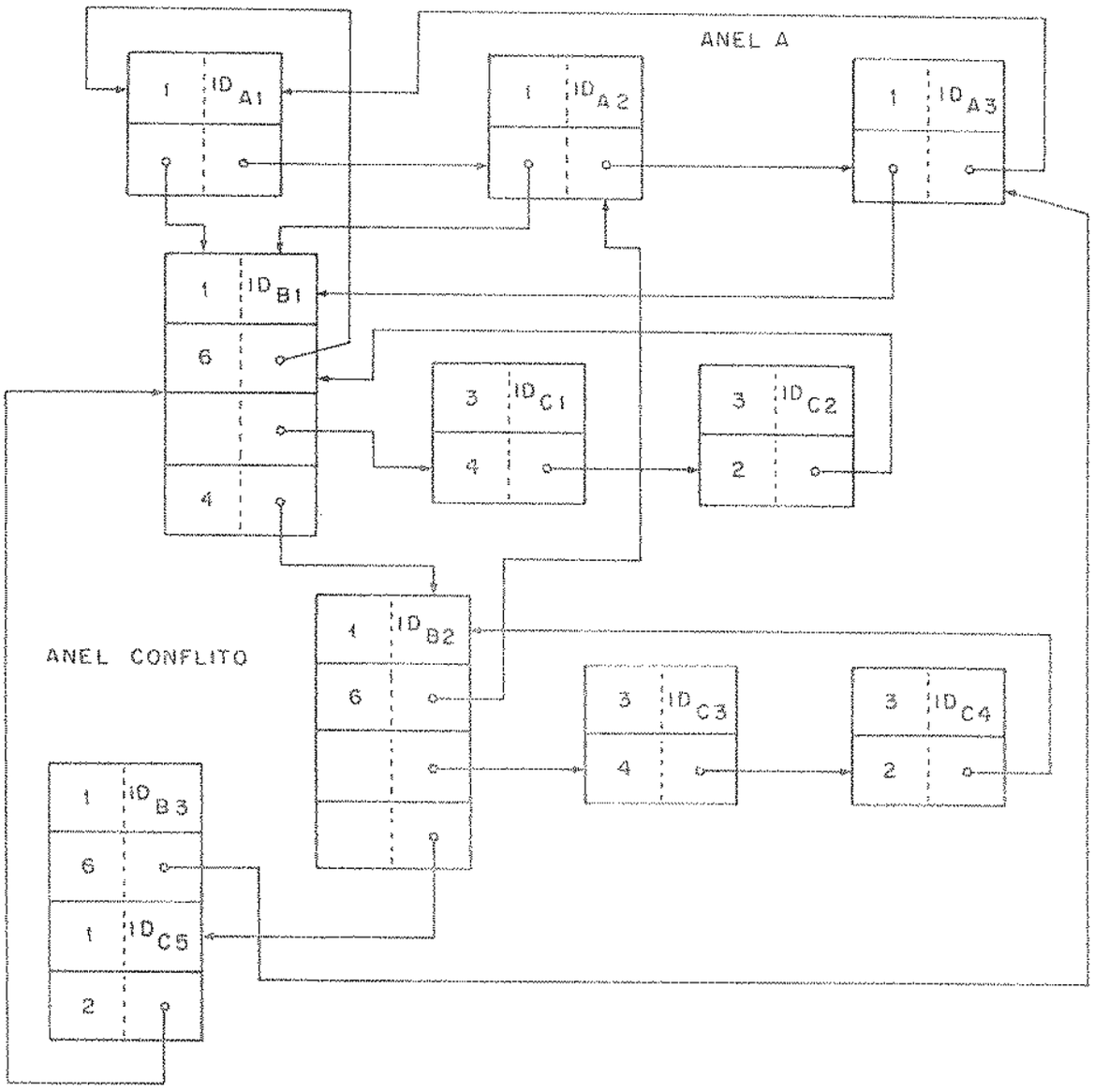


Fig. 8 - Estrutura de anéis em conflito (overflow)

3.3.3. Área de Memória de Dados

As primeiras 256 palavras de cada página de memória de dados contêm uma estrutura de anel duplo consistindo de um anel livre e um anel ocupado, cujas células são endereçadas através dos ID's (ou a segunda palavra dos ID's na representação de duas palavras). A primeira palavra desta área contém um apontador para a penúltima célula do anel ocupado e o endereço da última célula do anel ocupado. Cada célula desta área consiste de uma palavra contendo um

identificador de classe de entidade (em base octal, 10-entidade simples valor, 14-entidade multivalor e 04-entidade relação), um apontador para um endereço na área restante (a célula) contendo informação sobre a entidade em questão e o endereço da próxima célula do anel ocupado.

A estrutura desta primeira parte seria como se mostra na Fig. 9.

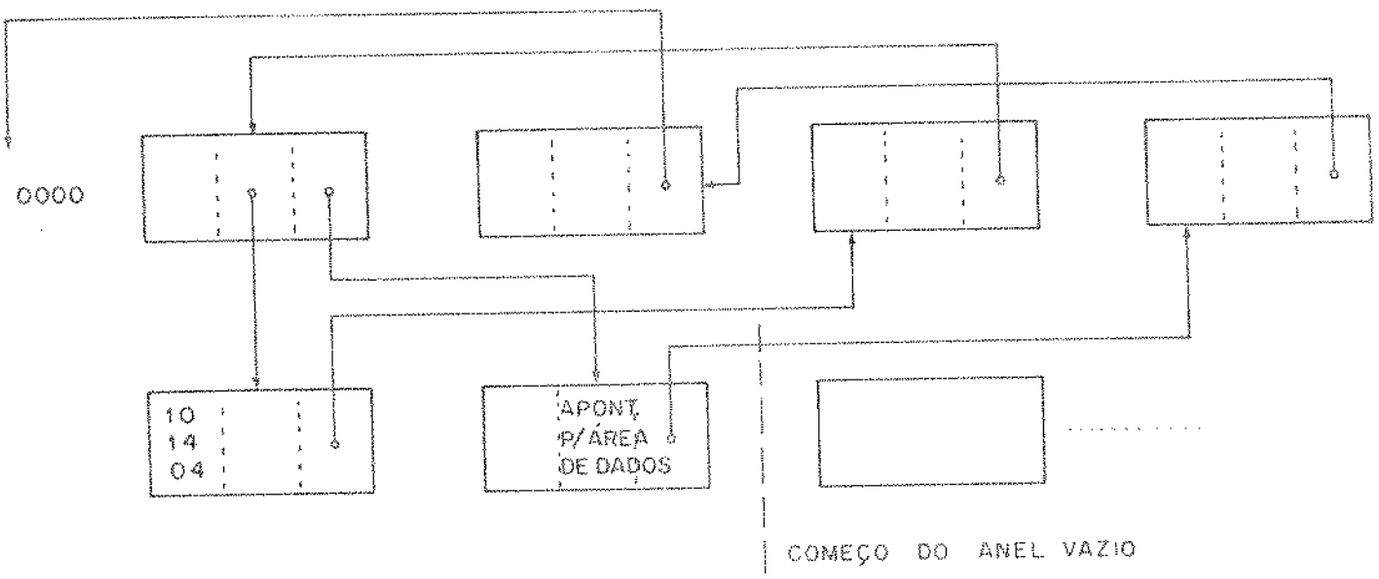


Fig. 9 - Estrutura de anéis de entidades

Na área restante, a partir da palavra 257, a primeira palavra contém o endereço da primeira palavra livre desta área. A seguir, as células desta área são sequenciais e de tamanho variável de acordo com a entidade. A primeira palavra de cada célula contém sempre o endereço da primeira palavra da próxima célula, de modo a determinar o tamanho da célula. A segunda palavra contém o número de sinônimos da entidade. Se for uma entidade relação conterá também o tipo de permutação e o número de páginas de triadas que contém triadas pertencentes a esta relação, e as palavras seguintes contém cada uma o número destas páginas de triadas. Se for uma entidade valor múltiplo, a segunda palavra conterá além do número de sinônimos, o número de elementos pertencendo ao conjunto representado pela entidade, e as palavras seguintes contém cada uma os ID's destes elementos. As próximas palavras para qualquer classe de entidade

