SÉRIE McGRAW-HILL/DATALÓGICA

dBASE II

PARA PRINCIPIANTES

SEGUNDA EDIÇÃO REVISADA

ALAN FREEDMAN

McGraw-Hill

DATALOGICA

dBASE II
PARA
PRINCIPIANTES

dBASE II
PARA PRINCIPIANTES

2ª edição revisada

Alan Freedman

Tradução e Revisão Técnica:
Equipe Datalógica

Revisão Técnica da 2ª edição
José Cláudio Boccia
Analista Consultor de Sistemas

McGraw-Hill
São Paulo
Rua Tabapuã, 1.105, Itaim-Bibi
CEP 04533
(011) 881-8604 e (011) 881-8528

Rio de Janeiro • Lisboa • Porto • Bogotá • Buenos Aires • Guatemala • Madrid • México • New York • Panamá • San Juan • Santiago

Auckland • Hamburg • Kuala Lumpur • London • Milan • Montreal • New Delhi • Paris • Singapore • Sydney • Tokyo • Toronto

Do original:

dBASE II for the First-Time User

Editor: Milton Mira de Assumpção Filho
Coordenadora de Revisão: Daisy Pereira Daniel
Capa-Criação: Datalógica/McGraw-Hill
Arte: Ciro Giordano

Dados de Catalogação na Publicação (CIP) Internacional
(Câmara Brasileira do Livro, SP, Brasil)

F93d 2. ed.
Freedman, Alan.
dBASE II para principiantes / por Alan Freedman ; tradução e revisão técnica Equipe Datalógica ; revisão técnica da 2a. edição José Cláudio Boccia. – 2. ed. rev. – São Paulo : McGraw-Hill, 1987.
(Série McGraw-Hill-Datalógica)

1. dBase II (Programa de computador) I. Título. II. Série.

86-2273 CDD-001.6425

Índices para catálogo sistemático:
1. dBASE II : Computadores : Programas : Processamento de dados 001.6425

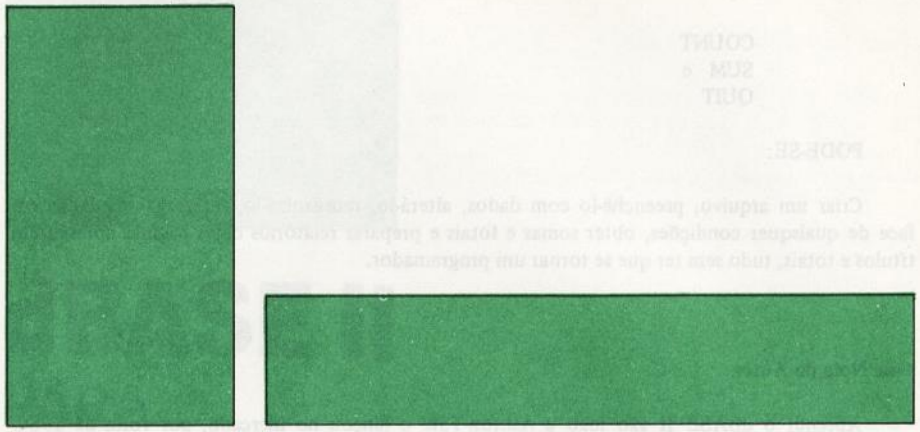
Para Doc – pelo Espaço e Tempo

AGRADECIMENTOS ESPECIAIS À EQUIPE DATALÓGICA

Ademir Avila
André Noschese
Denise T. de Souza
Eduardo José S. Engelmann
Eduardo P. da Conceição
Eliana De S. Figueiredo
José Rubens S. Toledo
Marcio Arruda Cunha
Marcos B. Prestes Cesar
Maria Raquel Luperi

SUMÁRIO

APRESENTAÇÃO



Utilizando-se o dBASE II, os dados são armazenados em um ARQUIVO dBASE ...

- INVENTÁRIO
- CHEQUES
- RELAÇÃO DE ENDEREÇOS
- ORDENS
- CLIENTES
- PESSOAL

Depois, usando a linguagem d BASE II, obtém-se respostas para perguntas do tipo:

Por que preço o produto X foi vendido na região Y?
Qual o valor total devido?
Temos programadores que falem espanhol?
Quais as ações e bônus que estão rendendo mais?
Há casas à venda que disponham de piscina?

ATRAVÉS DOS DOZE COMANDOS SEGUINTES:

CREATE
USE
APPEND
DELETE
EDIT
SORT
DISPLAY
BROWSE
REPORT

COUNT
SUM e
QUIT

PODE-SE:

Criar um arquivo, preenchê-lo com dados, alterá-lo, reorganizá-lo, interrogar os dados em face de quaisquer condições, obter somas e totais e preparar relatórios cujas páginas apresentem títulos e totais, tudo sem ter que se tornar um programador.

Uma Nota do Autor

Aprendi o dBASE II tão logo a Ashton-Tate o lançou no mercado, por volta de 1980. Fiquei muito entusiasmado porque, pela primeira vez, seria capaz de demonstrar, nos seminários de aperfeiçoamento sobre conhecimentos de computador que eu ministrava, os conceitos essenciais de processamento de dados utilizando um microcomputador.

Desde então tenho usado, com sucesso, o dBASE em cursos de treinamento e em meu próprio trabalho. Embora tenha examinado muitos sistemas de gerenciamento de dados surgidos após o dBASE II, ainda acredito ser este o produto mais inteligente no mercado. Enquanto os software de outras procedências disfarçam os conceitos tradicionais de processamento de dados, tentando torná-los mais fáceis de serem aprendidos, o dBASE II usa uma linguagem lógica, simples e direta para transmitir as funções essenciais de manipulação de dados. Através do dBASE II, os principiantes podem aprender a manipular seus dados ao mesmo tempo em que se tornam "peritos" em computadores.

Também o dBASE II permite introduzir, de modo mais inteligente, o ensino das linguagens de computador em nosso sistema escolar, pois concentra-se no banco de dados que constitui o fundamento de um sistema de informações. O processo comum de se abordar esse estudo, atualmente, consiste em ensinar a programar desde o início, sem, contudo, se explicar por que os programas são necessários. A verdade é que, uma vez que o principiante compreenda o conceito de um banco de dados, o objetivo de se escrever o programa e os problemas aí envolvidos, tudo passa a fazer mais sentido.

Este livro é destinado aos principiantes tanto em dBASE II, quanto no uso de um computador. Mesmo que jamais tenha tocado em um computador, você poderá tornar-se mestre de seus dados através do dBASE II. Tente... você se surpreenderá!

Boa sorte,

Alan Freedman

Como Usar este Livro

É neste livro que se aprenderá a lidar com o dBASE II. Esta página deve ser cuidadosamente estudada antes de se prosseguir com a leitura.

Representações da Tela

No Capítulo 4 (e em outros) vai-se criar e manipular um arquivo denominado "PESSOAL". Idealizaram representações para a tela, como a de fundo preto indicada abaixo, semelhantes às que aparecem na tela de um computador.

```
. USE PESSOAL
. DISPLAY NOEM
 ***ERRO DE SINTAXE***
          ?
DISPLAY NOEM
CORRIGIR E ENTRAR DE NOVO (Y/N)? Y
MUDAR DE:   EM
MUDAR PARA: ME
DISPLAY NOME
MAIS CORRECOES (Y/N)?
```

O texto em branco é o que se digita.
O texto em verde é o apresentado pelo dBASE.

Em outras partes do livro há representações da tela que aparecerão sob a forma mostrada abaixo. Elas ilustram comandos e procedimentos que não se baseiam no arquivo "PESSOAL" a ser criado. São apenas exemplos e não se destinam a uso interativo com este livro.

```
MODIFY STRUCTURE TRABALHO
MODIFY? (Y/N)
```

O texto em preto é o que se digita.
O texto em verde é o apresentado pelo dBASE.

Notas em Itálico

Ao longo do livro surgirão observações *em itálico*. São notas sobre opções ou detalhes com os quais se deve ter cuidado especial.

Por vezes torna-se extremamente importante fazer ou não fazer algo em especial. Isto é indicado pelas observações *em itálico*, em geral intitulada "Atenção".

Tecla Return

A tecla RETURN ou ENTER aparece na tela como <RETURN>. Significa que se deve pressionar <RETURN> uma vez.

PARTE 1

INTRODUÇÃO

CAPÍTULO 1

O QUE SÃO SISTEMAS DE GERENCIAMENTO DE DADOS?

Um sistema de Gerenciamento de Dados é um software que proporciona um modo de se criar um sistema completo de GUARDA DE REGISTROS sobre qualquer assunto. Permite também, a atualização dos registros e a obtenção de qualquer tipo de informação desejada relativa aos dados nele contidos. Os sistemas automatizados de guarda de registros, chamados SISTEMAS DE INFORMAÇÃO, têm sido o sustentáculo das aplicações de processamento de dados nos últimos trinta anos.

O centro de um sistema de informação é o seu BANCO DE DADOS, no qual os dados necessários ao assunto são armazenados. Em uma atividade comercial, alguns dos assuntos típicos são: empregados, produtos, clientes, pedidos, vendedores e compras. No lar, os bancos de dados podem ser usados em assuntos como: coleções de selos, moedas, livros ou discos, receitas culinárias, relação de artigos domésticos e investimentos financeiros.

Independentemente da natureza dos dados, uma vez projetado e criado o banco de dados, há um número de funções de PROCESSAMENTO necessárias à operação adequada de um sistema de informações. Para isso, deve-se ser capaz de introduzir os dados no banco de dados, alterá-lo, reorganizá-lo, fazer perguntas e preparar relatórios. Outras funções de manutenção como, por exemplo, excluir registros e realizar cópias auxiliares, também são necessárias.

O esquema seguinte mostra como os Sistemas de Gerenciamento de Dados enquadram-se hoje no mundo dos software disponíveis para microcomputadores.

O dBASE II é um sistema de gerenciamento de banco de dados (SGBD)

O Mundo do Software:

Sistemas de Gerenciamento de Dados

Permitem a criação e gerenciamento de arquivos eletrônicos de dados.

Usados para manter os registros atualizados e para se obter uma rápida recuperação e análise dos dados neles contidos. Este é um "Processamento Tradicional de Dados".

Programas de Gráficos para Atividades Comerciais

Convertem números em ilustrações, tais como diagramas de barras, GRÁFICOS CIRCULARES SETORIZADOS e anexos.

Processadores de Palavras

Permitem a criação e gerenciamento de arquivos eletrônicos de palavras (texto).

Usados para substituir qualquer função de datilografia, também para guardar registros de "texto"

Programas de Comunicação

Transmitem e recebem quaisquer dados eletrônicos entre o computador usado e outro computador ou terminal.

Pacotes de Software

Os pacotes ("enlatados") disponíveis atendem a todas as necessidades de processamento de dados comuns, do tipo folhas de pagamento, faturamentos e estoques e aplicações industriais específicas como: seguros, negócios bancários e manufaturas.

Atendendo às necessidades em questão, os pacotes software constituem o modo mais rápido de se implementarem os sistemas de informação. Porém, caso as necessidades sejam modificadas – como em geral acontece – , os pacotes não poderão ser alterados.

Planilhas

Permitem a criação e o gerenciamento de arquivos eletrônicos de dados numéricos para cálculos.

Usados para orçamentos, demonstrativos e planejamento em questões do tipo "e se?"

Programas Educacionais

Possibilitam o aprendizado e a prática de uma gama de assuntos, desde a introdução ao aprendizado das línguas estrangeiras até a eletrônica. Em geral conhecidos por "courseware", estes programas conduzem a um nível de aprendizado de assuntos complicados superiores aos anteriormente verificados.

Sistemas de Planejamento Financeiro

Efetuam análises estatísticas baseadas em dados financeiros. Podem calcular também valores ótimos para uma determinada meta. Eles executam muitas funções de planejamento automático que não podem ser realizadas pelas Planilhas.

Jogos

Proporcionam uma gama de entretenimentos, variando de "Invasores Espaciais", ao xadrez. Os jogos que requerem rapidez de movimentos necessitam, em geral, de dispositivos adicionais de entrada, tais como botões disparadores e transmissores de sinais ao computador e que possibilitam o controle dos movimentos de uma imagem na tela.

Programas de Gerenciamento de Projetos

Permitem o acompanhamento de projetos e determinam o impacto das alterações. Indicam o "Caminho Crítico" do projeto, que representa o conjunto de tarefas que não comportam atrasos em suas realizações.

Programas para Desenvolvimento de Projetos Assistidos por Computador

Fornecem um método eletrônico para a criação de projetos, partindo-se de desenhos feitos à mão ou de esquemas complexos de muita precisão. Os programas "PAC" também podem empregar dispositivos gráficos de entrada tais como "canetas ópticas" e as telas especiais que permitem o seu uso.

Sistemas Operacionais

Controlam o sistema do computador. São os programas principais de controle que ligam um programa ao computador que está sendo usado. Todos os programas devem ser escritos de forma a se comunicarem com um sistema operacional.

Linguagem de Programação

Os programas tradutores, chamados compiladores, interpretadores e montadores, traduzem as declarações da linguagem usada na programação, feitas em inglês, para uma linguagem de máquina que o computador possa entender. Todos os software descritos nestas páginas (inclusive as próprias linguagens de programação) foram escritos em uma linguagem

de programação. Programar o computador significa dizer-lhe exatamente o que fazer e, também, entender como colocar as instruções em uma seqüência lógica, adequada à resolução do problema.

A Estrutura dos Dados Eletrônicos e as Funções de Processamento Necessárias

I. Os Dados

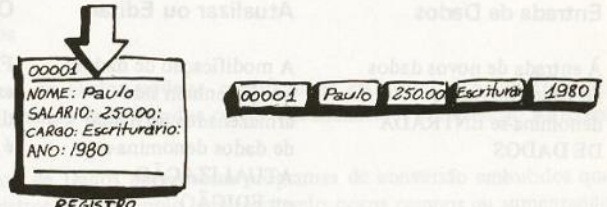
Dados são unidades de informação definidas de forma precisa. São exemplos de unidades de dados: nome, endereço, cidade, estado, total devido, total pago, descrição e número de telefone. Ao contrário dos arquivos-texto dos processadores de palavras, onde o formato é sempre o da estrutura comum de páginas e palavras, as estruturas dos registros de processamento de dados variam de acordo com as necessidades do usuário. Os dados devem ser definidos antes que qualquer processamento ocorra.

Quando em um sistema de computador se armazenam eletronicamente dados relativos a um assunto, eles ficam guardados em um REGISTRO. As diferenças entre as formas de armazenamento manual e eletrônico são as seguintes:

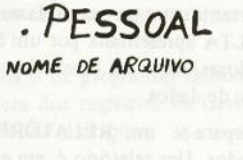
Sistema de Papel | Sistema Eletrônico

As categorias de informação transformam-se em nomes de campos:

Os registros, nas pastas de arquivo, transformam-se em registros em um arquivo de computador.



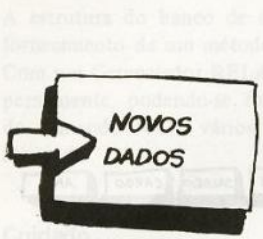
A pasta de arquivo transforma-se em nome de arquivo.



A gaveta de arquivo transforma-se em um banco de dados.

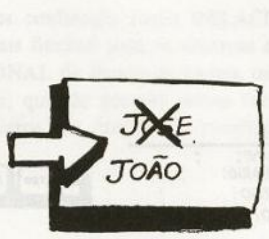
II. O Processamento

Qualquer ação que se realize com os dados recebe o nome de PROCESSAMENTO. As funções de processamento necessárias a um sistema de informações são as seguintes:



Entrada de Dados

À entrada de novos dados no banco de dados denomina-se ENTRADA DE DADOS



Atualizar ou Editar

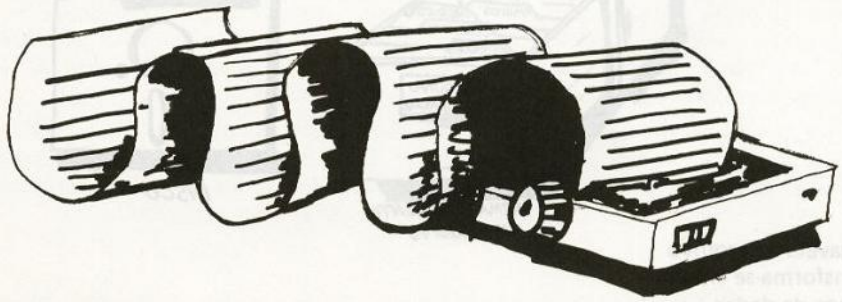
A modificação de dados que já tenham sido armazenados no banco de dados denomina-se ATUALIZAÇÃO ou EDIÇÃO.

Ordenar

Freqüentemente torna-se necessário examinar os dados em seqüência diferente. A reordenação de dados é chamada ORDENAÇÃO.

É importante que se possa fazer qualquer tipo de pergunta sobre os dados. A capacidade de CONSULTA apresentada por um Sistema de Gerenciamento de Dados é uma de suas características mais valiosas. Deve ser possível perguntar-se qualquer coisa sobre as condições dos dados contidos no banco de dados.

Prepara-se um RELATÓRIO quando se imprimem, em papel, os dados gerados pelo computador. Um relatório é, em geral, mais formal e elaborado do que uma consulta. Os relatórios contêm títulos e números de páginas, totais e subtotais etc. Algumas vezes, os Sistemas de Gerenciamento de Dados combinam as capacidades de consultar e de emitir relatórios em uma única função.



Classes dos Sistemas de Gerenciamento de Dados

Há duas classes de Sistemas de Gerenciamento de Dados, identificáveis por sua maneira de tratar os dados. Essas duas classes são chamadas Gerenciadores de Arquivo e Gerenciadores de Banco de Dados.

Gerenciadores de Arquivo

Com os Gerenciadores de Arquivo ocorre que, uma vez projetado um registro e aí armazenados os dados, sua leitura não pode ser facilmente alterada. Quando se precisa atender a outras finalidades e se quer armazenar tipos diferentes de dados no arquivo, deve-se, em geral, elaborar novos programas para converter os dados dos arquivos existentes, ou então começar tudo de novo, a partir do início.

Gerenciadores de Banco de Dados

Com um Gerenciador de Banco de Dados pode-se modificar as estruturas dos registros nos arquivos de dados existentes, à medida que os dados ou as necessidades de informação sofrerem variações*.

Os Gerenciadores de Banco de Dados apresentam programas de conversão embutidos que modificam as estruturas dos registros. Por exemplo, acrescentando novos campos ou aumentando os campos existentes. No novo arquivo, os dados do arquivo antigo são automaticamente convertidos sem que haja necessidade de se desenvolverem novos programas para realizar a conversão.