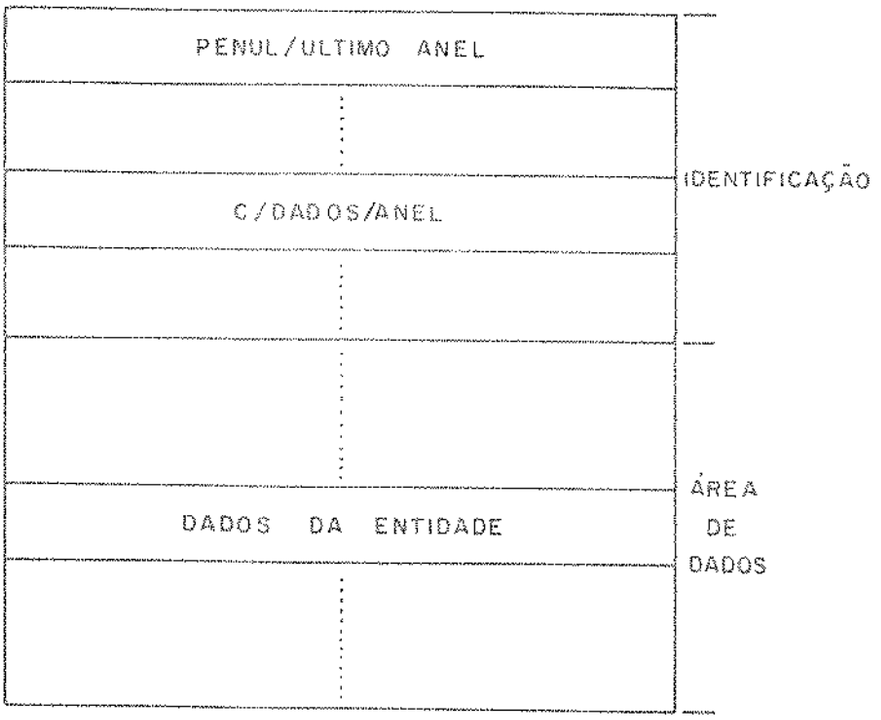
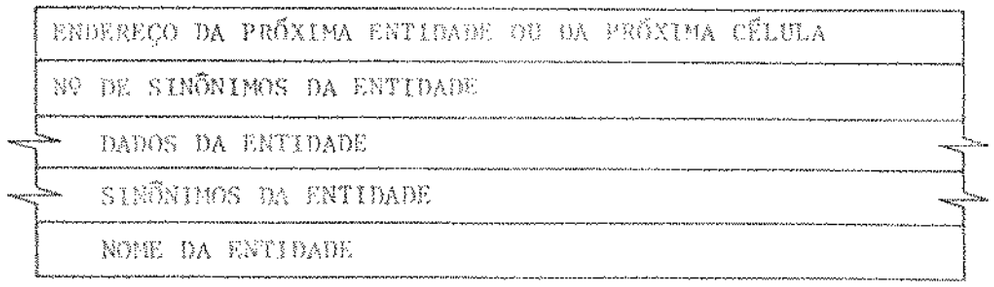


FIG. 10 - PÁGINA DE MEMÓRIA DE DADOS

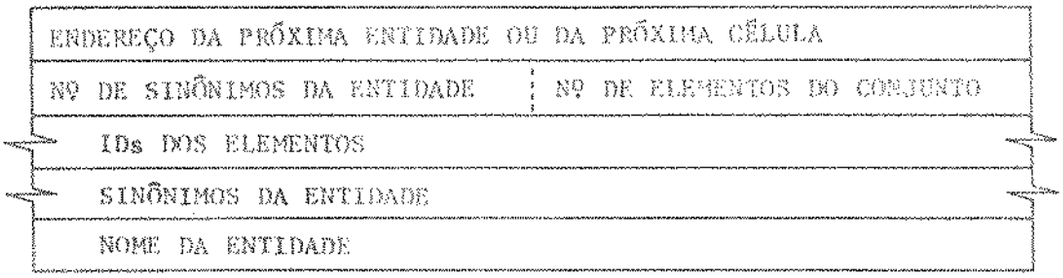
podem conter dados especificados pelo usuário; as últimas palavras contêm os sinônimos e o nome da entidade.

Podemos resumir da seguinte maneira:

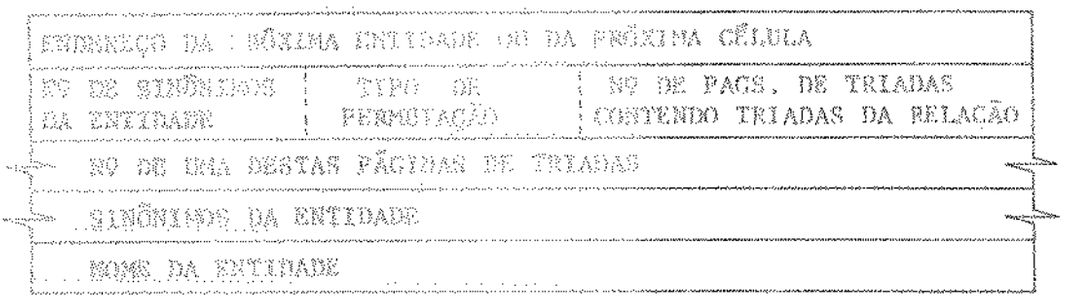
Entidade Valor Simples



Entidade Valor Múltiplo



Entidades Relação



3.4. Considerações Finais

Processamento associativo, isto é, o acesso à informação através de uma especificação parcial de seu conteúdo, tem sido por muito tempo um tópico de interesse em computação. O banco de dados associativo DATAS possibilita operações associativas além de possuir um software flexível à implementações adicionais que se façam necessárias para sua utilização em diversas aplicações. Apresenta ainda ao usuário, não só uma visão relacional através das triadas ("atributo-objeto-valor") como, também, a possibilidade de uma estruturação hierárquica, baseada na teoria de conjuntos matemáticos.

No capítulo seguinte, a estrutura semântica de um sistema de informações topográficas é analisada e representada segundo os elementos básicos do DATAS. A manipulação do sistema de informações é simulada. Várias operações de busca, modificação, inserção e remoção de informações são efetuadas segundo critérios da estrutura de dados implementada e da manipulação de objetivos.

CAPÍTULO 4

UM EXEMPLO DE APLICAÇÃO

INTRODUÇÃO

Neste capítulo apresenta-se um exemplo de estruturação de dados topográficos em DATAS, tendo como objetivo a manipulação eficiente, segura e flexível de dados para correção e atualização, e para a produção de desenhos e plantas. Um conjunto de programas em batch foi desenvolvido para facilitar a especificação das operações com os dados.

Para caracterizarmos melhor o exemplo apresentado, citamos algumas das principais dificuldades na utilização de dados topográficos:

- A quantidade de dados levantados e que devem ser codificados, armazenados e manipulados é muito grande.

- A variedade de tipos de plantas solicitadas tanto na forma e tipo de informação como na escala é grande.

- Os cadastros existentes precisam ser corrigidos (erros de levantamento) e atualizados (obras executadas após o levantamento).

A seguir descrevemos uma abordagem tradicional para o processamento automático de dados, através da utilização de arquivos em fitas magnéticas.

O processo consiste no encadeamento dos trabalhos de: coleta, tratamento numérico, manutenção e representação gráfica das informações topográficas e correlatas. Os dados podem ser obtidos através de levantamento de campo, recuperação de coordenadas de desenhos prontos, ou como resultados de outros programas.

No levantamento de campo, teodolitos eletrônicos são usados para perfurar os dados numa fita de papel. Na recuperação de coordenadas de desenhos prontos um digitalizador é usado e os dados são perfurados em cartões. Resultados de programas de cálculo como, por exemplo, dados de detalhamento civil, são

gravados diretamente em fitas magnéticas.

Para o tratamento numérico das informações, os dados são lidos de cartões, fitas ou discos magnéticos e são analisados do ponto de vista de consistência. As coordenadas de cada ponto, relativas ao sistema de referência utilizado, são calculadas. Em seguida, estas informações são organizadas, classificadas e relacionadas entre si, para se obter a sua forma final de armazenamento e são, então, gravadas em discos ou fitas magnéticas.

Essas informações podem receber tratamentos adicionais sempre que for necessário, por exemplo para o cálculo de novos dados. Além disso, os dados poderão ser selecionados, organizados e preparados para a obtenção de desenhos específicos através do "plotter".

Na representação gráfica, as interligações das informações são usadas para transformar dados em uma planta composta por linhas de diversas espécies. Estas, por sua vez, são compostas por pontos interligados entre si. Esta estruturação permite que os dados e as suas interligações possam ser consultados, selecionando-se os tipos de linhas que interessam em uma região específica. As linhas selecionadas podem ser apresentadas graficamente na forma e escala definidas pelo usuário.

Os elementos que caracterizam a estruturação topográfica e os tipos de acessos à estrutura, são apresentados a seguir. Pontos base são pontos bem definidos e estabelecidos cujas coordenadas foram precisamente determinadas a partir de um sistema de referência. Partindo-se do princípio de que os Pontos base são conhecidos, pontos são levantados tendo suas coordenadas calculadas por métodos de irradiação polar usando como referência uma estacionada, definida por dois pontos base. Cada ponto levantado pertence a um conjunto. Um conjunto coresponde a uma figura ou linha a ser desenhada. Um serviço é determinado por vários conjuntos de pontos levantados durante uma etapa de trabalho em campo. A forma geométrica e a espécie de detalhe descritas pelas linhas e pontos, são representadas na estrutura pelos seguintes elementos: identificação de serviço, forma geométrica, espécie de detalhe, concessionária a que pertence o detalhe, número de conjunto da linha ou figura dentro do serviço, número de sequência do ponto no conjunto, dado auxiliar (diâmetro, bitola, afastamento, etc.).

O acesso às informações pode ser efetuado como segue: as coordenadas dos pontos base, os dados da estacionada e as medidas dos pontos são acessados sempre que houver cálculos de coordenadas nas fases de cadastramento e revisão de dados; e as coordenadas e detalhes do ponto são acessadas para geração

de dados para desenhos ou para consultas. Neste último caso, os tipos de acesso são: por coordenadas norte e este do ponto, por serviço e por espécie de detalhe.

A utilização de um banco de dados para manipulação de dados topográficos, torna-se evidente na medida em que a empresa toma conhecimento das vantagens fornecidas ao bom desempenho do trabalho. (ref. CAP. 2)

Na criação da nova estrutura de dados a ser armazenada no DATAS, algumas observações iniciais devem ser feitas: a estrutura depende fortemente dos elementos básicos do modelo de dados do banco; as características de ligações entre elementos de informação devem ser mantidas; uma informação pode ser armazenada em DATAS de várias maneiras e, dependendo dos tipos de acessos o grau de dificuldade no manuseio da mesma pode variar muito.