Além disso, freqüentemente os procedimentos adotados e os programas elaborados para os dados antigos não são afetados pelas alterações na estrutura dos registros. Os Gerenciadores de Banco de Dados permitem uma resposta rápida às necessidades de alterações.

O Gerenciador Relacional de Banco de Dados

Os Gerenciadores Simples de Banco de Dados evoluíram para Sistemas Completos de Gerenciamento de Dados, tal como o *dBASE II*. A princípio, eles eram usados conjuntamente com linguagens convencionais de programação, como COBOL, de forma a facilitar o gerenciamento dos dados. Visando garantir o acesso aos dados, os primeiros Gerenciadores exigiam que todos os dados nesses bancos apresentassem certo grau de relacionamento.

Fixadas estas relações, para se recuperar certos dados, deverão ser "percorridos" todos eles, a partir dos considerados como a origem, até se conseguir os desejados. Isto significa obter respostas para perguntas *ad hoc* e preparar novos tipos de relatórios o que, em geral, implica o desenvolvimento de métodos sofisticados, ou a elaboração de novos programas.

---

* O Capítulo 6, "Planejando e Usando o Banco de Dados", discute os conceitos de dados *versus* informação.

Embora os dados não sejam fixos, devem ser corretamente planejados.
Ver Capítulo 6.

A estrutura do banco de dados conhecida como RELACIONAL foi desenvolvida visando ao fornecimento de um método mais flexível para se obterem as combinações de dados necessárias. Com um Gerenciador RELACIONAL de Banco de Dados, os dados não são mais ligados de forma permanente, podendo-se formar, quando preciso, novas versões para eles. Pode-se unir, através do comando JOIN, vários registros em funções de condições diferentes daquelas do início da atividade.

Cuidado ...

Nem todos os Sistemas de Gerenciamento de Dados são iguais. O termo tanto pode ser usado para descrever um sistema que gerencie uma simples lista de nomes e endereços, como um número fixo de funções, ou mesmo um sistema completo destinado a vários usuários, capaz de prover todas as manipulações de dados e de análises possíveis.

CAPÍTULO 2

AS CARACTERÍSTICAS E CAPACIDADES DO dBASE II

O *dBASE II* é um Sistema Relacional de Gerenciamento de Banco de Dados destinado tanto ao principiante quanto ao programador experiente. O dBASE II é processado na maioria dos computadores que usam os sistemas operacionais CP/M® ou MS-DOS™.

O dBASE é Construído em Dois Níveis

ATENÇÃO:
Este livro concentra-se no NÍVEL 1, que pode ser tudo o que se necessita para gerenciar e controlar os próprios bancos de dados.

O *NÍVEL 1* proporciona comandos poderosos que podem ser introduzidos no computador sob a forma de conversação e que permite operações do tipo: criação de banco de dados, entrada de dados, atualização, consulta, ordenação e emissão de relatórios, independentemente da experiência técnica do usuário. No NÍVEL 1 apenas se informa ao dBASE *o que* ele deve fazer, e ele imagina *como* executar a tarefa.

O *NÍVEL 2* é um conjunto completo de comandos de linguagem de programação que permite a elaboração de programas (listas de comandos) capazes de executar uma variedade infinita de tarefas de manipulação de dados. No NÍVEL 2 deve-se informar ao dBASE o *que* fazer e *como* a tarefa deverá ser feita passo a passo.

Quando os sistemas de informação se destinam ao uso de muitas pessoas no âmbito de uma organização, certos controles deverão ser projetados em função do sistema. Os controles seguintes requerem programação no NÍVEL 2:

1. Validação da Entrada de Dados: o programa de entrada de dados deve ser capaz de analisar entradas potencialmente errôneas. Os únicos testes de entrada de dados efetuados pelo dBASE no NÍVEL 1 são: a) verificação dos números quando os dados são definidos como Numéricos e b) verificação de T, F, Y ou N quando os dados são definidos como Lógicos (True/False, Yes/No – Verdadeiro/Falso, Sim/Não). As validações mais elaboradas requerem um programa escrito na linguagem de programação do dBASE (NÍVEL 2).

2. Pistas de Auditoria: os sistemas de informação ligados a negócios exigem que todas as alterações sejam registradas como transações separadas. Estas transações fornecem um modo de se voltar à condição anterior dos dados e, também, de indicar quando e como aquelas alterações foram realizadas. O método interativo (NÍVEL 1) não registra as alterações automaticamente. Deve-se usar a programação no NÍVEL 2 para gerar, de forma adequada, a pista de auditoria.

3. Integridade de Processamento: embora o dBASE possa ser usado como uma calculadora de mesa, nem sempre é uma boa idéia atualizarem-se os dados valendo-se de longos cálculos efetuados pelo próprio usuário. Um programa dBASE pode executar cálculos complexos e repetitivos envolvendo esses dados, com a garantia de processamento preciso.

Um Software como o dBASE II é constantemente ampliado pelo distribuidor. Os números da versão indicam a qual realese *do produto se está fazendo referência. A versão 2.4 foi liberada pela Ashton-Tate em 1983.*

Além disso, a programação no NÍVEL 2 também é necessária no preparo da impressão de saídas de dados sob formas previamente escolhidas. Contudo, a partir da versão 2.4 do dBASE II, o programa ZIP fornece um método alternativo de se lidar com estes formatos, sem implicar qualquer programação. Ver o manual de documentação do ZIP.

Sendo um novato em computador, mas querendo continuar com o aprendizado da programação, deve-se começar com este livro e ir em frente na programação dBASE, após todos os comandos do NÍVEL 1 terem sido dominados.

O dBASE Lida com Dados e Palavras

Os registros de dados compõem a estrutura tradicional de processamento de dados chamada *arquivo de dados*. Os registros de dados devem ser definidos de modo preciso antes que qualquer processamento ocorra. Os sistemas de processamento de dados são conhecidos por sua rápida capacidade de recuperação de informações.

As palavras, que formam o *arquivo-texto* tradicional, não possuem lugares predeterminados na estrutura do arquivo. A busca de uma palavra específica é feita examinando o texto do princípio ao fim. Embora de um modo tipicamente mais lento que o da recuperação no processamento de dados, as palavras podem ser localizadas, não importando onde se encontrem no arquivo.

O dBASE II trabalha igualmente bem com textos e dados, proporcionando a capacidade rara de se poder formular uma pergunta relacionada, ao mesmo tempo, com dados e textos. Este aspecto permite uma tremenda flexibilidade para se projetar o banco de dados. Com o tempo, esta será uma característica obrigatória em todos os Sistemas de Gerenciamento de Dados. (Ver Capítulo 6.)

Uma Introdução Lógica ao Aprendizado do Processamento de Dados e da Programação

A característica de destaque do dBASE é sua linguagem. O NÍVEL 1 proporciona uma maneira fácil e prática de se aprender os conceitos de processamento de dados, já que seus comandos abrangem todas as funções vitais para isso. A linguagem NÍVEL 1 é interativa: ordena-se que o dBASE desempenhe uma função e ele executa a ordem.

Por outro lado, o NÍVEL 2 é usado para se escrever programas de construção similares aos escritos em PASCAL. O NÍVEL 2 requer que se entenda como a lógica do programa deve ser construída a fim de se resolver o problema. No NÍVEL 1, toda aquela lógica já é feita pelo dBASE. O NÍVEL 2 exige mais tempo e dedicação para ser dominado. Porém, o aspecto mais encorajador do dBASE é que, aprendido o NÍVEL 1, o caminho para o NÍVEL 2 torna-se cada vez mais fácil. Isto porque muitos dos comandos do NÍVEL 1 são os mesmos do NÍVEL 2. Além do mais, a experiência adquirida com o NÍVEL 1 contribui para que os conceitos de programação comecem a fazer muito mais sentido.

Assim, o sistema e a linguagem dBASE II proporcionam uma introdução muito mais lógica ao aprendizado da programação e do processamento de dados – talvez a introdução mais lógica surgida até agora.

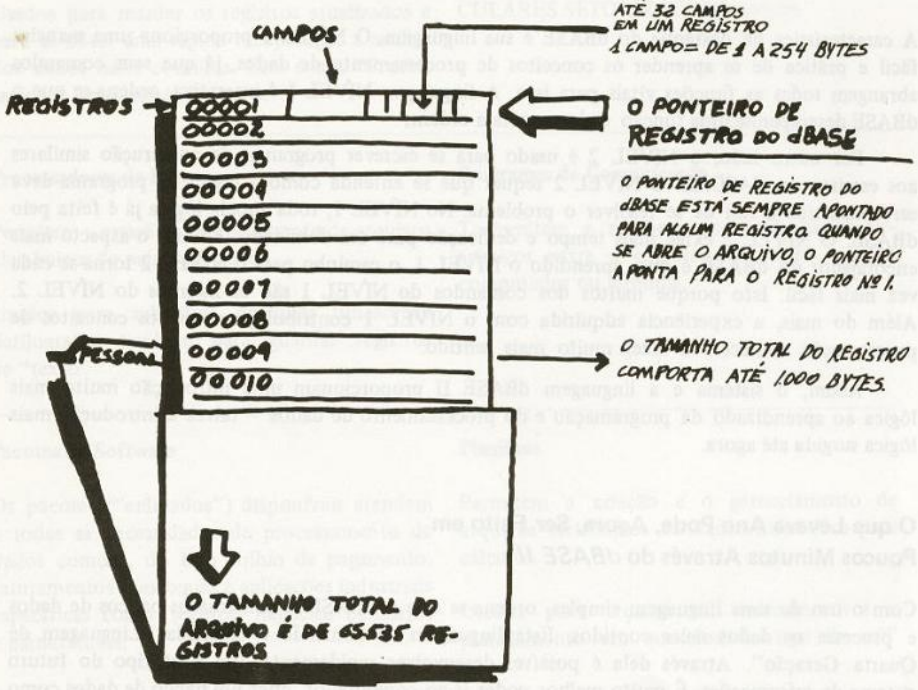
O que Levava Ano Pode, Agora, Ser Feito em Poucos Minutos Através do *dBASE II*

Com o uso de uma linguagem simples, ordena-se que o dBASE II construa os bancos de dados e processe os dados neles contidos. Esta linguagem de comando é chamada "Linguagem de Quarta Geração". Através dela é possível desenvolver rapidamente, um protótipo do futuro sistema de informações. É muito melhor poder ir ao computador, criar um banco de dados como exemplo e, em poucos minutos, dar entrada aos dados, do que ter de passar pelo suplício verbal de tentar explicar aos programadores ou distribuidores o que se espera auferir do sistema. Assim, na mesma hora, em vez de semanas ou meses depois, pode-se fazer vários tipos de perguntas e gerar quaisquer relatórios necessários ao sistema. À medida que os resultados forem aparecendo na tela ou no papel, eles fornecerão subsídios capazes de permitir a descoberta

de outras informações úteis ao sistema. Se alguns dos dados não estiverem no banco de dados, é possível modificá-lo rapidamente e entrar com os dados esquecidos. A seguir, pode-se prosseguir com as perguntas, até que todas as possibilidades se esgotem. Ao contrário dos métodos tradicionais, dispor-se-á, no final, de um conjunto operante de especificações que refletirão, verdadeiramente, as necessidades em questão.

Este processo recebe o nome de "prototipagem" e é o exercício mais proveitoso que se pode realizar, mesmo que o sistema final não seja escrito em dBASE II. Mas se o sistema final for o dBASE II, ter-se-á dado um passo à frente, pois a linguagem de interrogação dBASE II já terá sido aprendida.

Sumário das Capacidades Máximas de Armazenamentos de um Arquivo dBASE



PARTE 2

VAMOS EXPERIMENTÁ-LO

CAPÍTULO 3

INSTALANDO O dBASE II

O dBASE II Recém-Saído da Caixa

O disco recebido ao se adquirir o dBASE II contém todos os arquivos de programas necessários para colocar o dBASE em funcionamento no computador. Este disco é denominado "Disco de Distribuição" ou "Disco Original".

O Disco de Distribuição é usado uma única vez para se preparar uma "Cópia de Reserva". A partir daí, a cópia de reserva servirá para criar as "Cópias de Trabalho", a serem utilizadas nas atividades diárias.

Seguir o diagrama abaixo a fim de se criar uma cópia de trabalho do dBASE II.

1. Formatar os Discos em Branco

Usar o programa FORMAT do disco do sistema operacional para a criação de três discos novos e formatados. Copiar, também, o sistema operacional em dois desses discos. O usuário deverá adotar este procedimento em seu sistema operacional específico (caso este seja CP/M, confirmar a inclusão de PIP.COM.). Tais discos serão destinados a:

- Cópia formatada de reserva para o dBASE II:

Este disco contém o Sistema Operacional bem como os programas dBASE II adequados à tela do computador a ser utilizado.

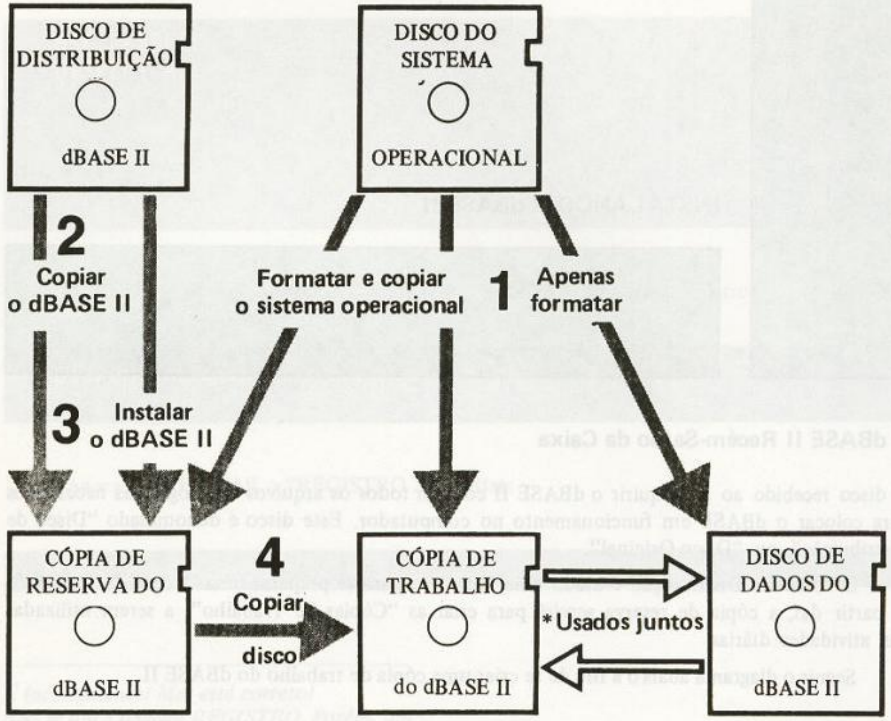
- Cópia formatada de trabalho para o dBASE II:

Este disco é uma cópia exata do Disco de Reserva do dBASE II (com o sistema operacional).

- Disco formatado de dados:

Este disco será usado para armazenar os dados a serem posteriormente introduzidos no dBASE.

Não deve conter nem o sistema operacional específico nem os programas dBASE II.



Se o dBASE já estiver pronto e em operação no computador, passar para o Capítulo 4

2. Preparar uma Cópia Reserva do dBASE II

Copiar, do Disco de Distribuição para o Disco Formatado de Reserva, os seguintes programas:

DBASE.COM	→	(O programa básico do dBASE II)
DBASEOVR.COM	→	(Disposição das superposições das telas em dBASE)
DBASEMSG.TXT	→	(Um arquivo de mensagens do HELP)
STARTUP.CMD	→	(Um programa para testar a operação adequada da tela) (Para o sistema IBM e outros de 16 bits este programa será denominado STARTUP.PRG)
INSTALL.COM	→	(Um programa para dar entrada aos parâmetros que compatibilizarão o dBASE II com a tela do computador a ser usado)

3. Instalar o dBASE II

O dBASE II deverá ser "personalizado" a fim de atender à operação específica da tela do vídeo do computador em uso. Executar os seguintes passos obedecendo à mesma ordem, para efetuar uma instalação compatibilizadora.

Executar o "Testando o dBASE II para a Operação Adequada da Tela", como segue abaixo. Se o teste for negativo, seguir o procedimento "Instalando o dBASE II", da página 21.

4. Preparar uma Cópia de Trabalho do dBASE II

Copiar o Disco de Reserva, inteiro, em um dos novos discos formatados que passa a ser o "Disco de Trabalho". É uma réplica exata do Disco de Reserva. Neste livro, considerar os dados armazenados no Disco de Trabalho. Posteriormente, este servirá para guardar os programas dBASE II, enquanto o disco formatado de dados será utilizado para armazenar os dados. Sempre que um novo Disco de Trabalho for necessário, preparar a cópia a partir do Disco de Reserva.

Testando o dBASE II para Operação Adequada da Tela

O vídeo das telas de computador emprega comandos diferentes para apresentar os dados. Visando garantir a exposição correta do dBASE II na tela do computador, passou a ser fornecido, a partir da versão 2.4, um programa especial chamado "STARTUP". Ele tem por finalidade testar se o dBASE II já foi compatibilizado com o monitor de vídeo específico do computador. Caso não tenha sido, ou se a versão de que se dispõe for mais antiga, o próprio usuário deverá executar o programa "INSTALL", a fim de realizar a compatibilização necessária, antes de poder usar o software dBASE II.

Procedimento para a Execução do Programa "STARTUP"

1. Colocar o disco intitulado "Disco de Reserva" no drive A (o sistema operacional e os programas dBASE II mencionados anteriormente já deverão ter sido copiados neste disco).

2. Carregar o sistema. Isto significa carregar, na memória, o sistema operacional. Esta operação é realizada pressionando-se uma determinada tecla, ou uma combinação delas, no teclado do computador. O próprio computador poderá também executá-la, automaticamente, ao ser ligado. Consultar os manuais referentes ao computador e/ou sistema operacional.

Carregar o dBASE na memória do computador, entrando com o dBASE depois do sinal A > que aparece na tela.

```
A > DBASE <RETURN>
```

O dBASE irá para a memória, identificando-se na tela, e perguntará pela data de hoje.

Em alguns computadores, como o IBM-PC, a entrada da data é feita automaticamente. Esta mensagem não será exibida.

```
DIGITE A DATA DE HOJE OU RETURN
 (DD/MM/AA):
```

Pressionar < RETURN > para saltar a data e o dBASE responderá com uma mensagem e um ponto situado no extremo esquerdo da tela.

```
Copyright (C) 1982 RSP Inc.
***dBASE II VER  2.4 26 Mar 1983
."Digite 'HELP', 'HELP dBASE', ou um comando"
```

Entrar com "DO STARTUP" depois do ponto, exatamente como o indicado:

```
.DO STARTUP <RETURN>
```

Obter-se-á um menu para seleção. Entrar com o numeral 1 (seguido de < RETURN >) para escolher a solução "RUN THIS TEST PROGRAM" ("EXECUTAR ESTE PROGRAMA DE TESTE") do STARTUP.

```
1 - EXECUCAO DESTE PROGRAMA TESTE
SELECIONE E PRESSIONE RETURN: 1 <RETURN>
```

Os quatro testes seguintes serão executados. O STARTUP solicitará a confirmação dos resultados que aparecerem na tela:

1. **TESTE PARA CLEAR AND HOME SCREEN**
   (TESTE PARA LIMPEZA E POSICIONAMENTO DA TELA)

   O CLEAR verifica se todos os caracteres serão apagados do console. O HOME testa se o cursor irá localizar-se, de novo, no canto superior esquerdo da tela do console (na realidade, o cursor se reposicionará no final da linha de confirmação, na parte superior da tela). Como resposta, entrar com um "Y" (S) ou um "N" (N) seguido de < RETURN >.

2. **TESTE PARA BACKSPACE**
   (TESTE PARA RETROCESSO)

   Verifica a operação adequada da tecla BACKSPACE.

3. **TESTE PARA O POSICIONAMENTO DIRETO DO CURSOR**

   Visa garantir que o cursor ocupe a posição correta na tela.

4. **TESTE PARA CARACTERÍSTICAS ESPECIAIS DE VÍDEO**

   Testa as características de INTENSIDADE REDUZIDA ou de REVERSÃO DO VÍDEO.

Se todos os testes forem positivos, o STARTUP completará sua operação e exibirá a seguinte mensagem na tela:

```
*** PARABENS VOCE FOI BEM SUCEDIDO dBASE II INSTALADO ***
```

O programa INSTALL deverá ser executado, caso qualquer um dos três primeiros testes seja negativo. A esta altura, executar o INSTALL significa ter de descobrir os códigos para efeitos especiais de tela pertinentes à tela específica do usuário. Se tais códigos não forem conhecidos, o INSTALL não fornecerá nenhuma indicação. Seguir os procedimentos do INSTALL indicados no programa STARTUP, ou os procedimentos do INSTALL resumidos abaixo:

Instalando o dBASE II

O programa INSTALL adapta o dBASE II ao computador específico do usuário. Se o programa STARTUP tiver sido executado e alguns dos testes apresentarem resultados negativos, é possível entrar com o programa INSTALL diretamente do STARTUP bastando, para tal, seguir as indicações da tela. Caso contrário, digitar QUIT após o ponto do dBASE, a fim de interromper o dBASE e de retornar ao sistema operacional. Digitar INSTALL depois de A >.

```
A> INSTALL <RETURN>
```

O programa INSTALL responderá com:

```
OPERACAO COM TELA CHEIA (Y/N)? Y
```

Responder Sim, através do Y.
Praticamente todos os computadores
populares podem posicionar o cursor
em qualquer lugar de suas telas de terminais.
Não se tendo absoluta certeza sobre
este assunto, consultar o fornecedor.

Em seguida, o dBASE apresentará dois menus de nomes de computadores e de terminais. Se o nome do computador em uso não constar de nenhum deles, consultar o manual de operações do computador. Ele indicará qual o terminal que poderá melhor atender às características do computador.

Se nem o computador nem o terminal pertencerem ao menu, entrar com os códigos que adaptam o dBASE ao monitor de vídeo em uso. Escolher, então, a opção "Z" do menu.

Deve-se entrar com os códigos adequados às seguintes características. Alguns esclarecimentos, por parte do fornecedor do hardware, podem ser necessários.

1. DELETAR UMA SEQÜÊNCIA DE CARACTERES
2. SEQÜÊNCIA DE POSICIONAMENTO DIRETO DO CURSOR
3. COMANDO DE LIMPEZA DA TELA
4. COMANDO DE REPOSICIONAMENTO DO CURSOR
5. OPCIONAIS: COMANDOS DE INTENSIDADE OU COMANDOS VÍDEO REVERSO.

Se o terminal em uso constar do menu, entrar com a letra referente ao terminal.

```
SELECIONE O TERMINAL:
```

Ao que o dBASE perguntará:

```
SUBSTITUI MACRO,DATA,ETC. (Y/N)? <RETURN>
```

O dBASE exibirá, então, a seguinte mensagem:

```
PRESSIONE "Y" PARA GRAVAR, OU OUTRA TECLA PARA ABORTAR O INSTALL
```

Digitando Y, preservam-se as especificações e códigos cujas entradas se realizaram no decorrer da sessão do INSTALL. Digitando qualquer outra tecla, aborta-se esta sessão do INSTALL, podendo, então, começar tudo de novo caso algum erro tenha sido cometido. O Disco de Reserva está, agora, preparado para ser copiado em um Disco de Trabalho.

Desejando-se alterar os efeitos de tela ou trocar o formato do disco de MM/DD/AA para DD/MM/AA, pode-se, posteriormente, reinstalar o dBASE.

Voltar, agora, à página 19 e preparar um Disco de Trabalho, partindo-se deste DISCO DE RESERVA INSTALADO. Preparado o Disco de Trabalho, prosseguir com os testes de funcionamento do dBASE.

CAPÍTULO 4

TESTANDO O FUNCIONAMENTO DO dBASE

Este capítulo visa descrever o dBASE passo a passo e explicar cada entrada. Qualquer dificuldade que surgir na tela será esclarecida no parágrafo seguinte ou na próxima página.

Não estando familiarizado com o teclado do computador, vale a pena praticar um pouco. Em geral, estes teclados são mais leves ao toque do que os de uma máquina de escrever; muitos deles, inclusive, podem repetir uma tecla de modo automático, quando pressionada. Deve-se cuidar, então, para apertar cada tecla apenas uma vez, especialmente a tecla < RETURN >.

Vamos Começar

Carregar o dBASE na memória do computador, entrando-se com:

```
A>DBASE <RETURN>
```

Terminar sempre a linha de entrada com a tecla < RETURN >.

O dBASE responderá com:

```
DIGITE A DATA DE HOJE OU PRESSIONE RETURN
(DD/MM/AA):
```

Apenas por brincadeira, entrar com uma data NÃO-VÁLIDA, tal como:

```
DIGITE A DATA DE HOJE OU PRESSIONE RETURN
(DD/MM/AA): 2/39/84  <RETURN>
```


Em alguns computadores, como o IBM-PC, a entrada da data será feita automaticamente não aparecendo, então, as mensagens. (Veja as duas últimas telas da página 24)

Pressionar < RETURN > para entrar com a linha de entrada de dados. Pressionar apenas uma vez!

Ter-se-á, então, provocado o humor do dBASE II. Infelizmente, este será todo o humor a ser proporcionado pelo dBASE.

Agora, pode-se inserir a data, com qualquer uma das formas a seguir:

Atenção:
Um l (ele) não é um 1 (um). Não usar a tecla da letra minúscula l em substituição à do número 1. Esta não é uma máquina de escrever!

1/1/84 1-1-84 1,1,84 1.1.84 1 1 84 01/01/84 etc.

Escolher qualquer formato desejado e digitar a data de hoje. Não esquecer de < RETURN >.

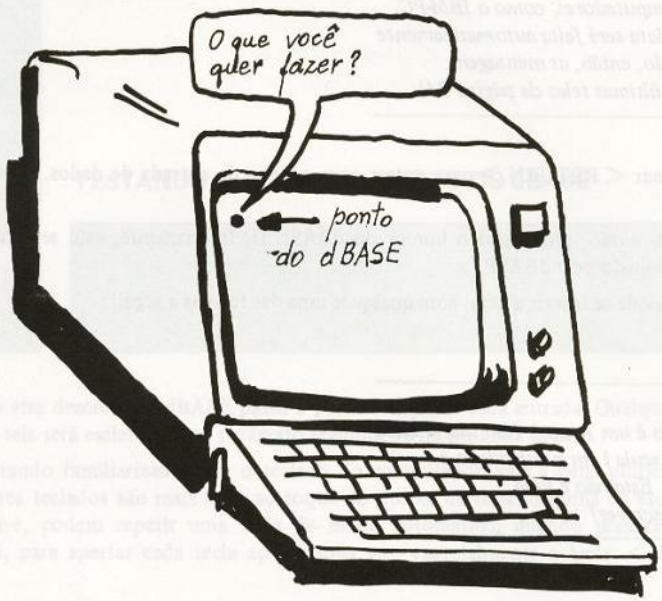
Tendo-se inserido a data (ou pressionado < RETURN > para saltá-la), o dBASE exibirá a seguinte mensagem:

```
Copyright (C) 1982 RSP Inc.
*** dBASE II Ver 2.4   26 Mar 1983
."Digite 'HELP', 'HELP dBASE', ou um comando"
```

O dBASE apresenta o seu ponto.

O Ponto do dBASE

Na tela, o ponto do dBASE pode não significar muito mas, na realidade, fica-se satisfeito cada vez que ele aparece. Ele significa que o dBASE está perguntando: "O que você quer fazer?"



Além disso, quando não se tem certeza se está tudo correndo bem, é muito bom ver o velho ponto do dBASE novamente na tela.

Acabou-se de aprender como acessar o dBASE. Querendo-se encerrar a sessão, digitar:

```
. QUIT <RETURN>
```

o dBASE RETORNARÁ AO SISTEMA OPERACIONAL (CP/M, MS-DOS etc.), passando-se a ter o A> novamente na tela.

Agora que se sabe como proceder, começar tudo do início e carregar o dBASE outra vez na memória do computador, entrando-se com:

```
. A> DBASE <RETURN>
```

Lembre-se de Como Sair

Pode-se encerrar o dBASE – através do comando QUIT – sempre que surgir o ponto. Se ele não aparecer na tela, acionar < ESCAPE > (ou as teclas < CTRL > e Q juntas, nos modos Append, Edit ou Browse, para trazê-lo de volta).

Em Caso de Erro

Cometendo-se um erro e inserindo-se um comando que o dBASE não compreende, recebe-se uma mensagem de erro e o dBASE pergunta se se deseja corrigir a linha de comando.

Tente, também, resolver uma situação-problema idêntica à apresentada, digitando:

```
. PREPARAR ALMOCO <RETURN>
```

O dBASE responde:

```
*** COMANDO DESCONHECIDO
PREPARAR ALMOCO
CORRIGIR E ENTRAR DE NOVO (Y/N)? N
```

Pressionando-se N ou <ESC >ou <RETURN >, traz-se o ponto de volta.

O momento adequado de se dizer SIM à pergunta CORRIGIR E TENTAR NOVAMENTE? é quando se entra com uma linha de comando muito grande e se deseja corrigir apenas uma pequena parte dela. (O processo de correção será descrito no Capítulo 5.) Para comandos menores, é mais fácil apertar < RETURN >, trazer o ponto de volta e escrever novamente a linha.

Vamos, agora, CRIAR um arquivo chamado PESSOAL

Pode-se entrar com os comandos com letras maiúsculas ou minúsculas.

Entrar com:

```
. CREATE PESSOAL <RETURN>
```

ATENÇÃO!
Apertar a tecla RETURN para inserir os dados, mas apenas uma vez!

Passo 1 – Inserir as quatro definições de campo seguintes, exatamente como aparecem nesta tela:

Pressionando-se < RETURN > ao final de cada linha, vai-se para a próxima linha. Começar, entrando com:

```
. CREATE PESSOAL
ENTRE COM A ESTRUTURA DO REGISTRO:
CAMPO       NOME, TIPO, TAMANHO, CASAS DECIMAIS
001         nome, c, 7
002         salario, n, 7,2
003         cargo, c, 15
004         ano, n, 4
005        <RETURN>
DESEJA COMECAR A ENTRADA DE DADOS AGORA? N
```

Passo 2 – Atingindo o CAMPO 005, pressionar < RETURN > para encerrar a definição dos campos e criar o arquivo.

Passo 3 – Entrar, aqui, com N (de Não). A entrada de dados será feita em um minuto.

Em caso de Erro ...

No início pode ser difícil acostumar-se ao toque leve exigido pelo teclado de um computador. Pressionando, inadvertidamente, < RETURN > muitas vezes, finaliza-se o comando CREATE antes do tempo. Se isto acontecer, entrar com N quando o dBASE pedir os dados de entrada.

```
DESEJA COMECAR A ENTRADA DE DADOS AGORA? N
```

A seguir, DELETAR o arquivo parcial criado, digitando:

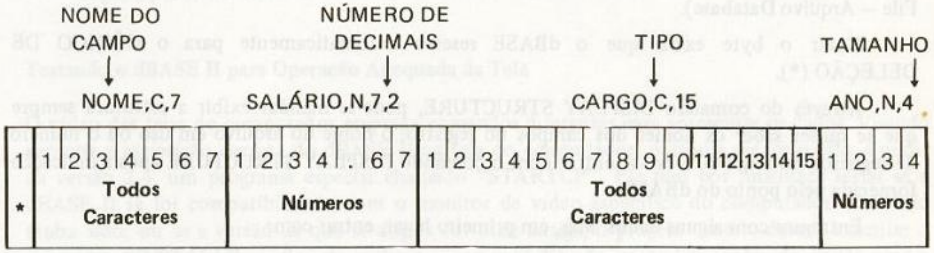
```
. CLEAR
. DELETE FILE PESSOAL
```

CLEAR fecha o arquivo, caso ele esteja aberto.
DELETE faz o arquivo desaparecer.

Começar novamente com:


```
. CREATE PESSOAL
ENTRE COM A ESTRUTURA DO REGISTRO:
CAMPO       NOME, TIPO, TAMANHO, CASAS DECIMAIS
001         nome, c, 7
002         salario, n, 7,2
003         cargo, c, 15
004         ano, n, 4
005        <RETURN>
DESEJA COMECAR A ENTRADA DE DADOS AGORA? N
```

Tendo entrado com o arquivo exatamente como no exemplo, a estrutura do registro recém-criado será:



CAMPO PARA O CÓDIGO DE DELEÇÃO
Este é o CAMPO PARA O CÓDIGO DE DELEÇÃO que o dBASE acrescenta automaticamente ao registro.

Pode conter apenas o código *.

Abrir o arquivo recém-criado. Inserir:

```
. USE PESSOAL
. DISPLAY ALL
.
```

Tudo que se tem é o ponto. Não existem dados no arquivo.

Em seguida, perguntar ao dBASE a descrição do arquivo. Digitar:

Ainda não se digitou qualquer dado.

Drive do Disco

```
. DISPLAY STRUCTURE
ESTRUTURA DO ARQUIVO: A:PESSOAL.DBF
NUMERO DE REGISTROS: 00000
DATA DA ULTIMA ATUALIZACAO: DD/MM/AA
BANCO DE DADOS DE USO PRIMARIO
CAMPO       NOME        TIPO        TAMANHO     DEC
001         NOME        C           007
002         SALARIO     N           007         002
003         CARGO       C           015
004         ANO         N           004
*** TOTAL  **                       00034
```

O dBASE acrescenta o tipo de arquivo "DBF" ao nome do arquivo (significando Database File – Arquivo Database).

Notar o byte extra que o dBASE reserva automaticamente para o CÓDIGO DE DELEÇÃO (*).

Através do comando DISPLAY STRUCTURE, pode-se mandar exibir a estrutura sempre que se quiser saber os nomes dos campos no registro, o nome do arquivo em uso ou o número de registros no arquivo. Deve-se entrar com o comando DISPLAY STRUCTURE após a indicação fornecida pelo ponto do dBASE.

Entremos com alguns dados. Mas, em primeiro lugar, entrar com:

```
. SET CONFIRM ON
```

Isto impedirá o deslocamento muito rápido de um registro para outro, no modo (APPEND) de entrada de dados. Tudo sobre o SET CONFIRM ON será visto mais tarde, no Capítulo 7.

Entrar, agora, no modo APPEND:

```
. APPEND
```

O dBASE responderá com:

```
REGISTRO # 00001
NOME      :          :
SALARIO   :          :
CARGO     :                    :
ANO       :     :
```


Entrar com os cinco registros seguintes na forma como eles aparecem neste exemplo. Pode-se usar tanto letras em caixa-alta, em caixa-baixa ou ambas, desde que a coerência seja mantida.

Pressionar <RETURN> para avançar para o próximo campo

```
REGISTRO # 00001
NOME      :Paulo     :
SALARIO   :250       :
CARGO     :Escriturario    :
ANO       :1980:
```

Pressionando <RETURN>, aqui, passa-se para o próximo registro

Saiu do APPEND Muito Rapidamente?

Pressionando < RETURN > em vez de entrar com os dados, e estando o cursor na primeira posição do primeiro campo no registro, sai-se do modo APPEND e traz-se o ponto de volta.

Está correto. Entrar apenas com:

```
. APPEND
```

para iniciar novamente o processo e entrar com os dados relativos ao próximo registro.

Digitar as seguintes informações:

	REGISTRO # 2	REGISTRO # 3	REGISTRO # 4	REGISTRO # 5
NOME	Luiz	Joao	Pedro	Lucio
SALARIO	650	350	850	700
CARGO	Editor	Escritor	Pres	Vice Pres
ANO	1972	1978	1950	1960

Chegando-se ao registro nº 6, pressione RETURN para interromper o modo APPEND.

ATENÇÃO!
Um arquivo que tenha recebido outros registros (através do comando APPEND), deverá ser reaberto antes de ser utilizado. Tal operação visa garantir que as alterações efetuadas no arquivo sejam escritas no disco.

```
. USE PESSOAL
. DISPLAY ALL
00001   Paulo          250.00  Escriturario    1980
00002   Luiz           650.00  Editor          1972
00003   Joao           350.00  Escritor        1978
00004   Pedro          850.00  Pres            1950
00005   Lucio          700.00  Vice Pres       1960
```

Observe o que foi colocado no computador. Tente exibir um único registro:

```
. DISPLAY RECORD 2
00002   Paulo          650.00  Editor          1972
```

Agora, para EDITAR o "REGISTRO 2", digitar:

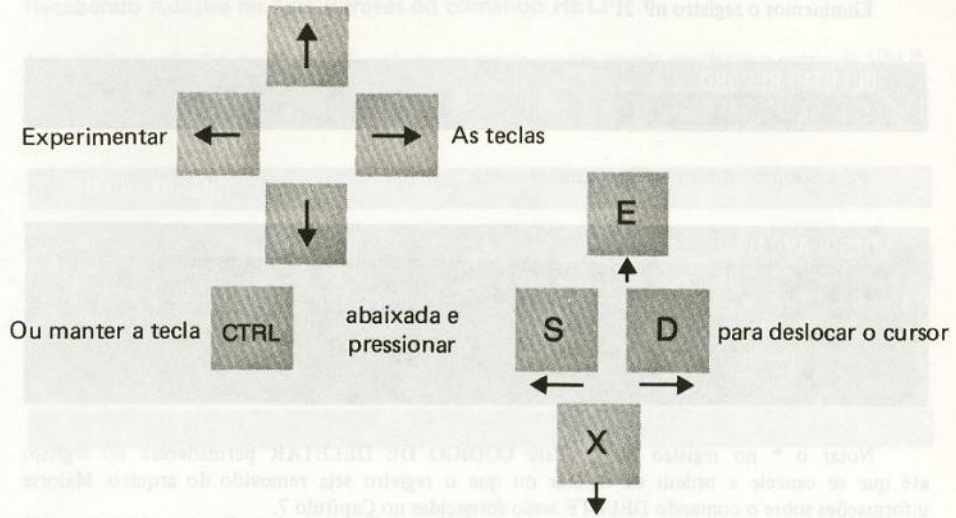
```
. EDIT 2
```

É inconsistente! Mas está correto!
Não se usa a palavra REGISTRO. Porém, um registro é a única coisa que se pode EDITAR.