A estrutura armazenada em DATAS é a seguinte: Pontos base, Ponto e Estacionada são armazenados como entidade simples com dados. Os Serviços e os Conjuntos de pontos são armazenados como conjuntos matemáticos ordenados de Conjuntos de pontos e Pontos, respectivamente. Cada Estacionada é associada a dois Pontos base através de uma relação. Cada Ponto é associado a uma Estacionada e a um início e a um fim de Espécie de detalhe (guias, alinhamento predial, etc.).

Considerando os tipos de acessos frequentes para desenho e consultas, podemos analisar como as estruturas de dados criadas foram adequadas à sua utilização. O acesso por serviço é uma simples operação de busca hierárquica. O acesso por espécie de detalhe justifica a criação de uma relação entre Pontos e espécie de detalhe. Somos obrigados, no entanto, a armazenar para cada ponto o início e o fim de suas espécies de detalhes, para obter assim, a definição de uma linha completa representativa de uma espécie de detalhe.

Para o acesso por coordenadas norte e este, por exemplo, obter todas as linhas compreendidas entre norte 10 e 20, e este 9 e 12, necessitaríamos armazenar as coordenadas norte e este associadas ao ponto através de uma relação; na recuperação de uma determinada área, uma busca exaustiva na relação criada e uma comparação de pertinência à área seriam efetuadas. Podemos afirmar que certamente o tempo gasto nessas operações e o número de interações necessárias bem como o espaço de memória utilizado não justificam tal estrutura. Isto deve-se principalmente ao fato de não existir a possibilidade de definição de variáveis que pudessem ser usadas diretamente na recuperação das áreas. Como processo alternativo, células são utilizadas para desenhar as plantas solicitadas por coordenadas norte e este. As células têm dimensões fixas e compreendem conjuntos de pontos. Os pontos não necessários ao desenho da célula podem ser

eliminados através de algoritmos de clipping.

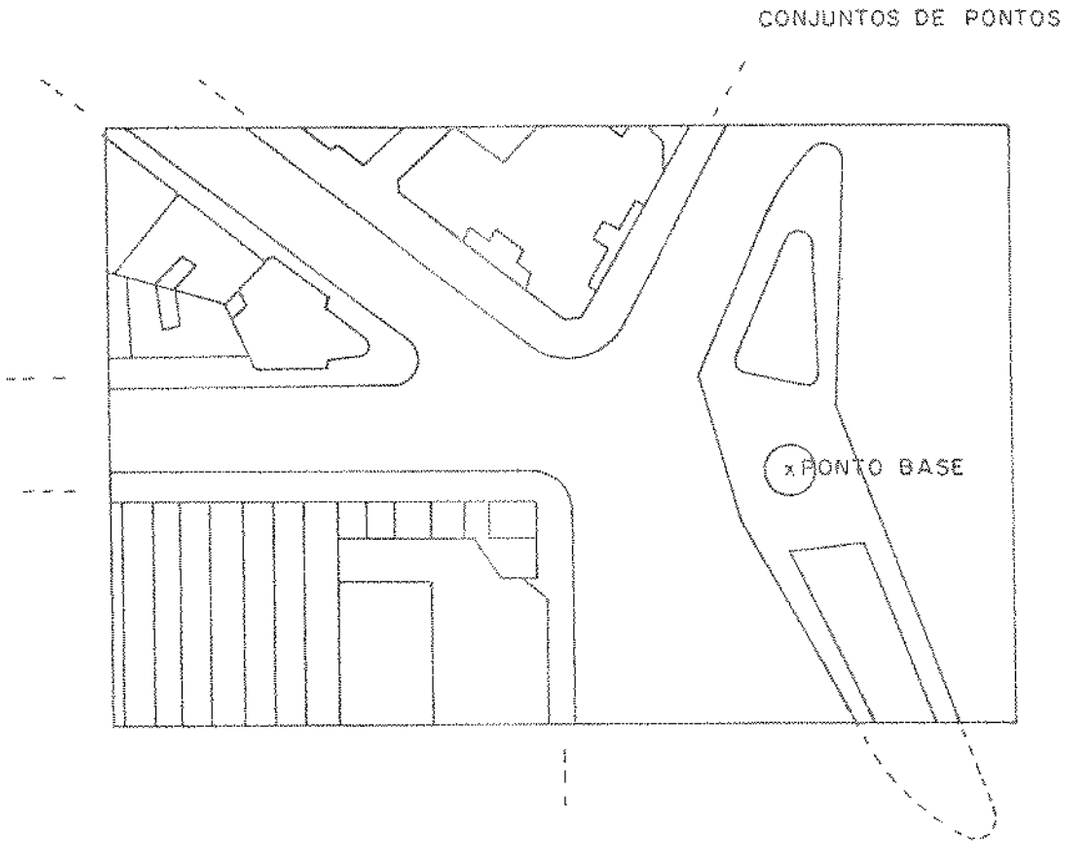


Fig. 1 - Exemplo de uma célula

A seguir, apresentamos maiores detalhes sobre a estrutura criada e fornecemos vários programas em assembler que executam as rotinas básicas de acesso (atualização e recuperação), em batch.

4.1. Estruturação da Informação

Pontos base ou Estações

Os pontos base devem ser armazenados no banco de dados DATAS para posterior consulta ou correção. Cada ponto base no DATAS é uma entidade valor simples de nome PB_i com dados NTPB, ETPB e ATPB (norte, este e altitude do pon-

to base).

Estacionadas

Uma estacionada é definida por uma estação vante e uma estação ré. Esta informação é armazenada no DATAS como uma relação que associa dois pontos base ou estações a um identificador de estacionada. Cada estacionada é no DATAS uma entidade simples de nome ID_i com dados HI (altura do aparelho medidor na estação) e ANGRE (ângulo horizontal da visada à estação ré) associada às entidades simples PB_i e PB_j através da entidade relação ESTAC.

Serviços, Células e Conjuntos de pontos

Cada serviço é representado no DATAS como uma entidade conjunto que contém conjuntos de pontos como elementos. Uma entidade representando um serviço é denominada $SERV_i$ e uma entidade representando um conjunto de pontos, $SERV_i DDD_j$.

O conceito de célula foi introduzido para permitir o desenho de seções de uma planta de levantamento topográfico, independentemente dos serviços relacionados com as mesmas. Cada célula contém conjuntos de pontos e é representada no DATAS como uma entidade conjunto de nome $CELNO_\ell$ que contém as entidades simples, conjuntos de pontos, $SERV_i DDD_j$, como elementos.

Conjuntos de pontos e Pontos

Cada conjunto de pontos é também definido como uma entidade conjunto de nome $SERV_i DDD_j$ que contém os pontos denominados $SERV_i DDD_j EEE_k$, armazenados como entidades simples com os dados: NORTE, ESTE, ALT, AA (forma geométrica do detalhe), CC (concessionária), AUX (dado auxiliar), IBB, FBB (respectivamente o início e o fim de espécies de detalhes do ponto), ANGHOR (ângulo horizontal do ponto irradiado), ANGVER (ang. vertical do ponto irradiado), HP (altura do prisma no ponto irradiado), DIST (distância do teodolito ao prisma no ponto irradiado). A seqüência terminal EEE determina a ordenação do ponto dentro do conjunto.

Espécie de detalhe e Estacionada

Este item é incluído por dois motivos: i) o cálculo das coordenadas do ponto é baseado nas coordenadas da estacionada e ii) necessidade eventual de geração de desenhos por espécie de detalhe.

Dessa maneira, as entidades simples ID_i, IBB_k e FBB_k respectivamente, identificador de estacionada, início e final de espécie de detalhe, são associadas à entidade simples $SERV_iDDD_jEEE_k$, através da entidade relação de nome DETALHE. Para facilitar algumas operações as espécies de detalhe IBB_k e FBB_k são também armazenadas como dados da entidade simples $SERV_iDDD_jEEE_k$.

4.2. Considerações sobre a Estrutura

A estrutura criada está baseada fundamentalmente no conceito matemático de conjuntos e em relações, e apresenta duas funções básicas:

. Manipular informações dispostas em estruturas de árvore.

. Manipular informações associadas através de relações binárias.

No caso de uma estrutura tipo árvore, uma entidade simples é redefinida como uma entidade conjunto, que contém elementos que são entidades simples. Estas, por sua vez, podem vir a ser entidades conjunto, adicionando-se mais um nível a hierarquia. As buscas são feitas usualmente de pai para filho. A busca invertida torna-se desnecessária na medida em que o nome do filho contém o nome do pai; no caso, o nome do ponto $SERV_iDDD_jEEE_k$ contém o nome do serviço $SERV_i$ e do conjunto DDD_j aos quais o ponto pertence.

No entanto, se nos interessa uma associação qualquer entre itens de informação, criamos uma relação que associa os itens armazenados como entidades simples. Por exemplo, a relação ESTAC associa duas estações à um identificador de estacionada.

A Fig. 2 representa esquematicamente o sistema de informações topográficas estruturado em DATAS (os dados de entidades não estão representados).