Mover o cursor para um campo qualquer e alterá-lo, digitando por cima do que já está escrito.

Movimento do Cursor

Não manter as teclas do cursor pressionadas.
No modo EDIT o deslocamento, tanto para o registro seguinte quanto para o anterior, pode ser feito muito rapidamente.
CTRL quer dizer: tecla de controle.



Deslocando o cursor para baixo ou pressionando < RETURN > quando ele se encontrar no último campo, avança-se para o registro seguinte. As alterações no registro anterior são automaticamente preservadas.

Pode-se retroceder no arquivo, movendo o cursor para cima, depois de passado o primeiro campo, ou pressionando < CTRL R >.

Encerra-se o modo EDIT e traz-se o ponto de volta, entrando com:

< CTRL W > para encerrar e preservar alterações no registro em curso.

< CTRL Q > para encerrar e abandonar alterações no registro em curso.

Preserva-se a alteração efetuada no registro nº 2, entrando com < CTRL W >

Olhe, agora, o registro editado:

```
. DISPLAY RECORD 2
00002   Luiz           650.00 Pescador        1972
```

Voltar e fazer o registro nº 2 retornar à sua forma primitiva. Editar o registro (através do comando EDIT) e mostrá-lo, novamente, através do comando DISPLAY. Tendo esquecido como proceder, consultar a página anterior.

Eliminemos o registro nº 2:

```
. DELETE RECORD 2
00001 ELIMINACAO (S)
```

Para exibir o arquivo inteiro, entra-se com:

```
. DISPLAY ALL
00001   Paulo          250.00  Escriturario    1980
00002  *Luiz           650.00  Editor          1972
00003   Joao           350.00  Escritor        1978
00004   Pedro          850.00  Pres            1950
00005   Lucio          700.00  Vice Pres       1960
```

Notar o * no registro nº 2. Este CÓDIGO DE DELETAR permanecerá no registro até que se cancele a ordem de deletar ou que o registro seja removido do arquivo. Maiores informações sobre o comando DELETE serão fornecidas no Capítulo 7.

Examine as seguintes saídas obtidas:

```
. DISPLAY ALL NOME
00001   Paulo
00002  *Luiz
00003   Joao
00004   Pedro
00005   Lucio
```

```
. DISPLAY ALL NOME, SALARIO
00001   Paulo       250.00
00002  *Luiz        650.00
00003   Joao        350.00
00004   Pedro       856.00
00005   Lucio       700.00
```

O " > " significa "maior que".

```
. DISPLAY FOR ANO > 1970
00001   Paulo       250.00  Escriturario    1980
00002  *Luiz        650.00  Editor          1972
00003   Joao        350.00  Escritor        1978
```


Os dados de um campo alfanumérico devem vir entre aspas para que possam ser procurados.

NOME = "Paulo".
Lembrar, também, que se a entrada de dados tiver sido feita em caixa-alta e caixa-baixa, os dados a serem procurados deverão ter a mesma forma.

```
. DISPLAY FOR NOME = 'Paulo'
00001   Paulo       250.00 Escriturario    1980
```

Os comandos DISPLAY ALL e DISPLAY FOR oferecem muitas variações quanto a exibir e interrogar o arquivo. Tudo sobre o comando DISPLAY será visto mais tarde, no Capítulo 10.

Pode-se usar aspas simples ou duplas, desde que o tipo adotado seja mantido no início e no fim dos dados a serem procurados.

CARGO = "Escritor".
Verificar, posteriormente, em "Convertendo Caixa-Alta em Caixa-Baixa", no Capítulo 18.

```
. DISPLAY FOR CARGO = "Escritor"
00003   Joao        350.00  Escritor    1978
```

Inserir agora:

```
ERASE
```

Deseja-se uma tela limpa? Basta usar o comando ERASE para apagá-la!

Examinar os resultados obtidos mediante os dois comandos seguintes:

COUNT e SUM (contagem de registros e soma de valores de campos, respectivamente):

```
. COUNT
CONTAGEM = 00005
. COUNT FOR SALARIO > 500
CONTAGEM = 00003
. SUM SALARIO
2150.00
. SUM SALARIO FOR SALARIO>500
1550.00
```

Tente, a seguir:

```
. SUM NOME
EXPRESSAO NAO NUMERICA
SUM NOME
CORRIGIR E ENTRAR DE NOVO (Y/N)?
```

Viu o que aconteceu?

Pressionar < RETURN > ou < ESC > para trazer o ponto de volta.

Experimentemos o comando BROWSE.

Deslocar, primeiramente, o PONTEIRO DO REGISTRO para o início do arquivo. Inserir:

```
. 1
. BROWSE
```

A força real do comando BROWSE não fica tão evidente quando se lida com arquivos pequenos. Quando, porém, os registros ultrapassam a largura da tela e o arquivo torna-se maior, a capacidade de rodar a tela apresentada por este comando passa a ser muito útil.

No modo BROWSE desloca-se o arquivo verticalmente, para cima, através de:

< CTRL R >

E, para baixo, através de:

< CTRL C >

Notar que o comando BROWSE exibe os nomes dos campos como títulos de colunas, ao passo que comando DISPLAY, não.

Observe que o número do REGISTRO, na parte superior da tela, varia à medida que se desloca o cursor para cima e para baixo, no arquivo. O PONTEIRO de REGISTRO permanece sempre visível no sistema BROWSE.

Encerra-se o modo BROWSE pelo mesmo processo usado no modo EDIT

< CTRL W > para encerrar e preservar alterações no registro em curso.

< CTRL Q > para encerrar e abandonar alterações no registro em curso.

Entrar com:

< CTRL Q > para trazer o ponto de volta.

Ordenemos o arquivo, através do comando SORT, em uma seqüência diferente, entrando com:

```
. SORT ON ANO TO POSSE
CLASSIFICACAO COMPLETA
```

Está sendo criado um novo arquivo de dados, denominado "POSSE".

Abrir, então, o arquivo recém-criado:

```
. USE POSSE
. DISPLAY ALL
00001   Pedro        850.00  Pres          1950
00002   Lucio        700.00  Vice Pres     1960
00003   Joao         350.00  Escritor      1978
00004   Paulo        250.00  Escriturario  1980
```

Voltemos ao arquivo original e criemos um RELATÓRIO, entrando com:

```
. USE PESSOAL
. REPORT FORM PESSOAL
```

Está sendo criada uma forma de relatório chamada "PESSOAL". Os nomes das formas de relatório podem ser idênticos aos dos arquivos de dados, pois o tipo de arquivo é diferente (ver Apêndice D).

O dBASE responderá com uma série de perguntas que servirão de auxílio à descrição da forma do relatório. Respondê-las exatamente como o indicado a seguir:

```
. REPORT FORM PESSOAL
ENTRAR COM AS OPCOES, M=MARGEM ESQUERDA, L=LINHA/PAG; W=TAMANHO PAG.
```

(Pressionar <RETURN> para aceitar o formato-padrão de página de 8 1/2 X 11).

O símbolo <, como o primeiro caractere, significa alinhar o título da folha em relação à margem esquerda.

```
CABECALHO? (Y/N) Y
ENTRE COM O CABECALHO:< O RELATORIO DE PESSOAL
RELATORIO COM ESPACO DUPLO?    (Y/N) N
DESEJA TOTAIS?                 (Y/N) Y
DESEJA SUBTOTAIS NO RELATORIO?(Y/N) N
COL  TAMANHO, CONTEUDO
001
```

Deseja-se saber, aqui, a largura e o conteúdo de cada coluna da forma do relatório.

Não se pode alterar uma seleção se já se passou para a seguinte. Caso se cometa um erro e se deseje começar novamente a descrição do relatório, pressionar < ESC > a fim de trazer o ponto de volta.

Deleta-se a descrição da forma, digitando-se:

```
. DELETE FILE PESSOAL.FRM
```

e começa-se outra vez, com:

```
. REPORT FORM PESSOAL
```



```
COL         TAMANHO, CONTEUDO
001         7, NOME
ENTRE COM O CABECALHO: Nome
002         7, SALARIO
ENTRE COM O CABECALHO: Os Dolares
DESEJA TOTAIS? (Y/N) Y
```

Nos campos numéricos o dBASE pergunta se se deseja uma totalização.

```
003         4,ANO
ENTRE COM O CABECALHO: Ano
DESEJA TOTAIS? (Y/N) N
004
```

Pressionar <RETURN> para obter-se o relatório.

Com um pouco de prática, pode-se projetar algumas formas bem sofisticadas. A vantagem apresentada pelo comando REPORT é que basta definir a forma de relatório apenas uma vez, não sendo mais necessário repetir tal operação.

```
. REPORT FORM PESSOAL
        PAGE NO. 00001
        01/01/80

        O RELATORIO DE PESSOAL

        Nome        Os          Ano
                  Dolares
        Paulo     250.00        1980
        Joao      350.00        1978
        Pedro     850.00        1950
        Lucio     700.00        1960
        ** TOTAL**
              2150.00
```

O mesmo relatório poderá ser reemitido. Tendo um arquivo de dados cujos campos tenham a mesma denominação do relatório, pode-se criar um relatório a qualquer momento que se desejar.

Caso se disponha de uma impressora acoplada e ligada, insira:

```
. REPORT FORM PESSOAL TO PRINT
```

Este comando deverá imprimir a FORMA DE RELATÓRIO PESSOAL.

Tentemos mais duas coisas antes de ENCERRAR (QUIT). Em primeiro lugar, observe os nomes dos arquivos criados, entrando com:

```
. DISPLAY FILES
ARQUIVOS DE BANCO DE DADOS      # RGTS       ULTIMA ATUALIZACAO
PESSOAL    DBF                   00005       01/01/80
POSSE      DBF                   00004       01/01/80
```

Notou o arquivo "POSSE"? Vamos apagá-lo:

```
. DELETE FILE POSSE
O ARQUIVO FOI ELIMINADO
```

Exibindo os nomes dos arquivos novamente, o POSSE não deverá aparecer.

```
. DISPLAY FILES
ARQUIVOS DE BANCO DE DADOS      # RGTS       ULTIMA ATUALIZACAO
PESSOAL    DBF                   00005       01/01/80
```

Encerrar, agora, a sessão dBASE:

Certifique-se de ter usado o QUIT antes de desligar o computador.

```
. QUIT
```

Você entrou em contato com o "Grupo dos Doze do dBASE" e aprendeu, também, coisas úteis, tais como:

SET CONFIRM ON
ERASE
DISPLAY STRUCTURE
DISPLAY FILES,

e como proceder ante a situação CORRIGIR E ENTRAR DE NOVO? (Y/N). Você está no caminho para o domínio do dBASE.

PARTE 3

O GRUPO DOS DOZE DO dBASE

CAPÍTULO 5

TRABALHANDO COM O dBASE

Os Comandos Fundamentais de Processamento

Usando o Banco de Dados

CREATE – permite que se defina um novo arquivo de dados.
USE – abre o arquivo.
QUIT – encerra a sessão em curso.

Entrada e Deleção de Dados

APPEND – permite que se acrescentem novos registros ao arquivo.
DELETE – assinala os registros a serem deletados.

Alterando os Dados

EDIT – permite que se alterem os dados.
BROWSE – permite que se alterem os dados.

Colocando os Dados em outra Seqüência

SORT – cria um novo arquivo, em ordem seqüencial.

Extraindo Dados e Informações

DISPLAY – exibe e interroga os dados.
COUNT – conta os registros.
SUM – totaliza os dados numéricos.
REPORT – prepara relatórios com títulos e totais.

Carregando o dBASE

O principal programa dBASE no disco é o DBASE.COM. Carregar o dBASE na memória do computador, digitando:

```
A> DBASE
```

O dBASE pedirá a introdução da data do sistema, cuja entrada poderá ser feita por * :

M/D/A **M-D-A** **M,D,A** ou **M D A**

O formato de datas do dBASE poderá ser alterado para (DD/MM/AA) através do programa INSTALL.

Pode-se pressionar < RETURN > e saltar a data do sistema. Porém, inserindo-se esta, o próprio dBASE se encarregará de verificar se ela está correta.

O Ponto do dBASE

O ponto do dBASE, seguido do cursor, indica que ele está à espera de uma ordem. Não se pode digitar nada até que o ponto do dBASE apareça na tela.

Inserindo Comandos

As linhas de comando são digitadas após o ponto do dBASE e são finalizadas por um < RETURN >, podendo conter até 254 caracteres de instrução.

* M = Mês; D = Dia; A = Ano.

Se a linha de comando exceder a largura da tela, será necessário subdividi-la em linhas menores. Dependerá do terminal.

É possível dividi-la por intermédio de um ponto e vírgula (;) e continuar o comando na linha seguinte. O dBASE não executa uma linha de comando, mesmo pressionando-se < RETURN >, se o RETURN foi imediatamente precedido por um ponto e vírgula.

Por exemplo:

```
  DISPLAY FOR COR = 'VERMELHO'  .OR.;
COR = 'VERDE'  .OR. COR;
= 'MARRON' .OR. COR = 'AMARELO';
.OR. COR = 'LARANJA'  .OR.COR=;
'CINZA'
```

O ponto e vírgula, porém, não deverá ser usado para separar palavras; a única coisa que pode vir após um ";" é um < RETURN >.

Repetindo um Comando

Para repetir o último comando inserido, pressionar < CTRL R > quando o ponto aparecer.

Cancelando um Comando

Para cancelar o comando em curso, pressionar a tecla < ESC > (escape).

Abreviando os Comandos

Todos os nomes dos comandos dBASE podem ser reduzidos aos seus quatro primeiros caracteres. Isto se aplica apenas à linguagem de comando dBASE e não a nomes de dados inventados pelo usuário. Por exemplo:

Forma Completa:	Forma Abreviada:
DISPLAY STRUCTURE	DISP STRU
DISPLAY RECORD	DISP RECO
DISPLAY FOR	DISP FOR
DISPLAY MEMORY	DISP MEMO
COPY STRUCTURE	COPY STRU
DESCENDING	DESC
MODIFY STRUCTURE	MODI STRU
MODIFY COMMAND	MODI COMM
SELECT SECONDARY	SELE SECO
DELETE RECORD	DELE RECO

Limpando a Tela

Limpa-se a tela, a qualquer momento, inserindo-se:

```
. ERASE
```

Correção de Erros

Se o dBASE não entender um comando, exibirá a mensagem de erro *** COMANDO DESCONHECIDO, juntamente com CORRIGIR E ENTRAR DE NOVO (Y/N)?.

Se a palavra de comando estiver correta, mas houver algum outro erro na linha de comando, obter-se-á uma mensagem *** ERRO DE SINTAXE, bem como CORRIGIR E ENTRAR DE NOVO (Y/N)? e um ponto de interrogação no início da área de confusão.

Pressionando-se < RETURN >, salta-se o CORRIGIR E ENTRAR DE NOVO e traz-se de volta o ponto. Pressionando-se o Y, obtém-se o modo de edição de "busca e substituição", que permite buscar e substituir os caracteres incorretos. Por exemplo:

```
. DISPLAY NOEM
***ERRO DE SINTAXE
      ?
DISPLAY NOEM
CORRIGIR E ENTRAR DE NOVO (Y/N)?  Y
MUDAR DE:   EM
MUDAR PARA: ME
DISPLAY NOME
MAIS CORRECOES (Y/N)?           N
```

Mensagens de Erro

O dBASE envia vários tipos de mensagens de erro que ajudam a esclarecer quaisquer problemas ocorridos. A partir da versão 2.4 do dBASE II, pode-se exibir, na tela, todas as possíveis mensagens de erro. Basta inserir:

```
. HELP ERROR
```

Pode-se também consultar o sumário de mensagens de erros no Apêndice C.

Recebendo Auxílio na Tela (através do comando HELP)

A partir da versão 2.4 do dBASE II, pode-se conseguir auxílio na tela mediante o comando HELP. Este, seguido por um nome de comando específico, exibirá uma breve explicação do uso do comando e uma indicação de sua sintaxe adequada. Digitar-se-ia por exemplo:

```
. HELP APPEND
```

```
. HELP DELETE
```

Pode-se exibir uma lista de todos comandos dBASE e das categorias adicionais do comando HELP, inserindo-se:

```
. HELP
```

Trabalhando com Arquivos em Drives Diferentes

Sempre que se faz referência a um arquivo, o dBASE vai, à revelia, para o mesmo disco em que programa dBASE está armazenado. Pode-se levar o disco procurado para um outro drive, digitando-se:

```
. SET DEFAULT TO B:
```

```
. USE B:PESSOAL
```

Sempre que se fizer referência a um arquivo que apresente uma localização de drive como parte do nome, o dBASE preterirá o drive default *e usará o drive de referência. Por exemplo:*

```
. REPORT FORM C:RAZÃO
```

Trocando Discos Flexíveis

Usando o CP/M, o sistema deverá ser sempre informado sobre qualquer remoção ou substituição de um disco flexível por outro. Inserir:


```
. RESET
```

A Data do Sistema

A data pode ser introduzida tanto ao se iniciar o dBASE quanto posteriormente, através do comando SET DATE.

Por exemplo:

```
. SET DATE TO 20/06/84
```

Usando o comando SET DATE não haverá validação de data, que poderá ser disposta segundo os seguintes formatos:
DD/MM/AA *ou* AA/MM/DD.
Nenhum controle de exatidão, porém, será realizado.

Quando a data do sistema for criada e editada, o dBASE irá, automaticamente, armazená-la em um arquivo.

Pode-se exibir a data do sistema através de:

```
. ? DATE( )
```

A data do sistema pode ser colocada em todos os registros mediante o comando REPLACE ALL. Dispondo, no registro, de um campo de caracteres de 8-bytes denominado DATADEHOJE, usar-se-ia:

Verificar se isto é realmente o que se pretende. REPLACE ALL altera todos os registros no arquivo.

```
. REPLACE ALL DATADEHOJE WITH DATE( )
```

As datas nos registros devem estar no formato AA/MM/DD, visando uma melhor ordenação e verificação. Por exemplo:

```
. SET DATE TO 84/06/20
```

CAPÍTULO 6

PLANEJANDO E USANDO O BANCO DE DADOS

Projetando a Estrutura do Registro

Critérios de Projeto

Há certos critérios que devem ser considerados quando se projeta a estrutura do registro. São eles:

- **Análise do Assunto**
- **Identificação do Campo-Chave**
- **Dados *versus* Informação**
- **Dados *versus* Palavras**

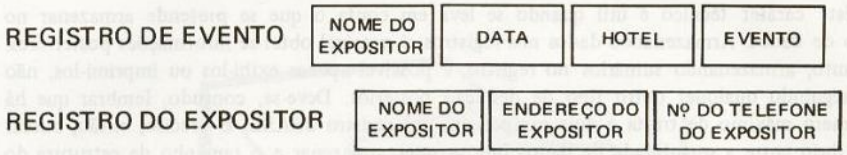
Análise do Assunto: Em primeiro lugar, é necessário escolher os assuntos do registro, o que, em alguns casos, pode ser tarefa óbvia. Por exemplo: nomes, endereços, produtos ou clientes. Muitas vezes, porém, o que parece ser um único assunto constitui, na realidade, dois ou mais e esta situação exige mais de um arquivo no banco de dados. Suponha, a título de ilustração, que você esteja encarregado de atender aos expositores incumbidos de apresentar os serviços da companhia onde trabalha, estando a seu cargo, também, a reserva de hotéis e de outras acomodações no decorrer desses eventos. Poderia considerar, primeiramente, um registro com o seguinte aspecto:

EVENTO	DATA	HOTEL	NOME DO EXPOSITOR	ENDEREÇO DO EXPOSITOR	Nº DO TELEFONE DO EXPOSITOR

Neste caso, o evento é o assunto e o seu registro cobre as informações necessárias sobre o orador. Suponha, agora, que haja um certo número de eventos relativos a um mesmo expositor. Com este projeto, nunca poderá ser esquecida a atualização de todos os registros de eventos

sempre que ocorrer uma alteração no endereço ou no número do telefone de um expositor. À medida que os arquivos aumentam, as probabilidades de dessincronização entre os dados também crescem.

Este problema de redundância pode ser resolvido mediante a criação de dois arquivos com as seguintes estruturas de registro:



Cada arquivo representa um assunto: o arquivo de eventos é o objetivo principal deste sistema, possuindo também um segundo arquivo, contendo dados relativos ao endereço do expositor. Quando o endereço ou o número do telefone de um dos expositores mudar, apenas um registro precisará ser corrigido a fim de se manter o banco de dados atualizado.

Usando-se o comando relacional JOIN, estes dois arquivos poderão, quando necessário, ser combinados a um terceiro, tendo em vista relatórios. Para combiná-los, porém, é preciso cuidado para que os nomes dos expositores sejam absolutamente idênticos em ambos os registros pois, caso contrário, não haverá critério de verificação e ligação. Como há maior probabilidade de ocorrerem erros de ortografia nos dados alfabéticos, obtém-se melhores resultados associar, a cada expositor, um algarismo pequeno, a fim de garantir maior precisão à verificação e ligação.

O volume de registros contidos no arquivo determinará o trabalho extra a ser executado; quanto menor for o arquivo mais fácil será o controle. É preferível desenvolver, com antecedência, métodos que evitarão problemas posteriores, à medida que os arquivos forem crescendo. Projetar estruturas de registro é uma das funções básicas de projeto de um Analista de Sistemas ou de um Analista de Banco de Dados.

Não é aconselhável, logo de início, empreender um projeto muito ambicioso; o ideal seria projetar um arquivo e familiarizar-se com o seu processamento.

Deve-se ter em mente que o dBASE II é um Gerenciador de Banco de Dados. As estruturas de registro poderão ser posteriormente modificadas, e os dados nelas contidos convertidos de modo automático, quando necessário. A conclusão será: à medida que os projetos forem se sofisticando, melhores resultados serão obtidos se primeiramente analisarmos o problema e organizarmos o banco de dados no papel.

Identificação do Campo-Chave: Todos os registros no arquivo deverão ser distinguíveis dos demais. Geralmente um campo único (p. ex., um nome ou um número) identifica um registro como sendo uma solução separada de dados. Algumas vezes necessitam-se dois campos, tais como número e data. Não importa quantos campos sejam exigidos, mas apenas que a combinação de dados diferencie, dos demais, cada registro de um arquivo. Estes campos de identificação são chamados CAMPOS-CHAVE.

Lembre-se: se puderem existir em seu arquivo registros inteiramente duplicados você não terá meios de saber se eles estão certos ou errados.

Dados *versus* **Informação**: Embora na maior parte do tempo os termos *dados* e *informação* sejam usados como sinônimos, existe uma diferença entre eles. Tecnicamente, os dados são fatos e cifras brutos que não podem ser separados. Os dados processados constituem a informação, isto é, dados que já foram, de algum modo, resumidos ou calculados. Por exemplo: as tarifas horárias e as horas trabalhadas constituem dados; o pagamento bruto, porém, obtido ao se multiplicarem os dois anteriores, é uma informação.

Este caráter técnico é útil quando se leva em conta o que se pretende armazenar no registro de dados. Armazenando dados nos registros, é possível obter-se informações posteriores. Entretanto, armazenando sumários no registro, é possível apenas exibi-los ou imprimi-los, não se conseguindo qualquer outro tipo de dedução posterior. Deve-se, contudo, lembrar que há um número máximo de trinta e dois campos em um registro dBASE. É preciso, então, buscar o equilíbrio entre a quantidade de dados que se quer armazenar e o tamanho da estrutura do registro.

Consultar Varredura de Campo, na parte referente ao DISPLAY no Capítulo 10, para a obtenção de exemplos de armazenamento de dados e palavras, no mesmo registro.

Dados *versus* **Palavras**: Os dados, ao contrário das palavras, são definidos por sua localização exata no registro. Necessitando de múltiplas descrições, características ou condições sobre o assunto, pode-se, simplesmente, escrevê-las em um grande arquivo de dados e tratá-las como palavras. O dBASE permite a criação de um campo que chega até 254 bytes. Sendo possível editar (inserir e deletar) e realizar busca em um campo, o acesso aos dados poderá ser feito em qualquer ordem desejada. Escrevendo cada uma das entradas de dado, em registro, segundo um mesmo modo, torna-se possível a busca de condições. As qualidades de um produto ou de uma casa à venda, as habilidades ou a educação de uma pessoa ou os ingredientes de uma receita culinária podem ser citados como exemplo destas condições. A última linha de uma entrada de dados visa à uniformidade e um planejamento antecipado.

Definindo os Dados

Definindo datas

Uma data pode ser definida em um campo numérico ou de caracteres. Definindo-o como um C, pode-se armazenar nele hifens e barras, com o cuidado de colocá-los sempre no mesmo lugar. Por exemplo, 11-12-42 e 20-06-36 estão corretos, mas 11-12-42 e 20-6-36 não serão ordenados ou verificados de forma correta.

Para fins de ordenação e de busca, as datas deverão estar na seqüência "ano-mês-dia". Pode-se definir três campos separados – para ano, mês, e dia – e, depois, conectá-los, tendo em vista a exibição (ver, no Capítulo 18, como conectar campos), embora este método ocupe até três campos, em vez de um.

Definindo o Primeiro e o Último Nomes

Pretendendo-se ordenar o arquivo em seqüência nominal, surgem duas opções: criar campos separados para o primeiro e o último nomes, ou um campo com o último nome primeiro. Na primeira situação, ordena-se o primeiro nome seguido do último, podendo ser apresentado o nome inteiro em ambas as ordens. É possível, também, exibir e imprimir o nome como dois campos separados ou conectar o primeiro e o último nomes através do símbolo + (explicado no Capítulo 18).

Com a programação no Nível 2 pode-se procurar a vírgula e inverter a ordem do nome.

Lidando-se com apenas um campo que tenha sido introduzido primeiro com o sobrenome e depois outros, é necessária uma única passagem do comando SORT para colocar o arquivo em seqüência nominal. Porém, o nome será exibido sempre na ordem: sobrenome primeiro, e depois outros.

Definindo o Código Postal

Os códigos postais devem ser definidos mais como campos de caracteres do que como campos numéricos. Em primeiro lugar, serão sempre exibidos precedidos por zeros; em segundo, a busca é mais fácil nos campos "C", porque pode-se comparar o primeiro par de caracteres, em vez de se trabalhar com o campo inteiro. Querendo, por exemplo, selecionar apenas os códigos postais 100XX, tem-se a seguinte expressão para a busca:

```
. DISPLAY FOR CEP = '100'
```

Mas, para os códigos postais definidos como numéricos, a expressão será:

```
. DISPLAY FOR CEP > 09999  .AND.  CEP <10100
```


Definindo Números de Contas

Se os números de contas contiverem apenas números, poderão ser definidos tanto como campos numéricos quanto como caracteres. Aqui, novamente, tudo dependerá de como exibir os zeros, à esquerda ou não. Os zeros aparecerão nos campos de caracteres, mas não nos numéricos. Se o número da conta for composto de elementos distintos, como em um código postal, onde cada conjunto de caracteres significa algo específico como um estado ou cidade, então eles poderão ser definidos como um campo "C".

Dando Nome ao Arquivo

Regras para o Nome do Arquivo

1. Um nome de arquivo pode apresentar de 1 a 8 letras ou dígitos numéricos.
2. Não pode haver espaçamento entre os caracteres.
3. O primeiro caractere deve ser uma letra.

Os nomes dos arquivos devem ser os mais descritíveis possível, dentro das limitações impostas pelos oito caracteres. Em dBASE não se encontram, com freqüência, arquivos cujos nomes empreguem todos os oito caracteres. Se o disco contiver múltiplos arquivos de uma mesma categoria, deve-se abreviá-la e reservar espaço para um número ou uma data. Por exemplo:

```
OR062084      ou      PY062084      ou      CHECK001
OR121184              PY121184              CHECK023
```

Exemplos de nomes de arquivos válidos:

```
PESSOAL       VALIDAS       A          A1234567
ORDENS        ARQU26        ARQU0026   U4R3ED6
```

Exemplos de nomes de arquivos não-válidos:

```
26ARQU        COMPRASNOVAS      Z:ARQU        ARQU 323
$ARQU         ARQUVENDEDORES    B7.6.55       (ARQU A)
```

Dando o Nome dos Campos

Regras para os Nomes dos Campos

1. Um nome de campo pode apresentar de 1 a 10 letras, dígitos numéricos ou dois pontos.

2. Não pode haver espaçamento entre os caracteres.

3. O primeiro caractere deve ser uma letra.

O nome do campo pode conter até dez caracteres, ao contrário dos oito permitidos nos nomes dos arquivos. Quanto menor for o nome, menor será o trabalho de digitação ao teclado; porém, o nome pode não ser descritível o suficiente para lembrar o significado do nome do arquivo. Deve-se, pois, escolher nomes pequenos e significativos, como: NOME, CIDADE, ESTADO e CEP. Um nome como ENDEREÇO poderá ser abreviado por ENDR ou END. O dBASE permite que os nomes de arquivos sejam posteriormente alterados, se necessário, embora um planejamento criterioso evite trabalho extra. O importante é adotar uma uniformização logo de início. Deve-se definir o próprio dicionário para uso futuro. É o modo mais adequado de se definirem os sistemas de uma organização.

Exemplos de nomes de campos válidos:

```
NOME          ESTADO          PARTE:NO        CLIENTE
ENDERECO      ENDERECO2       CEP             CODIGO:ARJ
```

Exemplos de nomes de campos não-válidos:

```
2OENDERECO      ULT;NOME      DESCRITIVOS  DINHEIRO$$
PRIMEIRO NOME   PERCENTUAL%     ENDERECO#2    INGREDIENTES
```

Compatibilização do LAYOUT do Registro com o Documento-Fonte

Existe a opção de se projetar, para clientes, telas de entrada e de edição de dados que tornam a entrada de dados no sistema mais fácil, tanto para o usuário quanto para quaisquer outras pessoas (ver Capítulo 19). Porém, mesmo que se obtenham telas de entrada e de edição mais compreensíveis, ainda assim será necessária a referência aos nomes dos campos originais, quando se quiser buscar ou exibir os dados.

Se os dados forem obtidos de um documento-fonte padrão, a estrutura do registro deverá ser criada de forma que seus campos obedeçam à mesma ordem dos documentos. Isto tornará a entrada dos dados muito mais rápida e precisa. Se os dados forem obtidos de tipos diferentes do documento-fonte, as telas de entrada de dados para o cliente poderão ser ajustadas de modo a se assemelharem a cada um dos documentos-fonte distintos (ver Capítulo 19).

Selecionando o Tipo de Campo

Há três tipos de campo em dBASE: C = Caractere, N = Numérico, L = Lógico

Lembrar-se de que, através de modificação posterior da estrutura do registro, o tamanho dos campos C e N poderá ser aumentado ou diminuído (ver Capítulo 13).

TIPO DE CAMPO	DESCRIÇÃO	TAMANHO
C	Todos os caracteres. Bom para nomes, endereços, códigos postais, números de contas, números de telefone, datas, texto ou quaisquer dados alfabéticos.	1 a 254 bytes ou caracteres.
N	Dados Numéricos. Quaisquer números a serem calculados deverão ser definidos como um campo "N".	1 a 63 bytes. Depois de se inserir a largura de um campo numérico, existe a opção de se especificar o número de casas decimais. Digite uma vírgula seguida pelo número de casas decimais desejadas à direita do ponto. Por exemplo: SALARIO,N,7,2 cria o seguinte campo NNNN.NN

L	TRUE/FALSE, YES/NO (Verdadeiro/Falso, Sim/Não) Este campo abrange apenas: T,t,F,f,Y,y,N,n.	Sempre 1 byte. Não é necessário inserir a largura de um Campo Lógico

Buscando os Três Tipos de Campo

Ao se buscar os dados, as condições de verificação são definidas de modo diferente:

Exemplos de busca de dados numéricos:

```
. DISPLAY FOR NUMERO = 500
. DISPLAY FOR NUMERO > 50 .AND. NUMERO < 351
. DISPLAY FOR ANO > = 1942
```

Exemplos de busca de dados de caracteres:

ATENÇÃO:
Os dados de caracteres devem vir entre aspas (são usadas aspas simples e duplas), mas o tipo escolhido no início deverá ser mantido no fim. A tecla de acento não deverá ser usada para simular aspas simples.

```
. DISPLAY FOR CARGO = "Escriturario"
. DISPLAY FOR CARGO = "Escriturario" .OR. CARGO = "Escritor"
. DISPLAY FOR CEP    > =  "125"  .AND.  CEP < = "234"
. DISPLAY FOR MOLESTIA = "Caimbra de escritor"
```

Exemplos de busca de dados lógicos:

```
. DISPLAY FOR BRASILEIRO
. DISPLAY FOR SR:CIDADAO
. DISPLAY FOR FUTEBOL
. DISPLAY FOR .NOT. BRASILEIRO
. DISPLAY FOR .NOT. SR:CIDADAO
. DISPLAY FOR .NOT. FUTEBOL
```


Exemplos de busca de combinações de dados:

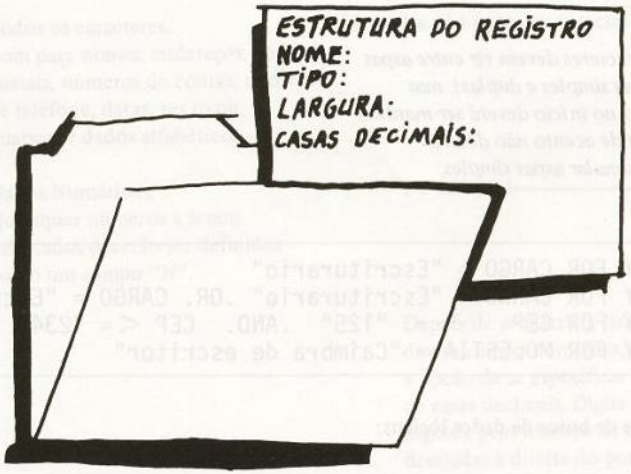
Notar o uso do ponto e vírgula para interromper a linha de comando.

```
. DISPLAY FOR FUTEBOL .AND. DEPARTAMENTO = "Executivo" .AND.;
IDADE>50
. DISPLAY FOR SR:CIDADAO .AND. FUTEBOL .AND. JURISDICAO =;
"Sul"
. DISPLAY FOR CARGO = "Escritor" .AND. Salario < 500
```

O Comando CREATE

- CREATE

abre um novo arquivo



O comando CREATE permite a entrada da definição de um novo arquivo dBASE. O dBASE apresentará instruções para a inserção do nome do arquivo e, depois, para nome, tipo e tamanho de cada um dos campos.

Acrescentando o nome do arquivo ao comando CREATE, o dBASE saltará a instrução para inserir o nome do arquivo.

```
  CREATE PESSOAL
ENTRAR COM A ESTRUTURA DO REGISTRO
NOME DO CAMPO, TIPO, TAMANHO, CASAS DECIMAIS
001
```

Para cada campo que se define, deve-se inserir o nome, tipo e tamanho.

As casas decimais são opcionais nos campos N.

Pode-se definir até trinta e dois campos. O modo CREATE é interrompido pressionando-se < RETURN >, quando o cursor estiver na primeira posição da linha.

Inserir os dados sem dar espaço entre as vírgulas. Por exemplo:

```
NOME,C,7
SALARIO,N,7,2
CARGO,C,10
ANO,N,4
```

Querendo-se um novo arquivo com a mesma estrutura de um já existente, pode-se saltar o procedimento do CREATE copiando-se, em um novo arquivo, a estrutura do arquivo existente.

```
. USE PESSOAL
. COPY STRUCTURE TO PESSOAS
```

ARQUIVO PESSOAL

1	Paulo	250.00	Escriturário	1980
2	Luiz	650.00	Editor	1972
3	João	350.00	Escritor	1978
4	Pedro	850.00	Pres	1950
5	Lucio	700.00	Vice Pres	1960

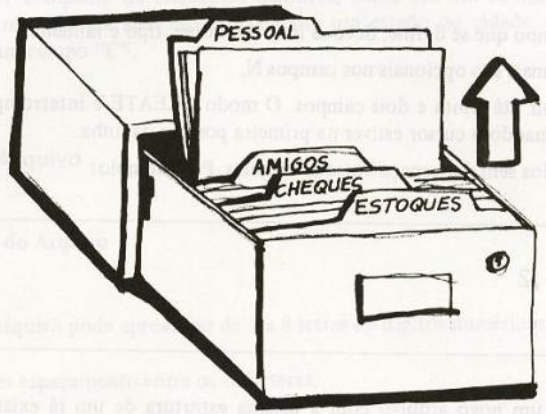
→ **PESSOAS**

Um novo arquivo foi criado sem que os campos tivessem de ser descritos novamente (ver Capítulo 11).