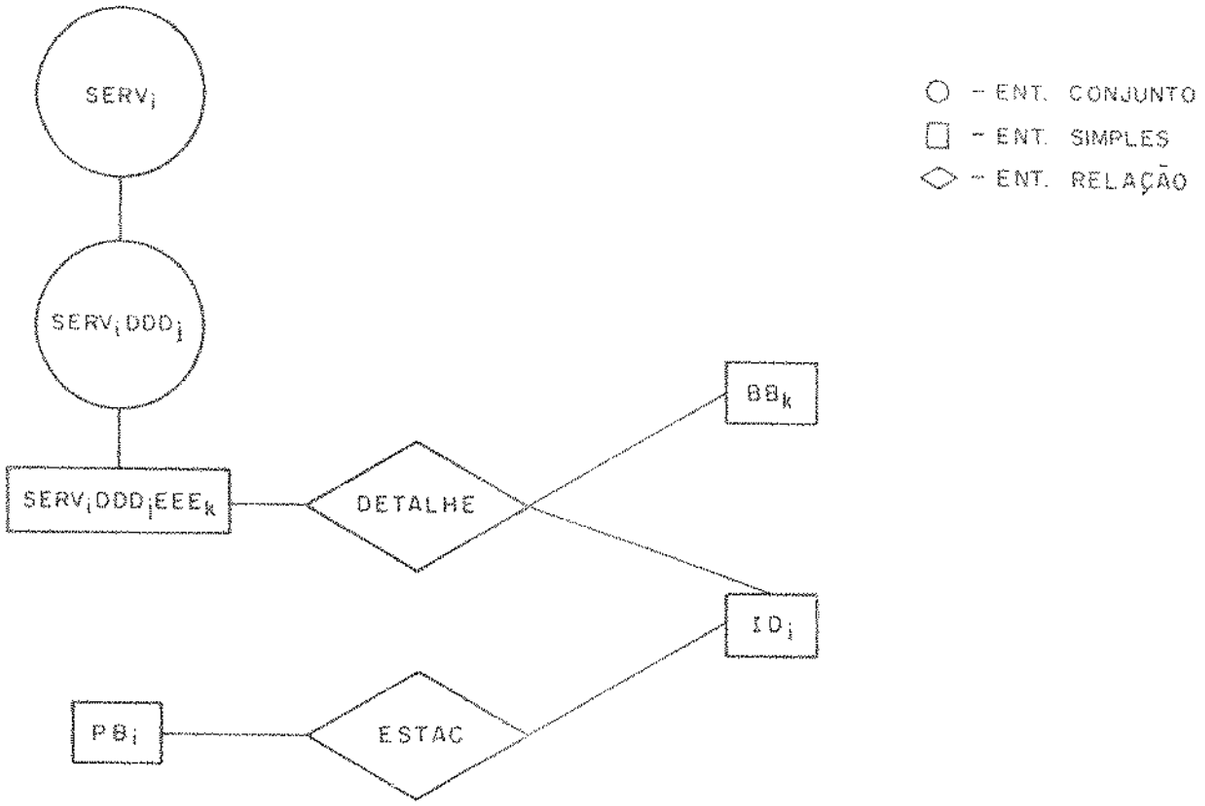


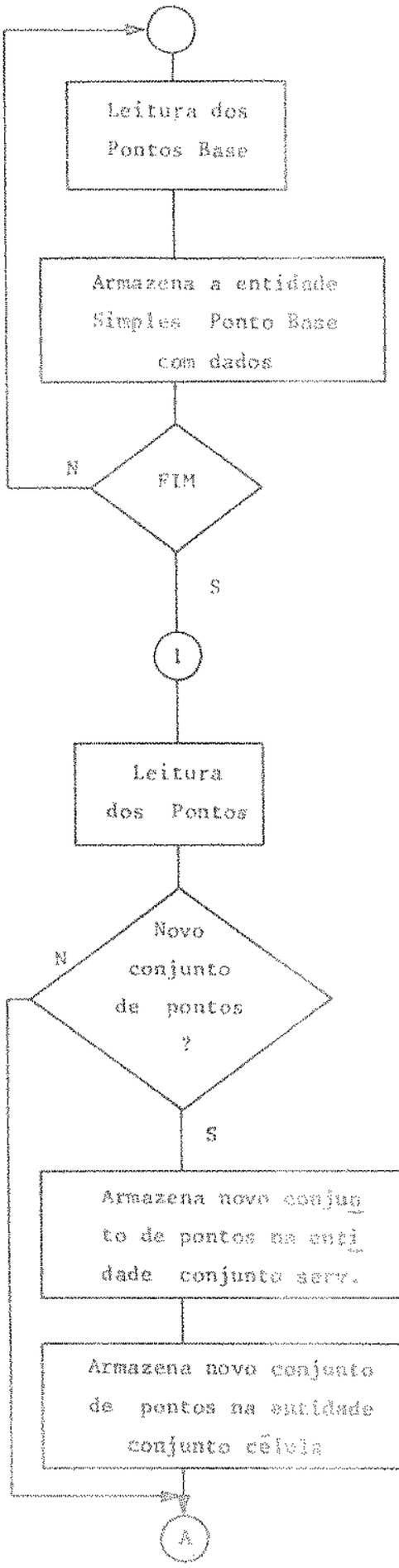
Fig. 2 - Representação esquemática da estrutura de informações

4.3. Geração da Estrutura em DATAS

O diagrama de operações mostrado a seguir representa o processo de armazenagem da estrutura topográfica segundo os elementos básicos do DATAS. Supondo-se que exista uma interface de leitura e armazenamento direto das informações em DATAS (o que não ocorre atualmente), os pontos base são lidos e armazenados com seus dados; quando termina a leitura, os pontos são lidos e seus dados e informações correlatas são estruturados e armazenados em um arquivo*. Ao lado do diagrama estão algumas instruções de armazenagem do assembler associativo.

---

* arquivo do sistema DATAS



STS, PB_i, NTBP, ATPB, ATBP *

STE, $SERV_i DDD_j$, $SERV_i$

STE, $SERV_i DDD_j$, $CELNO_k$

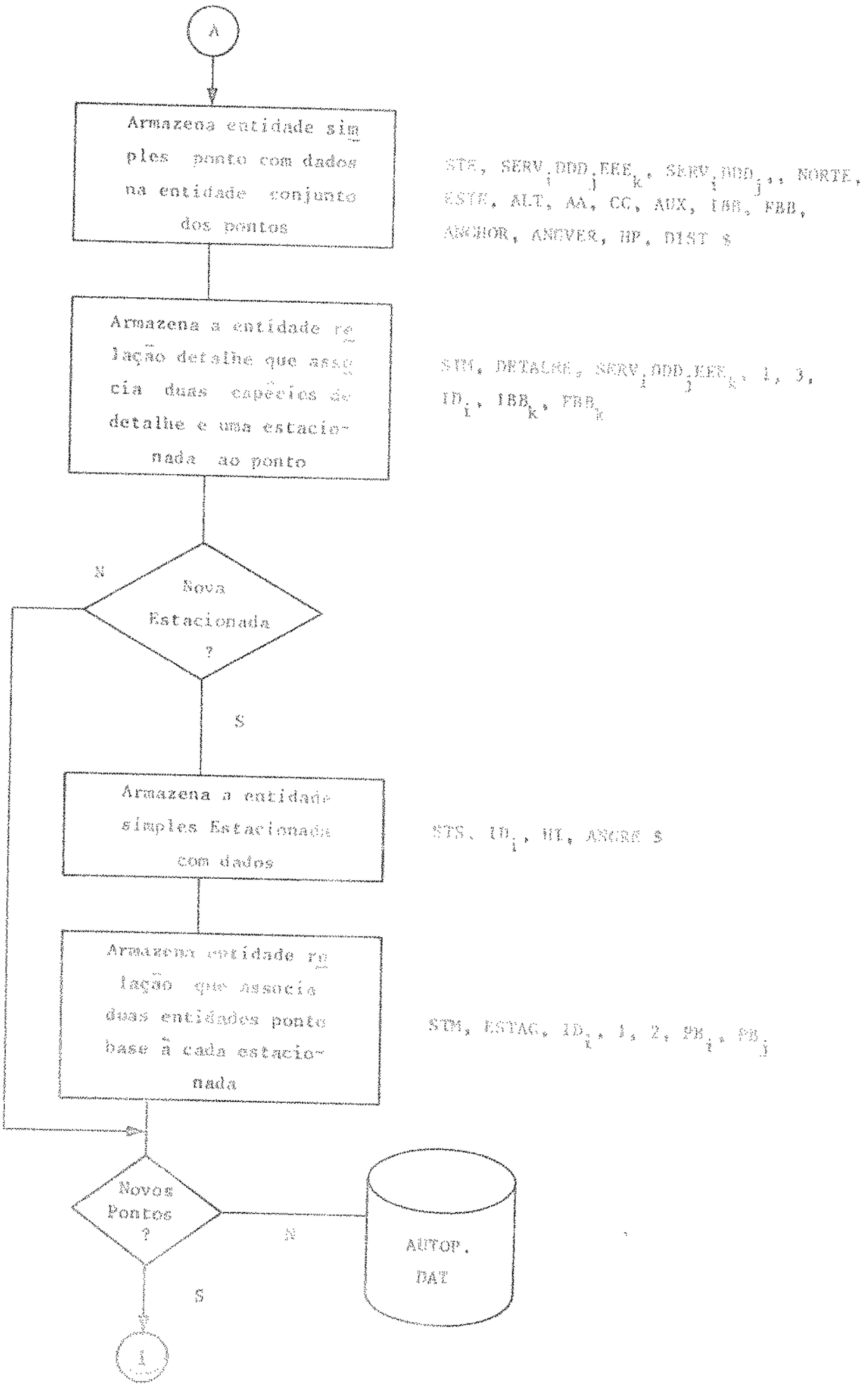


Fig. 3 - Diagrama de operações de armazenagem

4.4. Geração de Dados para Desenho

4.4.1. Por serviço ou célula

A seqüência de instruções abaixo foi implementada no DATAS e está armazenada no arquivo SERV.DAT, como uma opção alternativa ao uso interativo através de um diálogo orientado. Antes de utilizar-se o programa armazenado, as seguintes instruções devem ser executadas para passar-lhe qual a célula ou o serviço cujos dados são necessários para o desenho:

RDE, $SERV_i$ ou $CELNO_\ell$	Lê o nome do serviço ($SERV_i$) ou da célula ($CELNO_\ell$).
SVA, 7	Guarda o nome lido ($SERV_i$ ou $CELNO_\ell$) no registro 7.
SYR	Retorna o controle ao módulo supervisor do sistema.