O Comando USE

- USE

abre um arquivo existente



O comando USE abre um arquivo.
Se na ocasião houver um arquivo diferente, aberto, o comando USE irá fechá-lo e abrirá o arquivo especificado.

```
. USE PESSOAL
. USE PESSOAS
```

Sozinho, sem um nome de arquivo, o comando USE fecha o arquivo em curso.

```
. USE
```

Pode-se abrir e trabalhar com dois arquivos ao mesmo tempo (ver Capítulo 17).

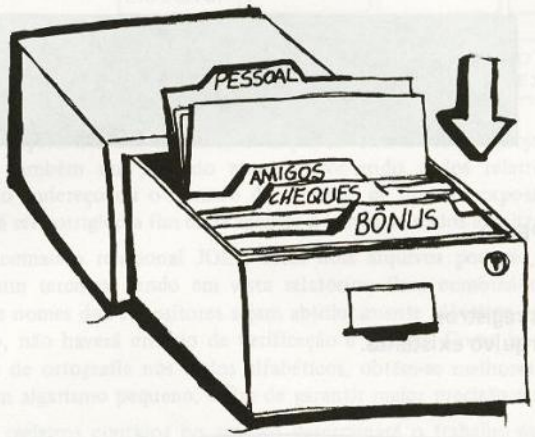
Desejando trabalhar com um arquivo que esteja em disco que não seja o do programa dBASE, por exemplo, no drive B, digitar:

```
USE B:PESSOAL
```

O Comando QUIT

- QUIT

fecha o arquivo,
salva-o e
retorna ao sistema.



O comando Quit encerra a sessão dBASE.

É muito importante encerrar o dBASE através do comando QUIT antes de se remover o disco e desligar o computador. Os dados ficam adequadamente preservados, em disco, mediante QUIT.

Se o sistema operacional em uso for o CP/M-80, pode-se empregar o comando QUIT e automaticamente executar outros programas, bastando, para tal, digitar QUIT TO seguido pelos nomes dos programas que se deseja executar. Por exemplo:

```
QUIT TO 'STAT', 'DBASE'
```

Cada programa a ser executado vem entre aspas.

Os nomes dos programas são separados por vírgulas.

No exemplo anterior, encerra-se o dBASE, executa-se o programa chamado STAT (um programa utilitário CP/M, que determina a quantidade de espaço disponível no disco), retornando-se, em seguida, ao dBASE.

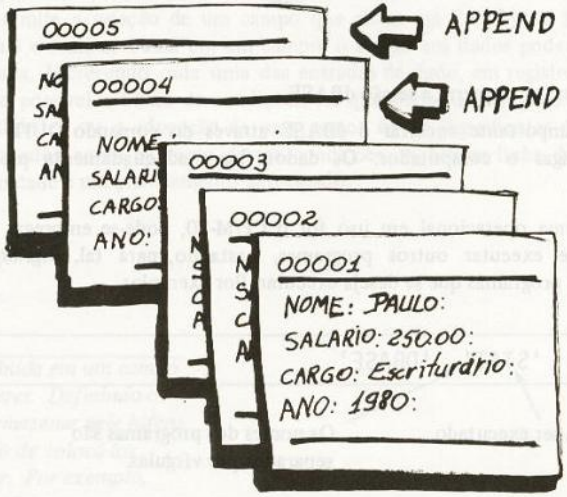
CAPÍTULO 7

ENTRADA E ELIMINAÇÃO (ATRAVÉS DO COMANDO DELETE) DE DADOS

O Comando APPEND

- APPEND

acrescenta novos registros
ao final de um arquivo existente.



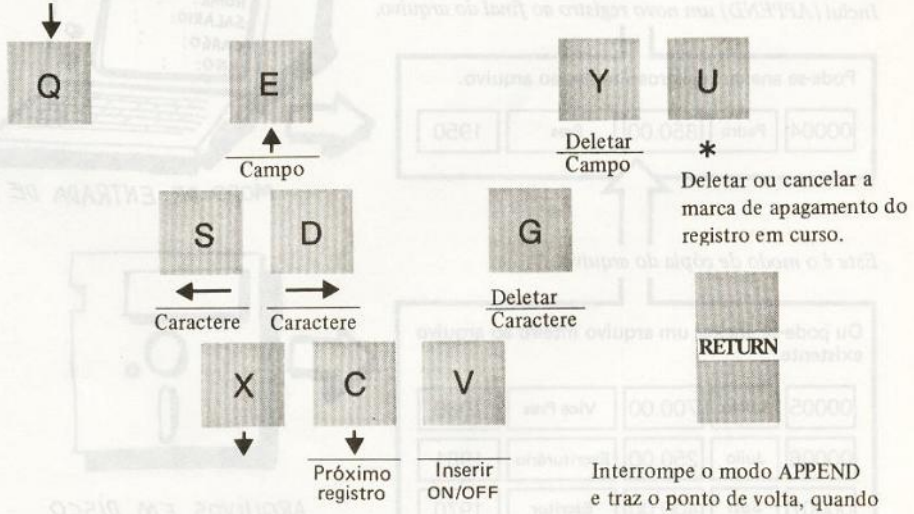
Sumário dos Comandos das TECLAS DE CONTROLE para Entrada e Edição de Dados

Estes comandos das teclas de controle requerem operações em full-screen. A maioria dos computadores permite tais operações e o funcionamento normal do dBASE é em FULL SCREEN ON. Se, porém, as operações em full-screen tiverem sido desligadas, deve-se ligá-las através de SET SCREEN ON.

Modo APPEND

Pressionar CTRL com as seguintes teclas:

Abandonar alterações no registro em curso e trazer de volta o ponto.

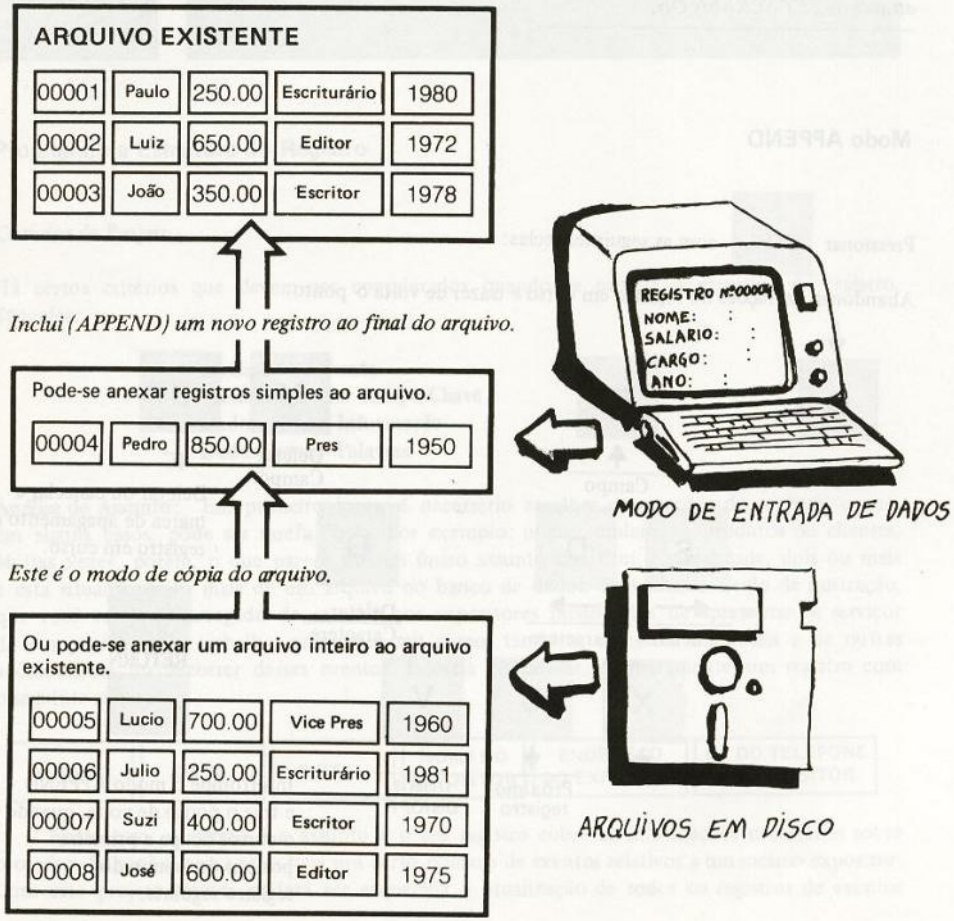


Se as operações em full-screen tiverem sido desligadas, ligue-as digitando:

```
SET SCREEN ON
```

antes dos modos APPEND, EDIT, INSERT ou BROWSE.

Este é o modo de inserção de dados.



Modo de Inserção de Dados

O comando APPEND introduz o modo de inserção de dados exibindo na tela uma estrutura de registro em branco. O APPEND começa pelo registro seguinte ao último do arquivo. Se não houver dados no arquivo, o APPEND começará pelo registro nº 1.

Os nomes dos campos são exibidos e os limites de cada um marcados por dois pontos:

```
. APPEND
REGISTRO # 00006
```

Assim que se insere um caractere na última posição do campo, o cursor se desloca para o campo seguinte ou, se o campo for o último, para o registro seguinte.

```
NOME      :Cristina:
SALARIO   :         :
CARGO     :         :
ANO       :         :
```

Pressionando-se < RETURN > quando o cursor se encontrar no último campo, desloca-se para o registro seguinte.

```
REGISTRO # 00032

NOME      :         :
SALARIO   :         :
CARGO     :         :
ANO       :         :
```

ATENÇÃO:
Lembre-se de fechar e de reabrir o arquivo usando o comando USE nome do arquivo, depois que os registros tiverem sido anexados (pelo APPEND). Isto garantirá que os novos registros serão preservados no disco e que o controle do arquivo será atualizado pela nova contagem correta dos registros.

Pode-se continuar com inserção de dados até chegar ao fim do disco ou ao registro nº 65 536, isto é, o que vier primeiro.

O modo APPEND é encerrado pressionando-se < RETURN > quando o cursor ocupar a primeira posição do registro seguinte.

É possível tornar mais lento o movimento do cursor, digitando:

```
. SET CONFIRM ON
```

antes de se usar o APPEND.

Enquanto o CONFIRM estiver acionado, será necessário pressionar <RETURN> para levar o cursor ao campo seguinte. Isto será de muita utilidade, especialmente se a última posição do registro encontrar-se sempre preenchida. Imediatamente após inserir o último caractere do registro, a estrutura em branco para o registro seguinte não mais aparecerá automaticamente, oferecendo chance para a confirmação dos dados de entrada.

Caso os registros contenham muitos dados repetitivos, pode-se transportar para o registro seguinte os dados cujas entradas já tenham sido realizadas.

```
. SET CARRY ON
. APPEND
```

```
REGISTRO # 00017

NOME      :Atila          :
COMPANHIA :Baja Bug       :
ENDERECO  :Estr Maua 961  :
CIDADE    :Rio de Janeiro :
ESTADO    :RJ:
CEP       :20000:
```

O registro seguinte será igual ao último. Os campos diferentes podem ser editados.

```
REGISTRO # 00018

NOME      :Atila          :
COMPANHIA :Baja Bug       :
ENDERECO  :Estr Maua 961  :
CIDADE    :Rio de Janeiro:
ESTADO    :RJ:
CEP       :20000:
```

Modo de Cópia do Arquivo

O comando APPEND FROM anexa um arquivo inteiro ao arquivo em uso.

```
. USE PESSOAL
. APPEND FROM AMIGOS
```

ARQUIVO PESSOAL

1	Paulo	250.00	Escriturário	1980
2	Luiz	650.00	Editor	1972
3	João	350.00	Escritor	1978
4	Pedro	850.00	Pres	1950
5	Lucio	700.00	Vice Pres	1960

ARQUIVO AMIGOS

1	Julio	250.00	Escriturário	1981	→	6	Julio	250.00	Escriturário	1981
2	Atila	400.00	Escritor	1970	→	7	Atila	400.00	Escritor	1970
3	Rogério	600.00	Editor	1975	→	8	Rogério	600.00	Editor	1975

O arquivo Amigos permanece inalterado.

O arquivo FROM (do qual os dados são extraídos) pode ser estruturalmente diferente do arquivo em uso. Apenas os dados dos campos com nomes correspondentes são copiados para o arquivo em uso.

ARQUIVO PESSOAL

	Nome	Salário	Cargo	Ano
1	Paulo	250.00	Escriturário	1980
2	Luiz	650.00	Editor	1972
3	João	350.00	Escritor	1978
4	Pedro	850.00	Pres	1950
5	Lucio	700.00	Vice Pres	1960

ARQUIVO AMIGOS

	Nome	Ordenados	Serviço	Início
1	Julio	250.00	Escriturário	1981
2	Atila	400.00	Escritor	1970
3	Rogerio	600.00	Editor	1975

Houve correspondência apenas nos campos "NOME"

6	Julio			
7	Atila			
8	Rogerio			

Se o campo do arquivo FROM for maior que o campo em uso, os dados "C" serão truncados e os dados "N" não serão copiados

	CAMPO FROM	campo no arquivo em uso
DADOS "C"	James Bond →	James
DADOS "N"	174532.6 →	0

O comando APPEND FROM FILENAME FOR anexa registros selecionados de um arquivo ao arquivo em uso:

```
. USE PESSOAL
. APPEND FROM AMIGOS FOR ANO > 1930
```

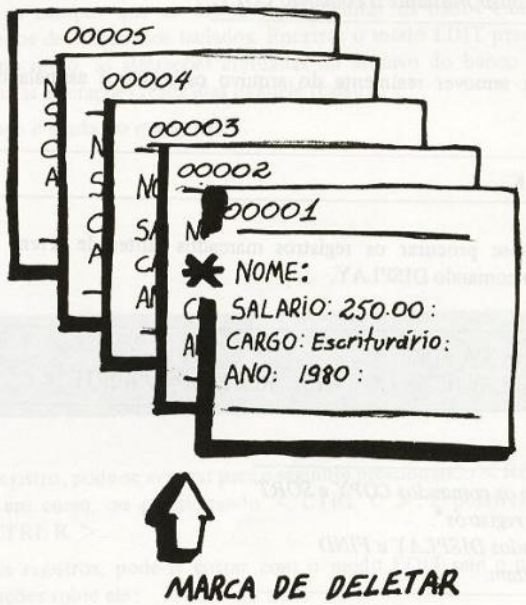
O comando APPEND FROM pode copiar arquivos de discos diferentes:

```
. USE PESSOAL
. APPEND FROM B:PESSOAL
```

O APPEND FROM é capaz de converter alguns arquivos de dados de outros programas em arquivos dBASE (ver Capítulo 19).

O Comando DELETE

- DELETE assinala um registro a ser deletado.
- PACK deleta os registros assinalados



O comando DELETE pode ser usado para deletar tanto registros como arquivos.

Lembrar-se de que o comando DELETE apenas assinala, através do código de deleção (*), o registro a ser deletado.

Para deletar um registro específico, digitar:

```
. DELETE RECORD 24
```

Para deletar todos os registros que atendam a uma condição, digitar:

```
. DELETE FOR COR = "VERMELHO"
```

Com o sistema operacional CP/M, o arquivo resultante ainda ocupará o mesmo espaço no disco. Para liberar o espaço no disco, é preciso copiar mediante o comando COPY.

Para remover realmente do arquivo os registros assinalados, apagando-os em definitivo, digitar:

```
. PACK
```

Pode-se procurar os registros marcados (antes de serem apagados pelo PACK), usando o * com o comando DISPLAY.

```
. DISPLAY FOR  *
. DISPLAY FOR  * .AND. REGIAO = "NORTE"
```

*Notar que os comandos COPY e SORT saltam os registros *. Os comandos DISPLAY e FIND não os saltam.*

Mudando de idéia (antes de se apagar o arquivo mediante PACK), pode-se cancelar a marca de deleção dos registros, digitando:

```
. RECALL RECORD 23
. RECALL ALL
. RECALL FOR DATA >= '830507'
```

A partir da versão 2.4, porém, os registros deletados podem ser tornados "invisíveis" ao se digitar:

```
. SET DELETE ON
```

Pode-se torná-los novamente visíveis através dos comandos DISPLAY e FIND:

```
. SET DELETE OFF
```

Para testar a condição de DELETAR INVISÍVEL, usar:

```
. DISPLAY STATUS
```

Não se dispondo da versão 2.4, saltar os registros deletados digitando:

```
. DISPLAY FOR .NOT.  *
```

Para deletar registros no meio de um arquivo use o comando DELETE NEXT. Suponhamos que se deseja DELETAR os registros de 321 a 326.

Ajustar a posição do ponteiro do registro.

```
. 321
. DELETE NEXT 6
```

Pode-se deletar os arquivos digitando:

```
. DELETE FILE PESSOAL
```

ATENÇÃO:
Não deletar o arquivo PESSOAL, pois ele será necessário para posteriores exemplos neste livro.

O comando DELETE FILE elimina o arquivo; e aqui não há segunda chance.

Um arquivo aberto não pode ser DELETADO. Pode-se fechar o arquivo aberto em curso, digitando:

```
. USE
```

Acrescentando o tipo de arquivo ao comando DELETE, pode-se deletar qualquer arquivo no disco:

```
. DELETE FILE RAZAO.FRM
```

Desejando deletar todos os dados do arquivo mantendo, ainda, sua estrutura para uso posterior, pode-se copiar a estrutura do arquivo original em um arquivo temporário e atribuir e este último um outro nome, depois que o arquivo original tenha sido deletado.

```
. USE ORIGINAL
. COPY STRUCTURE TO TEMP
. USE
. DELETE FILE ORIGINAL
  RENAME TEMP TO ORIGINAL
```

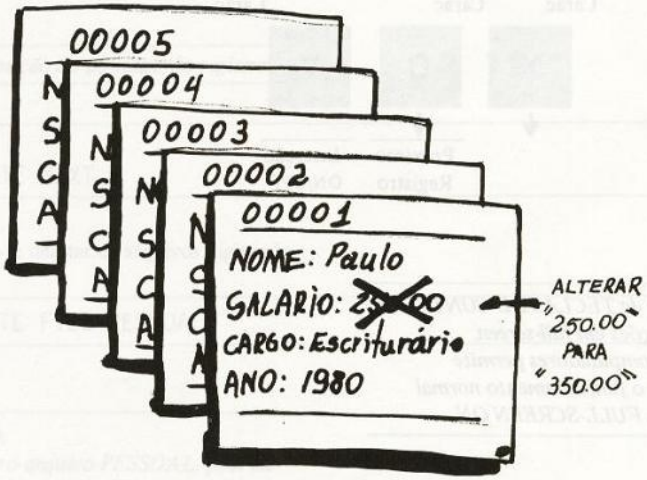
CAPÍTULO 8

ALTERANDO OS DADOS

O Comando EDIT

- EDIT

altera os dados existentes.