Desde que ambos, serviço e célula, são entidades conjunto que contêm conjuntos de pontos como elementos, os dados necessários para desenho de um serviço ou de uma célula podem ser obtidos através da mesma seqüência de operações.

```
TY SERV.DAT

8Ø
DATAS
SYP,1,1
AUTOP.DAT
SET,9
INC,9,1
LIE,@7,Z9
JMP,9,Ø,1,71Ø4
SYA,1Ø
SET,1Ø,1
LIE,@1Ø,Z1Ø
JMP,6,1,1,71Ø4
SYA,9
RDE,@9
INC,1Ø,1
JMP,5,1,Ø
SYR
```

4.4.2. Por espécie de detalhe

A seqüência de operações para gerar dados por espécie de detalhe também está implementada em DATAS batch e armazenada no arquivo ESPECI.DAT. As seguintes instruções comunicam a espécie de detalhe escolhida:

RDE, IBB_K	Lê a ent. simples início de espécie de detalhe IBB_K.
SVA, 1Ø	Guarda o nome IBB_K no registro 10.
RDE, FBB_K	Lê a ent. simples fim de espécie de detalhe FBB_K.
SVA, 8	Guarda o nome FBB_K no registro 8.
SYR	Retorna o controle ao módulo supervisor do sistema.

O programa ESPECI.DAT busca os pontos que são o início da espécie de detalhe escolhida, lê seus dados, busca então os pontos que são o fim da espécie de detalhe e lê seus dados. Portanto, gera os dados necessários para desenho por uma espécie de detalhe.

```
TY ESPECI.DAT

8Ø
DATAS
SYP,1,1
AUTOP.DAT
FDR,DETALHE,,@ 1Ø
FDT
JMP,7,,1,7Ø5Ø
JMP,3,,Ø
FDN
JMP,6,,1,7Ø54
SYA,9
RDE,@ 9
JMP,4,1,Ø
SYA,9
RDE,@ 9
JEQ,4,,1Ø,1
INC,1Ø,1
FDR,DETALHE,,@ 8
JMP,13,1,Ø
SYR
```

4.5. Atualização de Informações

4.5.1. Alteração nos dados de um ponto base

Na fase de levantamento de dados em campo, pode ocorrer que os da-

dos de um ponto base sejam incorretamente cadastrados. Quando detectado, o erro deve ser corrigido e para tanto, uma seqüência de operações adicionais deve ser executada já que os pontos base têm seus dados utilizados como referência para o cálculo dos pontos levantados. Para obtermos os pontos que devem ter suas coordenadas recalculadas, alteramos os dados do ponto base e guardamos seu nome no registro 9.

CHD, PB_i, Ø, NTPB, ETPB, ATPB$	Altera as três primeiras palavras de dados.
SVA, 9	Guarda o nome PB_i no registro 9.
SYR	Retorna o controle.

Usamos então o programa-arquivo ALTERB.DAT que lê os dados das estacionadas e dos pontos a serem alterados a partir do ponto base alterado.

```
TY ALTERB.DAT

8Ø
DATAS
SYP,1,1
AUTOP.DAT
FDR,ESTAC,,@ 9
FDT
JMP,4,,1,78Ø1
SVA,8
SVN,,9,1Ø
JMP,3,Ø,Ø
SVA,,8
INC,8,1
RDE,@ 8
FDR,ESTAC,@ 8
FDT
SVA,1Ø
RDE,@ 1Ø
FDN
SVA,1Ø
RDE,@ 1Ø
FDR,DETALHE,,@ 8
FDT
JMP,4,,1,78Ø1
SVA,7
RDE,@ 7
JMP,9,Ø,Ø
SVA,,7
RDE,@ 7
JEQ,11,,8,1
FDN,Z9,Z1Ø
JMP,9,Ø,1,78Ø4
SVA,8
SVN,,9,1Ø
JMP,21,1,Ø
FDN
JMP,7,1,1,78Ø4
SVA,7
RDE,@ 7
JMP,4,1,Ø
SYR
```

Como o DATAS não possui facilidades para o cálculo de funções trigonométricas e aritméticas, as coordenadas dos pontos levantados devem ser calculadas fora do DATAS. Isto feito, alteramos os dados dos pontos para os novos valores calculados: CHD, $SERV_i DDD_j EEE_k$, Ø, NORTE, ESTE, ALT$.

4.5.2. Alteração no nome do Ponto base

DATAS não permite a geração de dois pontos com o mesmo nome. Uma mensagem é gerada pelo sistema questionando a remoção do primeiro nome gerado.

Para alterar o nome do Ponto base, a operação é a seguinte : CHN, PB_{NOVO}, PB_{ANTIGO}.

Isto faz com que o Ponto base assuma o novo nome e o nome velho seja deletado (o novo nome contém o mesmo identificador lógico interno do nome antigo).

4.5.3. Alteração de PB_i e/ou PB_j de uma Estacionada

Pode ser que também na fase de levantamento de dados em campo, os pontos base de uma estacionada sejam incorretamente cadastrados. Para corrigir este erro, recuperamos os pontos base que devem ser alterados para a estacionada em questão, removemos as associações dos pontos base à estacionada na relação ESTAC, e criamos apenas novas associações dos pontos base ao identificador de estacionada caso os pontos base já pertençam ao conjunto de pontos base do banco de dados. Senão, armazenamos antes os novos Pontos base como entidade simples. O fato de removermos sempre as associações dos dois pontos base ao identificador de estacionada deve-se ao fato de que a ordem dos dois pontos na relação que os associa ao identificador de estacionada determina a estação vante ou ré; este detalhe é importante para o cálculo das coordenadas dos pontos levantados que devem ser alteradas a partir da alteração de sua estacionada.

As seguintes instruções iniciam a operação de alteração:

FDR, ESTAC, ID_i	Encontra os pontos base (em anel)
FDT	Encontra o topo do anel
SVA, 9	Armazena nome PB_i no registro 9
FDN	Encontra o próximo elemento do anel

SVA, 8	Armazena PB_j em @ 8
DLM,ESTAC,ID_i,,2,PB_i,PB_j	Deleta as triadas <ESTAC,ID_i,PB_i> e <ESTAC,PB_i,PB_j>
SYG	Limpa a memória (Garbage collection)
STR, ESTAC, ID_i, PB_k	Armazena a triada <ESTAC, ID_i, PB_k>
SVA, 8, 9	Armazena ID_i em @ 8 e PB_k em @ 9
STR, ESTAC, ID_i, PB_e	Armazena a triada <ESTAC, ID_i, PB_e>
SVA,,,1Ø	Armazena PB_e em @ 1Ø
SYR	Retorna o controle ao módulo supervisor do sistema

O programa em batch que obtém os dados dos pontos base, da estacionada e todos os pontos que devem ter suas coordenadas recalculadas é mostrado a seguir, e usa os parâmetros armazenados em registros como dados de entrada:

```
TY ALTEST.DAT

8Ø
DATAS
SYP,1,1
AUTOP.DAT
RDE,@8
RDE,@9
RDE,@1Ø
FDR,DETALHE,,@8
FDT
JMP,6,Ø,1,7Ø%1
SVA,7
RDE,@7
FDN
JMP,4,Ø,1,7Ø%4
JMP,4,1,Ø
SVA,,7
RDE,@7
SYR
```

Os pontos obtidos como um resultado deste programa devem ter suas coordenadas recalculadas fora do DATAS e alteradas via a instrução:

CHD, SERV$_i$DDD$_j$EER$_k$, Ø, NORTE, ESTE, ALT$

4.5.4. Alteração nos dados HI e/ou ÂNGRE de uma estacionada

A altura do aparelho na estação e/ou o ângulo horizontal de visada a estação ré podem ter sido incorretamente cadastrados. Estes dados são necessá

rios para o cálculo das coordenadas dos pontos que contêm a estacionada como sistema de referência.

A alteração é feita através das instruções:

```
CHD, ID_i, Ø, HI, ANGRE$
SVA, 1Ø
SYR
```

Trata-se agora de encontrar os pontos que devem ter suas coordenadas recalculadas, ler seus dados e os dos pontos base associados à estacionada. O programa-arquivo ALTDAD.DAT executa este processo:

```
TY ALTDAD.DAT

8Ø
DATAS
SYP,1,1
AUTOP.DAT
RDE,@ 1Ø
FDR,ESTAC,@ 1Ø
FDT
SVA,9
RDE,@ 9
FDN
SVA,9
RDE,@ 9
FDR,DETALHE,,@ 1Ø
FDT
JMP,6,Ø,1,78Ø1
SVA,9
RDE,@ 9
FDN
JMP,4,,1,78Ø4
JMP,4,1,Ø
SVA,,9
RDE,@ 9
SYR
```

4.5.5. Alteração nos dados NORTE, ESTE, ALT, ANGHOR, ANGVER, HP e DIST de um ponto

As alterações acima podem ser por erros de cálculo ou de levantamento em campo. Os dados devem ser corrigidos e as coordenadas norte, este e altitude calculadas com os dados alterados do ponto e/ou os dados relativos ao sis

tema de referência tendo por base a estacionada.