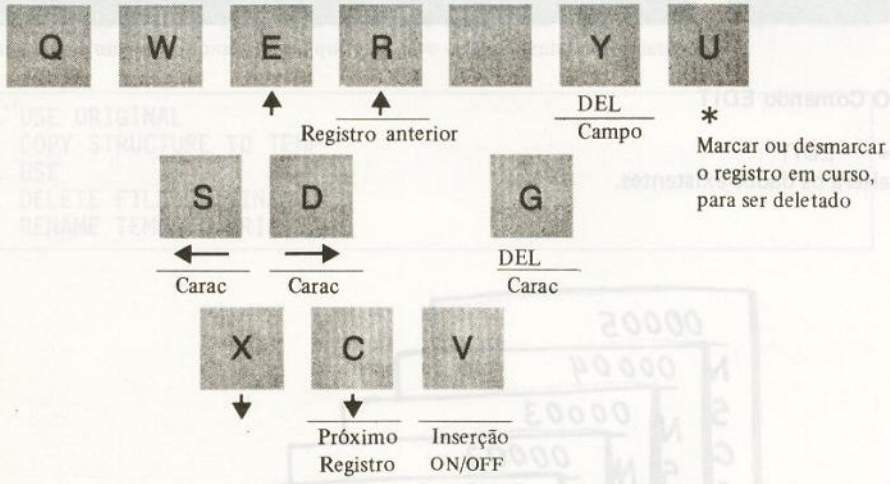


Sumário dos Comandos das TECLAS DE CONTROLE para a Entrada e Edição de Dados:

Modo EDIT

Pressionar CTRL com as seguintes teclas:

Abandonar ou preservar as alterações no registro em curso e trazer de volta o ponto.



Estes comandos de TECLAS DE CONTROLE requerem operações em full-screen. A maioria dos computadores permite tais operações e o funcionamento normal do dBASE é em FULL-SCREEN ON.

Se as operações em full-screen tiverem sido desligadas, religá-las através de:

```
. SET SCREEN ON
```

antes de se entrar nos modos APPEND, EDIT, INSERT ou BROWSE.

O modo EDIT exibe um único registro para a edição.

O registro é apresentado na tela na mesma forma do modo APPEND.

```
. EDIT 2
REGISTRO # 00002
NOME       :Luiz
SALARIO    :650.00:
CARGO      :Editor:
ANO        :1972:
```

Deslocar o cursor para os campos que se deseja alterar. Editar os dados contidos em tais campos mediante os comandos de edição nos teclados. Encerrar o modo EDIT pressionando < CTRL W > para escrever, no disco, as alterações efetuadas no arquivo do banco de dados e < CTRL Q > para abandonar as alterações realizadas naquele registro.

A palavra "REGISTRO" não é usada no modo EDIT.

A sintaxe do comando EDIT é diferente.

```
. EDIT 2
. DELETE RECORD 2
. DISPLAY RECORD 2.
```

Ao terminar a edição do registro, pode-se avançar para o seguinte pressionando <RETURN> até alcançar o fim do registro em curso, ou pressionando < CTRL C >. É possível voltar ao registro anterior através de < CTRL R >.

Querendo editar múltiplos registros, pode-se entrar com o modo EDIT sem o número do registro e o dBASE pedirá instruções sobre ele:

```
. EDIT
ENTRE COM O REGISTRO #:
```

Depois de editar o registro e de encerrar o modo EDIT através de < CTRL W > ou de < CTRL Q >, o dBASE pedirá novamente instruções sobre ele:


```
ENTRE COM O REGISTRO #:
```

Para editar um outro registro, entrar com um outro número de registro ou pressionar <RETURN> para encerrar o modo EDIT.

Um Modo Rápido de EDITAR Dados

Até aqui, foi descrita a edição no modo FULL-SCREEN (o que significa que os dados poderão ser posicionados em qualquer parte da tela). Caso o terminal em uso não trabalhe com operações em FULL-SCREEN ou se se deseja um modo mais direto e rápido de editar dados, prosseguir com a leitura.

Pode-se entrar, simultaneamente, com o número do registro, o nome do campo que se deseja alterar e com os novos dados.

Para tal, deve-se sair do modo full-screen através de:

```
. SET SCREEN OFF
```

Suponhamos que se deseja mudar o nome no registro nº 3 para JOSE. Neste caso, digitar:

```
. EDIT 3,NOME,JOSE
```

Querendo ver, com antecedência, o que está sendo alterado, inserir o número do registro e o nome do campo e aguardar a instrução do dBASE.

Pressionando < RETURN >, as alterações são transpostas.

```
. EDIT 3,NOME
NOME:  JOAO
CHANGE?
```

Há dois modos de editar os campos "C", mas apenas um para os campos "N" e "L".

Editando os Campos "C"

```
. EDIT 3,NOME
NOME:  JOAO
CHANGE?  ■
```

Aqui surge uma opção a ser feita: pode-se buscar o AO e trocá-lo por SE, ou entrar com os dados novos sobre os velhos.

Busca & Substituição

```
CHANGE?  AO
TO    SE
NOME:  JOSE
CHANGE?  ■
```

Pressionando < CTRL Y > e, depois, < RETURN >, salta-se a operação de busca e substituição.

Sobreposição

```
CHANGE?   CTRL Y
TO  JOSE
NOME:  JOSE
CHANGE?    ■
```

Editando um campo "N" ou "L"

Não existe edição através de busca e substituição nos campos N e L:

```
. EDIT 3,SALARIO
SALARIO:  850.00
TO:       ■
```


Digitar os novos dados ou pressionar < RETURN > para saltar.

Deslocando de um Registro a Outro

Pode-se permanecer no modo EDIT entrando com EDIT sozinho:

```
. EDIT
ENTRE RECORD, # FIELD  (# OR NAME), NOVO VALOR
RECORD  ■
```

Depois de cada edição, o dBASE perguntará por um novo conjunto de coordenadas:

```
RECORD: 3,NOME,JOSE
RECORD:  ■
```

Pressionando < RETURN >, encerra-se o modo EDIT.

Como antes, o dBASE perguntará pelos novos dados, caso eles não tenham sido incluídos.

```
. EDIT
ENTRE RECORD #, CAMPO (# OR NOME), NOVO VALOR
RECORD: 3,NOME
NOME:  JOAO
CHANGE?  AO
TO  SE
NOME:  JOSE
CHANGE?    ■
```

Pressionando < RETURN >, obtém-se, novamente, a pergunta relativa a RECORD:

Inserindo apenas um número de registro, veja pela tela a seguir (RECORD:5), o dBASE informa o nome do campo que foi anteriormente processado.
< RETURN >
É possível trocar o campo a qualquer momento.

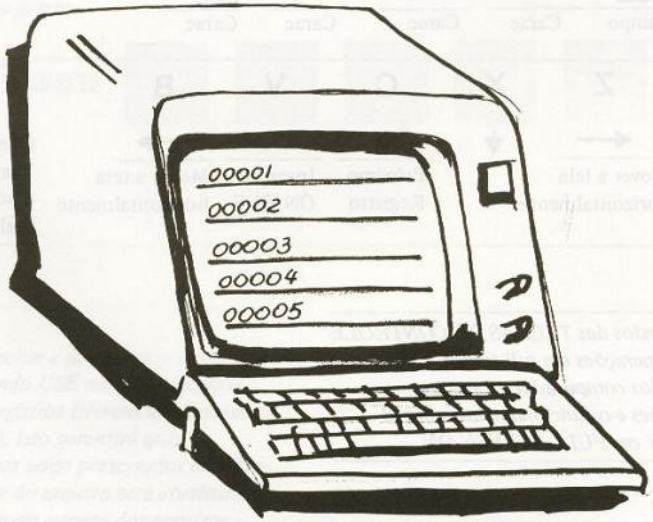
Os dados também podem ser editados através dos comandos BROWSE e REPLACE.

```
RECORD:  5
NOME:  Lucio
CHANGE?  o
TO  a
NOME:  Lucia
CHANGE?
RECORD: 4,SALARIO
SALARIO: 850.00
TO
```

O Comando BROWSE

- BROWSE

exibe todos os registros horizontalmente.



Exibe todos os registros horizontalmente, de modo a se poder "folheá-los".

Sumário dos Comandos das TECLAS DE CONTROLE para a Entrada e Edição de DADOS

Modo BROWSE

Pressionar CTRL com as seguintes teclas:

Abandonar ou Preservar as alterações no registro em andamento e trazer de volta o ponto.

Q	W	E	R		Y	U
		↑	↑		DEL	*
			Registro anterior		Campo	Marcar ou desmarcar o registro em curso para ser deletado.

A	S	D	F	G
←	←	→	→	DEL
Campo	Carac	Carac	Carac	Carac

Z	X	C	V	B
←	↓	↓		→
Mover a tela horizontalmente		Próximo Registro	Inserção ON/OFF	Mover a tela horizontalmente

Observação: Usar < CTRL Z > e < CTRL B > para deslocar horizontalmente.

Estes comandos das TECLAS DE CONTROLE requerem operações em full-screen. A maioria dos computadores permite tais operações e o funcionamento normal do dBASE é em FULL-SCREEN-ON.

Se as operações em full-screen tiverem sido desligadas, religá-las mediante:

```
. SET SCREEN ON
```

antes de entrar nos modos APPEND, EDIT, INSERT ou BROWSE.

O modo BROWSE é uma combinação dos comandos DISPLAY e EDIT.

Através dele, pode-se percorrer o registro em qualquer direção: para cima, para baixo, para a esquerda ou direita. O modo BROWSE apresenta os nomes dos campos como títulos de colunas.

O número de registro que aparece na parte superior da tela indica em qual linha de registro o cursor está localizado.

```
. USE PESSOAL
. BROWSE
REGISTRO # 00001
NOME---SALARIO--CARGO-------ANO
Paulo  250.00   Escriturario 1980
Luiz   650.00   Editor       1972
Joao   350.00   Escritor     1978
Pedro  850.00   Pres         1950
Lucio  700.00   Vice Pres    1960
```

Entrando com o comando BROWSE, a exibição em tela começa pelo registro em curso. Quando abrir um arquivo, como no exemplo precedente, o PONTEIRO DE REGISTRO se posiciona no registro nº 1.

Caso deseje começar pelo registro nº 4, digitar:

```
. 4
. BROWSE
```

```
REGISTRO  # 00004
NOME ----  SALARIO  --- CARGO -------  ANO
Pedro      850.00      Pres           1950
Lucio      700.00      Vice Pres      1960
```

A partir da versão 2.4, é possível especificar quais os campos que deverão ser percorridos pelo comando BROWSE:


```
. 1
. BROWSE FIELD NOME,SALARIO
```

ATENÇÃO:
Não esquecer das vírgulas.

Editando:

Ao mesmo tempo em que se percorre o arquivo através do comando BROWSE, pode-se deletar ou cancelar a marca de deleção de registros através de < CTRL U >. Pode-se, também, inserir novos dados, deletar ou escrever por cima, em qualquer lugar onde o cursor estiver localizado.

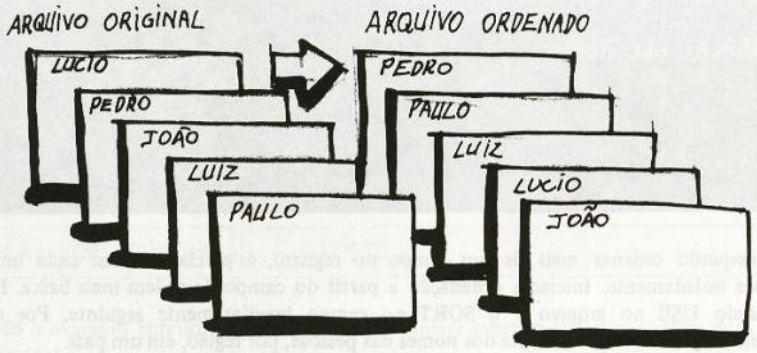
CAPÍTULO 9

DISPONDO OS DADOS EM OUTRA SEQÜÊNCIA

O Comando SORT

- SORT

cria um novo arquivo com
registros ordenados de modo diferente.



Ordenando os registros do arquivo em um campo específico

O comando SORT cria um novo arquivo e organiza a ordem dos registros segundo uma nova seqüência. Entra-se com o nome do campo pelo qual os registros serão ordenados e com o nome do novo arquivo que está sendo criado.

Não se esqueça de abrir o novo arquivo criado.

```
. USE PESSOAL
. SORT ON NOME TO ARQNOME
. USE ARQNOME
. DISPLAY ALL NOME
00001  Joao
00002  Lucio
00003  Luiz
00004  Paulo
00005  Pedro
```

A ordenação é feita, geralmente, da ordem menor para a maior (ASCENDENTE). Inverte-se a ordem acrescentando DESCENDING.

Usando-se o mesmo nome de arquivo, sobregrava-se o arquivo original.

```
. USE PESSOAL
. SORT ON NOME TO ARQNOME DESCENDING
. USE ARQNOME
. DISPLAY ALL NOME
00001  Pedro
00002  Paulo
00003  Luiz
00004  Lucio
00005  Joao
```

Desejando ordenar mais de um campo no registro, é preciso realizar cada uma dessas operações isoladamente. Iniciar a ordenação a partir do campo de ordem mais baixa. Empregar o comando USE no arquivo e o SORT no campo imediatamente seguinte. Por exemplo, suponhamos que se deseja uma lista dos nomes das pessoas, por região, em um país.

```
. SORT ON NOME TO PRIMEIRO
. USE PRIMEIRO
. SORT ON REGIAO TO SEGUNDO
. USE SEGUNDO
. SORT ON PAIS TO FINAL
. USE FINAL
```

Querendo ordenar apenas parte do arquivo, deve-se criar, em primeiro lugar, um arquivo parcial através do comando COPY:

```
. COPY TO PARCIAL FOR DATA > '810630'
. USE PARCIAL
. SORT ON campo TO arquivo
```

A ordenação do dBASE pode levar algum tempo, no caso de grandes arquivos. É possível, porém, economizar tempo indexando-se o arquivo (através do comando INDEX), ao invés de ordená-lo, especialmente se estiver ornando múltiplos campos.

Por exemplo:

```
. USE PESSOAL
. INDEX ON NOME TO NOMEINDX
. DISPLAY ALL NOME
00003  Joao
00005  Lucio
00002  Luiz
00001  Paulo
00004  Pedro
```

Notar que os registros não estão em seqüência nominal, e sim segundo a seqüência de ÍNDICE.

Ao ordenar cria-se, no disco, um novo arquivo, cujo tamanho é idêntico ao do original não ordenado. A indexação ocupa menos espaço no disco (ver Capítulo 14).

CAPÍTULO 10

EXTRAINDO DADOS E INFORMAÇÕES

O Comando DISPLAY

- DISPLAY

exibe um único registro, o arquivo inteiro ou registros selecionados de um arquivo.

Os comandos DISPLAY ALL e DISPLAY FOR param e aguardam a cada quinze registros. Pressionando qualquer tecla, dá-se continuidade à exibição.

< ESC > cancela
o comando DISPLAY

< CTRL S > Impede, temporariamente, que a exibição continue. Qualquer tecla dá prosseguimento ao processo.

O comando LIST, por outro lado, não pára até alcançar o final do arquivo. Este comando torna-se, então, muito útil quando se deseja imprimir todos os registros, sem que eles apareçam na tela. O comando LIST funciona de modo idêntico ao DISPLAY, exceto pelo fato de LIST ser sempre uma condição total (ALL), a menos que se especifique o contrário.

O comando DISPLAY exibe um registro único, o arquivo inteiro ou registros selecionados de um arquivo. O comando DISPLAY ALL começa pelo registro nº 1 e exibe todos os registros contidos no arquivo.

```
. USE PESSOAL
. DISPLAY ALL
00001  Paulo      250.00  Escriturario     1980
00002  Luiz       650.00  Editor           1972
00003  Joao       350.00  Escritor         1978
00004  Pedro      850.00  Pres             1950
00005  Lucio      700.00  Vice Pres        1960
```

Pode-se Imprimir o que se Vê na Tela

Se houver uma impressora em linha com o computador, digitar conforme a tela a seguir.

Quando o comando "PRINT" está em operação, os dados de saída vão para a impressora e para a tela.

```
. SET PRINT ON
. DISPLAY ALL
00001  Paulo      250.00  Escriturario     1980
00002  Luiz       650.00  Editor           1972
00003  Joao       350.00  Escritor         1978
00004  Pedro      850.00  Pres             1950
00005  Lucio      700.00  Vice Pres        1960
. SET PRINT OFF
```


Usando CP/M, pode-se empregar < CTRL P > para acionar e desacionar a impressora.

É possível exibir um registro específico através de seu número.

```
. DISPLAY RECORD 2
00002  Luiz       650.00  Editor           1972
```

Observe o que acontece quando pressionar apenas DISPLAY:

```
. DISPLAY
00002  Luiz       650.00  Editor           1972
```

Sob esta condição de entrada, o comando DISPLAY continuará a exibir o registro em curso até reposicionar o PONTEIRO DE REGISTRO do dBASE. Por exemplo:

```
. 3
. DISPLAY
00003  Joao       350.00  Escritor         1978
```

Reportando-se a um registro que não conste do arquivo, é recebida a seguinte mensagem:

```
. DISPLAY RECORD 13706
REGISTRO FORA DE ALCANCE
```

Elimina-se o número do registro, à esquerda da tela, através de OFF.

Notar que os números dos registros não aparecem.

```
. DISPLAY ALL OFF
Paulo      250.00  Escriturario     1980
Luiz       650.00  Editor           1972
Joao       350.00  Escritor         1978
Pedro      850.00  Pres             1950
Lucio      700.00  Vice Pres        1960
```

Acrescentando um ou mais nomes de campos depois do comando DISPLAY, apenas tais campos serão exibidos.

Para exibir apenas os campos desejados:

```
. DISPLAY ALL NOME
00001  Paulo
00002  Luiz
00003  Joao
00004  Pedro
00005  Lucio
```

```
. DISPLAY ALL NOME SALARIO
00001  Paulo  250.00
00002  Luiz   650.00
00003  Joao   350.00
00004  Pedro  850.00
00005  Lucio  700.00
```

```
. DISPLAY ALL NOME OFF
Paulo
Luiz
Joao
Pedro
Lucio
```

Pode-se exibir qualquer parte de um campo pelo uso da função $ (ver Capítulo 18 para maiores detalhes).