Altera-se os dados através das instruções:

```
CHD, SERV_i.DDD_j.EEE_k, 8, ANGHOR, ANGVER, HP, DIST$
SVA, 1Ø
SYR
```

A seguir, chama-se o arquivo ALTPON.DAT que obtém as informações necessárias para que novo cálculo de coordenadas seja efetuado para o ponto. O programa lê os dados do ponto, de sua estacionada e de seus dois pontos base.

```
 TY ALTPON.DAT

RDE,@1Ø
FDR,DETALHE,@1Ø
FDT
SVA,9
RDE,@9
FDR,ESTAC,@9
FDT
SVA,8
RDE,@8
FDN
SVA,8
RDE,@8
SYR
```

4.5.6. Alteração nos dados AA, CC e AUX de um ponto

A forma geométrica (AA) a qual pertence o ponto, sua concessionária e detalhes auxiliares (e.g., diâmetro) são dados necessários à geração de desenhos. A alteração destes é feita diretamente, sem outras modificações, através da instrução:

```
CHD, SERV_i.DDD_j.EEE_k, 3, AA, CC, AUX$
```

4.6. Um Exemplo de Estruturação com Dados Ilustrativos

O exemplo a seguir mostra a estrutura de dados topográficos e sua

utilização. Uma amostra de dados representativa de uma situação real é apresentada:

1 serviço
2 células fixas
4 conjuntos
18 pontos
4 estacionadas
7 pontos base

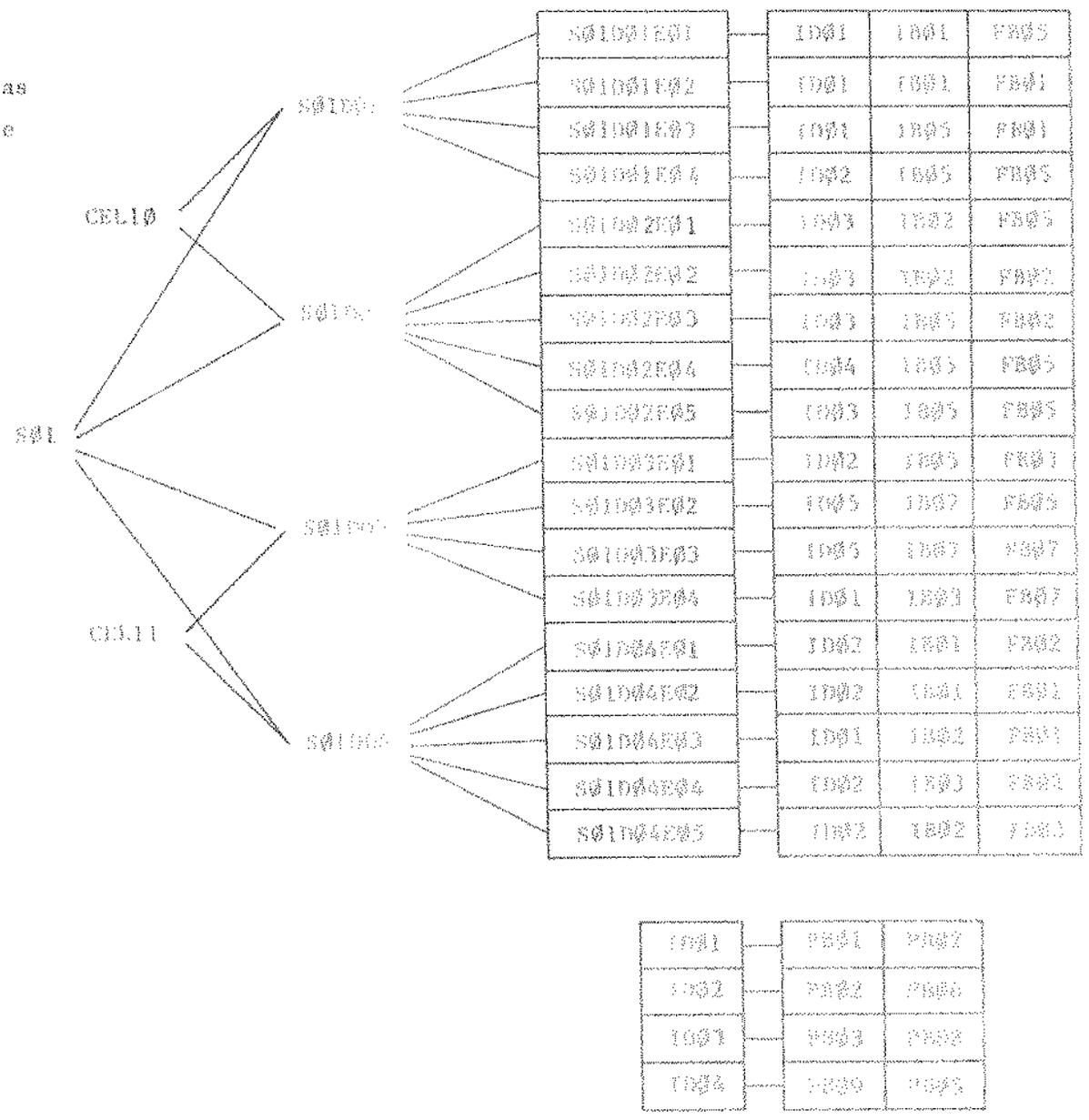


Fig. 4 - Exemplo da estrutura topográfica

Alguns exemplos de operações de armazenagem, atualização e recuperação usando a linguagem assembler associativo são listados abaixo.

1. Armazenagem

```
STS,PBØ1,1Ø.1,11.2,13.7$        Armazena a ent. simples PBØ1 c/ dados
STR,ESTAC,IDØ3,PBØ3             Armazena a triada ESTAC,IDØ3,PBØ3
STE,SØ1DØ3,SØ1                  Armazena a ent. simples SØ1DØ3 como ele-
                                    mento da ent. conj. SØ1

STM,DETALHE,SO1D04E04,1,
    3,ID02,IBØ3,FBØ2            Armazena três triadas na relação DETALHE
```

2. Atualização

```
CHN,PBØ1,PBØ7                   Altera o nome da ent. simples PBØ1 para PBØ7
CHD,SØ1DØ2EØ3,5,ØØØ2           Altera a 6ª palavra de dados da ent. simples
                                    SØ1DØ2EØ3 para ØØØ2
```

3. Recuperações

```
RDE,PBØ1                        Lê os dados da ent. simples PBØ1

comp. de dados=3    classe=2

1Ø.1,11.2,13.7

LIE,SØ1DØ2                      Lista os elementos da ent. conj. SØ1DØ2

    SØ1DØ2EØ1
    SØ1DØ2EØ2
    SØ1DØ2EØ3
    SØ1DØ2EØ4
    SØ1DØ2EØ5

nº de elementos=5

FDR,ESTAC,,PBØ2                 Encontra as estacionadas associadas ao PBØ2

- TRIADS ABOVE-STRINGS BELOW -

    IDØ3
    IDØ2
```

A obtenção dos pontos necessários para a geração de uma planta por espécie de decibel pode ser efetuada pelo programa-arquivo ESPECI.DAT. Supondo

que estamos operando em modo diálogo do DATAS, executamos as seguintes instruções para chamar o arquivo e lhe comunicar a espécie de detalhe a ser traçada:

RDE,IBØ5	Lê a ent. simples de nome IBØ5
SVA,1Ø	Guarda IBØ5 no registro 1Ø de nome
RDE,FBØ5	Lê a ent. simples de nome FBØ5
SVA,8	Guarda FBØ5 no registro 8 de nome
SYR	Retorna o controle ao supervisor
Ø	Opta por utilização em batch
ESPECI.DAT	Executa o arquivo ESPECI.DAT

Este programa gera os pontos e seus dados necessários para desenho pela espécie de detalhe codificada como BØ5 que poderia ser alinhamento predial ou sarjeta, por exemplo. Os pontos são:

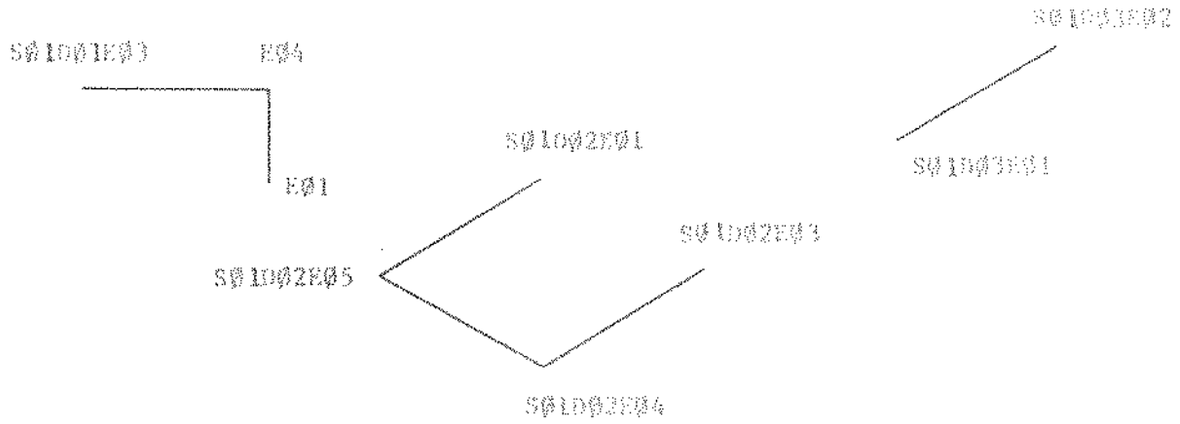


Fig. 5 - Representação gráfica dos pontos

A geração de dados para desenho por serviço e célula, e as atualizações necessárias à manutenção do sistema de informações são executadas para este exemplo, utilizando os programas criados em batch, e estão contidas no relatório, assim como o jogo de instruções do assembler associativo (9).

4.7. Conclusões

O exemplo apresentado neste capítulo mostra uma utilização do banco de dados DATAS, contribue para a análise de algumas de suas características a serem discutidas no próximo capítulo, e expõe algumas necessidades básicas de implementação adicional para a execução completa do exemplo.

A seguir listamos as principais implementações que devem ser efetuadas para uma adequação do DATAS ao exemplo apresentado e de um modo geral, para o estabelecimento de novas opções no sistema de banco de dados:

- Uma interface para leitura de dados em cartões e armazenagem de estruturas associativas em DATAS. Atualmente existe um programa que lê cartões e armazena entidades com seus dados.

- Criação de armazenagem temporária de dados, desde que o número de registros utilizados para a armazenagem de informações intermediárias pode ser insuficiente para algumas aplicações.

- Uma extensão da linguagem incluindo as seguintes operações:

  1) Funções aritméticas e trigonométricas, para o cálculo das coordenadas de um ponto do levantamento topográfico.

  2) Definição de variáveis como, por exemplo, estabelecer que a variável X pertence ao intervalo (a, b), através de uma instrução de declaração do assembler associativo tipo DCV, X, a, b.

  3) Operações com variáveis como, por exemplo, assumindo que as coordenadas do ponto estão associadas ao mesmo através de uma relação, obter todos os pontos que pertencem a uma área determinada.

  Ex: DCV, X, 10, 20
  DCV, Y, 15, 22
  FDR, NORTE, , X
  FDR, ESTE, , Y

Este último exemplo obtém todos os pontos dentro da área definida pelas variáveis X e Y.

Seguramente muitas outras observações podem ser feitas e num sistema de banco de dados novas implementações são sempre necessárias para sua utilização.

CAPÍTULO 5

CONSIDERAÇÕES FINAIS

Neste capítulo discutimos as vantagens e limitações encontradas no banco de dados DATAS, e fazemos algumas sugestões para prosseguimento do trabalho iniciado nesta tese.

5.1. Análise do DATAS

Nesta seção o DATAS é analisado quanto a algumas características desejáveis em um sistema de banco de dados.

5.1.1. Redundância

Em DATAS, apenas um arquivo do sistema de dados pode ser operado por um usuário de cada vez. Dentro de cada arquivo não existe redundância de informação desde que as entidades lógicas de informação são únicas dentro da estrutura. A tentativa de armazenar uma entidade já existente é abortada.

5.1.2. Inconsistência

Desde que uma mesma informação não pode ser armazenada duas vezes em um arquivo, não há inconsistência do mesmo. Contudo, entre arquivos distintos pode haver inconsistência de informações.

Os testes de consistência necessários devem ser feitos durante a entrada de dados no banco de dados. O sistema KEY-IN de coleta e verificação de dados para entrada no DATAS está sendo implementado e testado.

5.1.3. Compartilhamento

Os arquivos do sistema de banco de dados DATAS podem ser utilizados por qualquer usuário que possua a senha da área de trabalho onde estão armazenados os programas fontes de criação do sistema e o(s) arquivo(s) de trabalho.

Recentemente foi implementada uma versão multiusuário do DATAS no qual diversos usuários podem ler simultaneamente o mesmo arquivo; contudo, apenas um usuário pode efetuar acessos de escrita a um arquivo de cada vez.

A segurança de um banco de dados pode estar mais comprometida do que em sistemas de arquivos tradicionais caso não existam procedimentos de testes de autorização para o acesso a certos tipos de dados. Diferentes procedimentos devem ser estabelecidos para a recuperação, remoção, armazenagem e atualização de informações.

Em DATAS, além da senha imposta, podemos impedir que entidades sejam removidas do banco de dados. Os nomes das entidades protegidas devem iniciar-se com "$".

5.1.4. Integridade

O banco de dados deve conter somente dados corretos. Procedimentos de validação de dados devem ser estabelecidos. Com este intuito está sendo implementado e testado o sistema KEY-IN de coleta e verificação de dados para entrada no banco de dados.

5.1.5. Portabilidade

Os programas fontes escritos em FORTRAN-1Ø permitem modificações sem muito esforço para adaptação em computadores usando hardware e software diferentes do sistema DEC-1Ø. Entretanto, algumas peculiaridades devem ser observadas, na medida em que certas características usadas na implementação do sistema não possuem padrões definidos em FORTRAN-IV.

5.1.6. Capacidade de interação e resposta rápida

O DATAS opera em modo de diálogo, orientando as interações do usuário e fornecendo resposta rápida através da linguagem assembler associativo que também permite a utilização em batch do sistema.

5.1.7. Expansão automática da estrutura de dados

O DATAS possui um sistema de paginação que permite o crescimento da estrutura sem necessidade de reorganização da memória utilizada.

5.2. Sugestões para Trabalho Futuro

Para concluir listamos abaixo alguns tópicos de trabalhos futuros que julgamos importantes para o prosseguimento do trabalho iniciado com esta tese.

- Implementação de facilidades para a armazenagem de dados temporários.
- Implementação de uma memória relacional n-ária baseada no conceito de triplas associativas do DATAS.
- O desenvolvimento de uma linguagem de alto nível de organização e manipulação de dados.

BIBLIOGRAFIA

(1) DATE, C.J. : "An Introduction to Database Systems", 2nd edition - The Systems Programming Series - Adisson, Wesley Publishing Company.

(2) MARTIN, J. : "Principles of Database Management", Prentice Hall, Inc., Englewood Cliff, New Jersey.

(3) CODD, E.F. : "Recent Investigations in Relational Database Systems" - IBM Research - Digital Computer Laboratory (DEC, 10, 1974).

(4) CHEN, P. : "The Entity - Relationship Approach to Logical Data-Base Design", Q.E.D. Information Science, Inc., 1977.

(5) FURTADO, A.L. e SANTOS, C.S. : "Organização de Banco de Dados" Editora Campus.

(6) BO, KETIL : "Computer Aided Design Modelling, Systems Engineering, CAD - Systems", editado por J. ENCARNAÇÃO - CAPÍTULO 5 DATABASE DESIGN.

(7) ENCARNAÇÃO, J. : "An Graphical Information System" GRIFO - CONVÊNIO TH DARMSTADT - UNIV. DE CAMPINAS.

(8) FELDMAN, J.A. e ROUNER, P.D. : "An Algol-Based Associative Language" - (ACM - vol. 12, nº 8 - AUGUST 1969).

(9) DALTRINI, B.M.; JINO, M. e MESSINA, L.A. : Relatório Técnico Final - Convênio Metro S. PAULO - UNICAMP nº 5778925000 (MAIO 1980).

(10) DALTRINI, B.M.; JINO, M.; TOZZI, C.L. e MESSINA, L.A. : "Um sistema Integrado Núcleo Gráfico/Banco de Dados para Tratamento da Informação" - II Simpósio sobre aplicações gráficas por computador e sistemas gráficos interativos (SUCESU - AGOSTO 1980 - S. PAULO).

APÊNDICE

O ASSEMBLER ASSOCIATIVO É UMA LINGUAGEM RELACIONALMENTE COMPLETA

A demonstração a seguir não constitue uma prova completa pois alguns operadores algébricos não são considerados em sua forma geral. A prova apresentada consiste em mostrar que existem análogos, construídos com os operadores do assembler de dados, para cada um dos operadores comuns da álgebra relacional. A demonstração está baseada no exercício 7.1 de (1).

Demonstração

Baseado na estrutura atual de dados do DATAS, demonstra-se a completude do assembler associativo para relações binárias. Sejam A e B duas relações conhecidas quaisquer, para as quais mostramos o equivalente semântico para cada um dos operadores algébricos (operadores tradicionais de conjuntos e operadores relacionais especiais). Os equivalentes semânticos correspondem ao conjunto de programas em assembler apresentados a seguir:

<operador algébrico> ↔ <programa assembler>

A UNION B ↔

```
RDE,A
SVA,1
RDE,B
SVA,2
DCR,AUB
SVA,1Ø
SYR
Ø
UNION.DAT
```

Para os operadores de união, interseção e diferença, as relações devem ser UNION-COMPATIBLE

ARQUIVO : UNION.DAT

```
8Ø
DATAS
SYP,1,1
nome do arquivo de dados
SYC
FDR,@1
FDT
SVA,3
SVN,,1,2
FDR,@1,@3
FDT
JMP,4,Ø,1,78Ø1
SVA,4
SVN,,3,4
JMP,2,Ø,Ø
SVA,,,4
STR,@1Ø,@3,@4
FDN,%3,%4
JMP,2,Ø,2,78Ø4,78Ø1
JMP,7,1,Ø
FDN,%1,%2
JMP,2,Ø,1,78Ø4
JMP,15,1,Ø
JEQ,5,Ø,1Ø,1
FDR,@2
SVA,1
INC,1Ø,1
JMP,21,1,Ø
SYR
```

```
A INTERSECT B  ↔   RDE,A
                   SVA,1
                   RDE,B
                   SVA,2
                   DCR,AIB
                   SVA,1Ø
                   SYR
                   Ø
                   INTER.DAT
```

```
ARQUIVO : INTER.DAT
          8Ø
          DATAS
          SYP,1,1
          nome do arquivo de dados
          SYG
          DCC,SETA
          Y
          DCC,SETB
          Y
          DCC,SETAB
          Y
          DCC,SET12
          Y
          DCC,SET1
          Y
          DCC,SET2
          Y
          SET,9
          FDR,@ 1
          FDT
          SVA,3
          STE,@ 3,SETA
          FDN
          JMP,2,Ø,1,78Ø4
          JMP,4,1,Ø
          FDR,@ 2
          FDT
          SVA,3
          STE,@ 3,SETB
          FDN
          JMP,2,Ø,1,78Ø4
          JMP,4,1,Ø
          SOI,SETA,SETB,SETAB
          SET,8
          DLS,SET12
          INC,9,1
          LIE,SETAB,%9
          JMP,13,Ø,1,71Ø4
          SVA,3
          DLS,SET1
          FDR,@ 1,@ 3,,SET1
          DLS,SET2
          FDR,@ 2,@ 3,,SET2
          SOI,SET1,SET2,SET12
          INC,8,1
          LIE,SET12,%8
          JMP,13,1,2,71Ø4,71Ø3
          SVA,4
          STR,@ 1Ø,@ 3,@ 4
          JMP,5,1,Ø
          SYR
```

```
A MINUS B      ↔      RDE,A
                      SVA,1
                      RDE,B
                      SVA,2
                      DCR,AMB
                      SVA,1Ø
                      SYR
                      Ø
                      MINUS.DAT
```

```
ARQUIVO : MINUS.DAT
          8Ø
          DATAS
          SYP,1,1
          nome do arquivo de dados
          SYC
          DCC,SETA
          Y
          DCC,SETB
          Y
          DCC,SETAB
          Y
          DCC,SETAMB
          Y
          DCC,SET1
          Y
          DCC,SET2
          Y
          DCC,SET12
          Y
          B
          SOD,SETA,SETB,SETAB
          INC,1Ø,1
          LIE,SETAB,Z1Ø
          JMP,13,Ø,1,71Ø4
          SVA,3
          FDR,@1, @3
          FDT
          JMP,3,Ø,1,78Ø1
          SVA,4
          JMP,2,Ø,Ø
          SVA,,,4
          STR,@1Ø, @3, @4
          FDN
          JMP,12,1,2,78Ø4,78Ø1
          JMP,6,1,Ø
          SOI,SETA,SETB,SETAMB
          INC,9,1
          LIE,SETAMB,Z9
          JMP,12,Ø,1,71Ø6
          SVA,3
          FDR,@1, @3,,SET1
          FDR,@2, @3,,SET2
          SOD;SET1;SET2,SET12
          SET,8,1
          LIE,SET12,Z8
          JMP,9,1,2,71Ø4,71Ø3
          SVA,4
          STR,@1Ø,@3,@4
          INC,8,1
          JMP,5,1,Ø
          SYR
```

```
A TIMES B   ↔   RDE,A
                SVA,1
                RDE,B
                SVA,2
                DCR,ATB
                SVA,1Ø
                SYR
                Ø
                TIMES.DAT
```

ARQUIVO : TIMES.DAT

```
8Ø
DATAS
SYP,1,1
nome do arquivo de dados
SYG
DCC,SET1
Y
DCC,SET2
Y
FDR,@ 1
JMP,36,Ø,1,75Ø8
FDT
SVA,3
SVN,,1,2
FDR,@ 1,@ 3,,SET1
Y
SET,1Ø,1
LIE,SET1,%1Ø
JMP,25,Ø,1,71Ø4
SVA,4
STR,@ 1Ø,@ 3,@ 4
INC,1Ø,1
FDR,@ 2
JMP,23,Ø,1,75Ø8
FDT
SVA,5
SVN,,3,4
STR,@ 1Ø,@ 3,@ 5
STR,@ 1Ø,@ 4,@ 5
FDR,@ 2,@ 5,,SET2
Y
SET,9,1
LIE,SET2,%9
JMP,7,Ø,1,71Ø4
SVA,6
STR,@ 1Ø,@ 3,@ 6
STR,@ 1Ø,@ 4,@ 6
STR,@ 1Ø,@ 5,@ 6
INC,9,1
JAE,7,1,3
FDR,EX,%4
JMP,94,1,2,78ØA,7ØØ1
JMP,17,1,Ø
```

```
FDN,%1,%2
JMP,2,Ø,1,78Ø4,78Ø1
JMP,33,1,Ø
SYR
```

SELECT A WHERE P	↔	RDE,A
		SVA,1
		RDE,X
		SVA,2
		DCR,SELA
P é <ser igual a X>		SVA,1Ø
		SYR
		Ø
		SIGUAL.DAT

```
ARQUIVO : SIGUAL.DAT
          8Ø
          DATAS
          SYP,1,1
          nome do arquivo de dados
          SYG
          SET,10,1
          FDR,@1,@2
          JMP,13,Ø,1,75Ø4
          FDT
          JMP,3,Ø,1,78Ø1
          SVA,4
          JMP,2,Ø,Ø
          SVA,,,4
          JEQ,3,Ø,1Ø,1
          STR,@1Ø,@2,@4
          JMP,2,Ø,Ø
          STR,@1Ø,@4,@2
          FDN
          JMP,6Ø,2,78Ø4,78Ø1
          JMP,9,1,Ø
          FDR,@1,,@2
          JMP,3,Ø,1,75Ø4
          INC,1Ø,1
          JMP,15,1,Ø
          SYR
```

PROJECT A OVER{B, C} ↔

```
RDE,A
SVA,1
SET,1Ø,2 (quando C)
DCC,PRD
SVA,1Ø
SYR
Ø
PROJECT.DAT
```

ARQUIVO : PROJECT.DAT

```
8Ø
DATAS
SYP,1,1
nome do arquivo de dados
SYG
DCC,SET1
Y
FDR,@ 1
FDT
SVA,3
SVN,,1,2
JEQ,5,Ø,1Ø,2
STE,@ 3,@ 1Ø
FDN
JMP,1Ø,Ø,1,78Ø4
JMP,6,1,Ø
FDR,@ 1,@ 3,,SET1
Y
SOU,SET1,@ 1Ø,@ 1Ø
FDN,%1,%2
JMP,4,Ø,1,78Ø4
SVA,3
SVN,,1,2
JMP,7,1,Ø
LIE,@ 1Ø
SYR
```

EQUI-JOIN

```
JOIN A AND B OVER X   ←→   RDE,A
                           SVA,1
                           RDE,B
                           SVA,2
                           SET,1Ø,1 p/ join sobre as células C de A
                           SET,9,1  p/ join sobre as células C de B
                           DCR, JOINAB ///
                           SVA,1Ø
                           SYR
                           Ø
                           JOINAB.DAT
```

ARQUIVO : JOINAB.DAT

```
8Ø
DATAS
SYP,1,1
nome do arquivo de dados
SYG
DCC,SET1
Y
DCC,SET2
Y
DCC,SET12
Y
DCC,SETA
Y
DCC,SETB
Y
SET,8
FDR,@ 1
FDT
SVA,3
SVN,,1,2
JEQ,CELULACA,Ø,1Ø,1
STE,@ 3,SET1
FDN,Z1,Z2
JMP,2,Ø,1,78Ø4
JMP,6,1,Ø
FDR,@ 2
FDT
SVA,3
SVN,,1,2
JEQ,CELULABA,Ø,9,1
STE,@ 3,SET2
FDN,Z1,Z2
JMP,2,Ø,1,78Ø4
JMP,6,1,Ø
SOI,SET1,SET2,SET12
Y
SET,7
INC,8,1
LIE,SET12,Z8
JMP,FIM,Ø,1,71Ø4
SVA,3
JEQ,3,Ø,1Ø,1
FDR,@ 1,@ 3,,SETA
Y
JMP,2,Ø,Ø
FDR,@ 1,,@ 3,SETA
Y
SET,6
INC,7,1
LIE,SETA,Z7
JMP,14,1,1,71Ø4
SVA,4
STR,@ 1Ø,@ 3,@ 4
JEQ,3,Ø,9,1
FDR,@ 2,@ 3,,SETB
Y
JMP,2,Ø,Ø
FDR,@ 2,,@ 3,SETB
Y
INC,6,1
LIE,SETB,Z6
JMP,14,1,1,71Ø4
SVA,5
STR,@ 1Ø,@ 3,@ 5
STR,@ 1Ø,@ 4,@ 5
JMP,6,1,Ø
LBL,CELULACA
FDR,@ 1,@ 3
FDT
JMP,3,Ø,1,78Ø1
SVA,4
JMP,2,Ø,Ø
SVA,,,4
STE,@ 4,SET1
FDN
JMP,53,1,2,78Ø4,78Ø1
JMP,6,1,Ø
LBL,CELULABA
FDR,@ 2,@ 3
FDT
JMP,3,Ø,1,78Ø1
SVA,4
JMP,2,Ø,Ø
SVA,,,4
STE,@ 4,SET2
FDN
JMP,,Ø,2,78Ø4,78Ø1
JMP,6,1,Ø
LBL,FIM
SYR
```

```
DIVIDE A BY B   ↔   RDE,A
                   SVA,1
                   RDE,B
                   SVA,2
                   DCC,DIVIDE
                   SVA,1Ø
                   SYR
                   Ø
                   DIVIDE.DAT
```

```
ARQUIVO : DIVIDE.DAT
          8Ø
          DATAS
          SYP,1,1
          nome do arquivo de dados
          SYG
          DCC,SET1
          Y
          DCC,SET
          Y
          FDR,@ 1
          FDT
          SVA,3
          SVN,,1,2
          FDR,@ 1,@ 3,,SET1
          Y
          SOD,@ 2,SET1,SET
          Y
          LIE,SET
          JMP,2,Ø,1,71Ø3
          JMP,2,Ø,Ø
          STE,@ 3,@ 1Ø
          FDN,%1,%2
          JMP,2,Ø,1,78Ø4
          JMP,12,1,Ø
          LIE,@ 1Ø
          SYR
```