Nome do campo
Posição de partida
Número de caracteres

```
. DISPLAY ALL    $(NOME,1,2)
Pa
Lu
Jo
Pe
Lu
```



```
. DISPLAY ALL    $(NOME,2,2)
au
ui
oa
ed
uc
```

Não esquecer de separar, por vírgulas, estas expressões parciais de campos.

```
. DISPLAY ALL   $(NOME,1,2),$(CARGO,1,2)
pa   Es
lu   Ed
jo   Es
pe   Pr
lu   Vi
```

Usando OFF e o símbolo de *cardinal* (#), pode-se exibir o número em qualquer posição da linha.

```
. DISPLAY CARGO NOME, # OFF
Escriturario      Paulo          1
```

O símbolo # deve ser separado dos nomes dos campos por intermédio de vírgulas.

```
. DISPLAY CARGO NOME, # , SALARIO OFF
Escriturario    Paulo  1   250.00
```

Os Dados Exibidos podem ser Calculados

Este tempo de serviço é calculado na memória e não é preservado.

```
. DISPLAY NOME ANO 1984-ANO
00001    Paulo  1980     4
. DISPLAY NOME SALARIO SALARIO * 1.1
00001    Paulo    250.00   275.000
```

E se houvesse um aumento de 10% no salário?

Como 1,1 vezes 250.00 resulta em três casas decimais, pode-se usar a função STR para filtrar a terceira casa decimal:

```
. DISPLAY NOME SALARIO STR (SALARIO*1.1,7,2)
00001    Paulo    250.00     275.00
```

Exibindo Registros do Meio do Arquivo:

Os comandos DISPLAY ALL (e LIST) sempre partem do registro número 1.

00001	Paulo	Escriturário	1980	◀ PONTEIRO DE REGISTRO
00002	Luiz	Editor	1972	
00003	João	Escritor	1978	
00004	Pedro	Pres	1950	
00005	Lucio	Vice Pres	1960	

Desejando iniciar mais abaixo, no arquivo, posicionar o ponteiro de registro no registro escolhido. Inserir, apenas, o número do registro:

```
. 3
```

Isto posicionará o PONTEIRO DE REGISTRO no registro número 3.

00001	Paulo	Escriturário	1980	
00002	Luiz	Editor	1972	
00003	João	Escritor	1978	◀ PONTEIRO DE REGISTRO
00004	Pedro	Pres	1950	
00005	Lucio	Vice Pres	1960	

O ponteiro de registro está, agora, posicionado no número do registro com que se entrou.

Para exibir o número do registro para o qual o PONTEIRO DE REGISTRO está apontado, digita-se:

```
.? #
```

Tendo movido o PONTEIRO DE REGISTRO, pode-se usar o comando DISPLAY NEXT para exibir os registros seguintes. Por exemplo:

```
. 3
DISPLAY NEXT 2
00003   Joao        350.00      Escritor      1978
00004   Pedro       850.00      Pres          1950
```

Querendo exibir os registros situados do meio ao fim do arquivo, deve-se escolher um número para vir depois de NEXT, suficientemente grande para alcançar o final do arquivo.

```
. 3
DISPLAY NEXT 100
00003   Joao        350.00      Escritor      1978
00004   Pedro       850.00      Pres          1950
00005   Lucio       700.00      Vice Pres     1960
```

```
. 3
. DISPLAY NEXT 100 NOME
00003    Joao
00004    Pedro
00005    Lucio
```

Buscando os Dados

Os computadores buscam os dados comparando-os com outros dados. Assim, ao se buscarem dados em dBASE, é preciso fornecer dados de tal forma que a comparação possa ser realizada e também determinar se os dados desejados são iguais, maiores ou menores do que os apresentados. Por exemplo, o pedido: "Mostre-me quem recebe mais de 700 dólares" deverá ser traduzido para:

```
. DISPLAY FOR SALARIO > 700
```

Os símbolos para comparar, no teclado, são:

>	MAIOR DO QUE
> =	MAIOR OU IGUAL A
<	MENOR DO QUE
< =	MENOR QUE OU IGUAL A
=	IGUAL A
# ou < >	DIFERENTE DE

O comando DISPLAY FOR exibe todos os registros que atendam à condição FOR:

```
. DISPLAY FOR ANO > = 1980
00001     Paulo    250.00          Escriturario      1980
. DISPLAY FOR ANO < 1970
00004     Pedro    850.00          Pres              1950
00005     Lucio    700.00          Vice Pres         1960
. DISPLAY FOR NOME  = "Luiz"
00002     Luiz     650.00          Editor            1972
. DISPLAY NOME FOR ANO <1970 OFF
Pedro
Lucio
```

É possível comparar dados dentro do próprio registro. Suponhamos que se disponha, em um mesmo registro, do preço de custo de um artigo e de seu valor corrente. Para buscar todos os registros onde o valor corrente é superior ao de custo, digita-se:

```
DISPLAY FOR CORRENTVAL > = CUSTO
```

Usando o exemplo anterior, imaginemos que se queira saber quais artigos valem duas vezes seu preço original:

```
DISPLAY FOR CORRENTVAL > = CUSTO * 2
```

Em primeiro lugar multiplica-se CUSTO por 2 sendo esse valor, depois, comparado.

Para saber que artigos valem mais 10% do que seu preço de custo, digitar:

```
. DISPLAY FOR CORRENTAVAL > = CUSTO * 1.1
```

Aqui, multiplica-se por 110%.

Desejando proceder corretamente sob o ponto de vista econômico, levando-se em conta que o valor corrente está inflacionado, poder-se-á reduzi-lo em 15%.

```
. DISPLAY FOR CORRENTVAL - CORRENTVAL * .15 > = CUSTO * 2
```

(2) (1) (3) (1)

(1) As multiplicações e divisões são efetuadas em primeiro lugar;
(2) a seguir vêm as adições e subtrações e, por último,
(3) uma comparação.

Verificações Exatas:

Note que:

```
. DISPLAY FOR CARGO = 'Editor'
. DISPLAY FOR CARGO = 'Edi'
```

apresentarão, *todos*, o mesmo resultado:

```
00002     Luiz     650.00          Editor            1972
```

Introduzindo o comando SET EXACT ON, veja o que acontece:

```
. SET EXACT ON
. DISPLAY FOR CARGO = 'E'
.
```

Obtém apenas o ponto. Não há verificação!

Utilizando o comando SET EXACT ON, o conteúdo do campo inteiro deve ser exatamente verificado. Por outro lado, não se desejando mais que esta condição seja atendida, usa-se o comando SET EXACT OFF. Nota-se, também, que uma verificação exata poderá ser necessária se ocorrerem prefixos semelhantes nos dados. Por exemplo:

Editor
Editor assistente

AND/OR/NOT

Busca lógica de dados atendendo a uma ou mais condições.

Lógica Booleana

Na metade do século XIX, o matemático e lógico inglês George Boole estabeleceu as operações lógicas envolvidas nas decisões. Ele mostrou que uma decisão baseia-se na avaliação das condições E, OU, NÃO (AND, OR, NOT). A lógica Booleana é a base para o projeto de todos os circuitos dos computadores digitais.

Utilizando adequadamente as três pequenas palavras E, OU e NÃO, obtém-se qualquer combinação possível das condições existentes nos dados dos registros.

.AND.

.AND. é usado na busca de dados que atendam a duas ou mais condições.

IMPORTANTE: .AND. é necessário para selecionar um conjunto de registros.

```
. DISPLAY FOR SALARIO > 700 .AND. ANO > 1948
00004     Pedro    850.00    Pres       1950
. DISPLAY FOR NOME < 'M' .AND.  ANO > 1975
00003     Joao     350.00    Escritor      1978
. DISPLAY FOR SALARIO <300 .AND. CARGO = 'E'
00001     Paulo    250.00    Escriturario     1980
```

```
. DISPLAY FOR ANO > 1971 .AND. ANO < 1976
00002     Luiz   650.00    Editor  1972
```


Não há limite para o uso do número de .AND.s na busca, a menos do comprimento máximo de 254 caracteres de uma linha de comando. Todos os três, .AND., .OR. e .NOT. requerem um ponto antes e depois da palavra.

.OR.

.OR. é usado na busca de dados que atendam a, pelo menos, uma dentre duas ou mais condições.

No caso do arquivo (exemplo, PESSOAL) todos os registros aparecem porque, ao se usar .OR., caso o registro não atenda à primeira condição, ainda poderá atender a uma segunda ou terceira condições.

```
. DISPLAY FOR SALARIO > 700 .OR. ANO > 1948
00001  Paulo    250.00    Escriturario     1980
00002  Luiz     650.00    Editor           1972
00003  Joao     350.00    Escritor         1978
00004  Pedro    850.00    Pres             1950
00005  Lucio    700.00    Vice Pres        1960
```

É importante notar que o AND (= e) da língua inglesa e o .AND. Booleano não têm o mesmo significado. Ou seja, a solicitação feita em inglês "Let's get a listing of all the writers *and* editors in our company" (Providencie uma listagem de todos os escritores e editores da nossa companhia) é, na verdade, uma condição OU, isto é, um .OR. Booleano:

```
. DISPLAY FOR CARGO = 'Escritor'  .OR. CARGO = 'Editor'
```

O .AND. Booleano seria correto apenas se as pessoas em questão fossem, ao mesmo tempo, escritores e editores. Se tal acontecesse e se ambos os cargos estivessem, de fato, armazenados em um campo para cargo, a busca dos nomes desses profissionais seria feita através de $ (ver VARREDURA DE CAMPO, na página 98).

```
. DISPLAY FOR 'Escritor' $CARGO .AND. 'Editor' $CARGO
```

.NOT.

.NOT. é usado na busca de dados que são o *oposto* da condição NOT (Não):

```
. DISPLAY FOR .NOT. ANO > 1970
00004     Pedro     850.00      Pres          1950
00005     Lucio     700.00      Vice Pres     1960
```

```
. DISPLAY FOR .NOT.NOME < 'M'
00001     Paulo     250.00      Escriturario      1980
00004     Pedro     850.00      Pres              1950
```

Nos exemplos acima, pode-se evitar o uso do .NOT. alterando a expressão:

```
. DISPLAY FOR ANO < 1971
. DISPLAY FOR NOME > 'L'
```

.NOT. é necessário para se exprimir, através de dados lógicos, uma condição "diferente de":

```
. DISPLAY FOR .NOT. SR:CIDADAO
. DISPLAY FOR .NOT. MBA
```

Campos "L" de 1 byte.

Combinando .AND.s com .OR.s

Lembrar-se de que os dados de caracteres vêm entre aspas. Os dados numéricos, não.

Vamos supor que se deseje buscar os Vice-Presidentes que começaram a trabalhar a partir de 1955 ou os que ganhem menos de $750.00. Inserindo-se:

```
. DISPLAY FOR CARGO = 'Vice Pres' .AND. SALARIO < 750;
.OR. ANO > 1955
00001     Paulo     250.00      Escriturario      1980
00002     Luiz      650.00      Editor            1972
00003     Joao      350.00      Escritor          1978
00005     Lucio     700.00      Vice Pres         1960
```


Notar o ponto e vírgula interrompendo a linha de comando.

Acrescentando parênteses ao comando, obtém-se o resultado correto.

Não esquecer de tentar acrescentar os parênteses caso o complexo critério de busca não produza os resultados desejados.

```
. DISPLAY FOR CARGO = 'Vice Pres' .AND.(SALARIO<750 .OR.;
ANO > 1955)
```

Tudo que estiver entre parênteses será tratado como um único grupo. O comando, agora, funcionará como: Vice Presidente cargo .AND. (uma ou outra dessas condições).

Varredura de Campo

É a capacidade de VARREDURA DE CAMPO que permite ao dBASE trabalhar tanto com palavras (texto), quanto com dados. Podem-se criar grandes campos (de até 254 caracteres), preenchê-los com frases e buscar quaisquer palavras que tenham sido gravadas.

Mediante o uso do $ pode-se buscar os dados no meio de um campo de caracteres. Por exemplo:

```
. DISPLAY FOR 'Pres' $CARGO
00004     Pedro     850.00      Pres          1950
00005     Lucio     700.00      Vice Pres     1960
. DISPLAY FOR "rito" $CARGO
00003     Joao      350.00      Escritor      1978
. DISPLAY NOME FOR 'a' $NOME OFF
Paulo
Joao
```

Lembre-se de que OFF significa desligar a exibição do número do registro.

O $ significa "dentro dos limites de" e ordena que o dBASE compare os dados entre aspas com cada posição possível no campo. Pode-se interpretar esta expressão como:

"ISTO" dentro "DAQUILO".

Os exemplos seguintes ilustram o armazenamento de dados e de palavras em um mesmo registro:

Receita Culinária

```
CAMPO       NOME, TIPO, TAMANHO

001         RECEITA,C,15
002         CATEGORIA,C,10
003         INGRED,C,200
```

Pode-se buscar receitas de sopa que levem cenouras:

```
DISPLAY FOR CATEGORIA = 'Sopa' .AND.'Cenoura' $INGRED
```

Selecionaria o seguinte registro:

Receita	Categoria	Ingred
Sopa de Vegetais	Sopa	Água Cenoura Ervilhas Aipo
"Dados"	"Dados"	"Palavras" (Texto)

Arquivo Pessoal

```
CAMPO       NOME, TIPO, TAMANHO

001         NOME,C,25
002         REGIAO,C,10
003         COMENT,C,250
```

Pergunta-se: Quem fala sueco na região setentrional?

```
DISPLAY FOR REGIAO = 'Norte' .AND.'Sueco' $COMENT
```


O registro seguinte seria selecionado:

Nome	Região	Coment
Freedman, Karen	Norte	Gosta de Música Rock, Fala Sueco
"Dados"	"Dados"	"Palavras" (Texto)

Imóveis:

```
CAMPO       NOME, TIPO, TAMANHO
001         TITULO,C,10
002         PRECO,N,7
003         DESCRICAO,C,200
```

Supondo que todas as características da casa estejam uniformemente codificadas no campo destinado à DESCRIÇÃO, pode-se buscar uma casa de três quartos, com lareira e piscina, de preço inferior a $175,000, do seguinte modo:

```
  DISPLAY FOR PRECO < 175000 .AND.;
"3QUAR" $DESCRICAO .AND.;
"LAR" $DESCRICAO .AND.;
"PIS"  $DESCRICAO
```

Título	Preço	Descrição
Schiller House	150.000	AQUEC LAR 3 QUAR PIS Sauna
"Dados"	"Dados"	"Palavras" (Texto)

Todas as Condições FOR Operam do Mesmo Modo

As condições DISPLAY FOR discutidas neste capítulo valem para todos os comandos que usam a opção FOR, tais como:

```
COPY TO_____FOR
DELETE FOR
SUM ____ FOR
```

Isto inclui os .AND.s, .OR.s e .NOT.s e, também, a VARREDURA DE CAMPO.

As capacidades de busca e de seleção proporcionam uma flexibilidade completa na manipulação dos dados.

Display Structure

O comando DISPLAY STRUCTURE permite o exame da estrutura do registro do arquivo em uso. Veja exemplo a seguir.

Se o registro contiver um grande número de campos pode-se impedir, temporariamente, que seus nomes fujam do campo da tela, pressionando < CTRL S >. A exibição de novas linhas é reiniciada ao se pressionar qualquer tecla.

Drive do Disco Tipo de Arquivo

```
. USE PESSOAL
. DISPLAY STRUCTURE
ESTRUTURA PARA O ARQUIVO:        A: PESSOAL      .DBF
NUMERO DE REGISTROS:             00005
DATA DA ULTIMA ATUALIZACAO:      DD/MM/AA
BANCO DE DADOS DE USO PRIMARIO
CAMPO   NOME        TIPO      TAMANHO      DEC
001     NOME        C         007
002     SALARIO     N         007          002
003     CARGO       C         015
004     ANO         N         004
**TOTAL **                    00034
```

O total de bytes inclui 1 byte reservado para o código de Deleção (*).

Display Status

O comando DISPLAY STATUS permite o exame da situação em curso do dBASE.

Pressionar qualquer tecla para prosseguir.

** = Coberto no Nível 1 (ver Índice dos Comandos do Nível 1). O resto poderá ser encontrado no Capítulo 20 (Sumário dos Comandos do Nível 2).*

```
. DISPLAY STATUS
BANCO DE DADOS SELECIONADOS     A: PESSOAL       .DBF
BANCO DE DADOS DE USO PRIMARIO

ESPERANDO

 DATA DE HOJE           --DD/MM/AA
 DEFAULT DISK DRIVE     --A:
*ALTERNATE  - OFF       BELL          - ON
*CARRY      - OFF       COLON         - ON
*CONFIRM    - OFF       CONSOLE       - ON
 DEBUG      - OFF      *DELETE        - OFF
 ECHO       - OFF      *EJECT         - ON
 ESCAPE     - ON       *EXACT         - OFF
 INTENSITY  - ON       *LINKAGE       - OFF
*PRINT      - OFF       RAW           - OFF
 STEP       - OFF       TALK          - ON
```

Display Files

O comando DISPLAY FILES exibe os nomes dos arquivos do diretório do disco. Para se ver os nomes dos arquivos de dados do dBASE, digitar:

```
. DISPLAY FILES
```

Querendo os nomes dos arquivos no drive B, digitar:

```
. DISPLAY FILES ON B
```

Através do comando DISPLAY FILES LIKE, nome do arquivo.tipo do arquivo pode-se selecionar diferentes tipos de arquivos. Por exemplo:

```
. DISPLAY FILES LIKE *.FRM
```

Exibirá os nomes de quaisquer arquivos com formato de relatório do disco.

Os nomes dos arquivos, no disco, são formados por duas partes.

A primeira parte corresponde ao nome que se dá ao arquivo.

A segunda parte é um tipo de arquivo em geral, criado automaticamente pelo software.

nome do arquivo . tipo do arquivo

As duas partes são separadas por um ponto.

O dBASE associa os seguintes tipos aos seus arquivos:

CÓDIGO DO TIPO	TIPO DO ARQUIVO	GERADO POR
DBF	arquivo de banco de dados	create, copy, sort
FRM	formato de relatório	report
NDX	índice	index
TXT	arquivo de texto	copy, set alternate
CMD ou PRG	arquivo de comando	modify command
FMT	arquivo de formato	ou qualquer editor de texto

Selecionando tipos e nomes de arquivos tendo em vista a exibição, pode-se usar * e ? como caracteres-chave para verificação. Por exemplo:

```
. DISPLAY FILES LIKE *.*
```

Todos os arquivos no diretório serão, pois, exibidos. Como * significa qualquer combinação de caracteres, tudo estará dentro de *. *. O símbolo ? é um caractere-chave composto de um único caractere. Querendo exibir os nomes de todos os arquivos TXT iniciados por N, digitar:


```
. DISPLAY FILES LIKE N*.TXT
```

Porém, querendo todos os arquivos TXT que contenham um N na segunda posição do nome, digitar:

```
. DISPLAY FILES LIKE ?N*.TXT
```

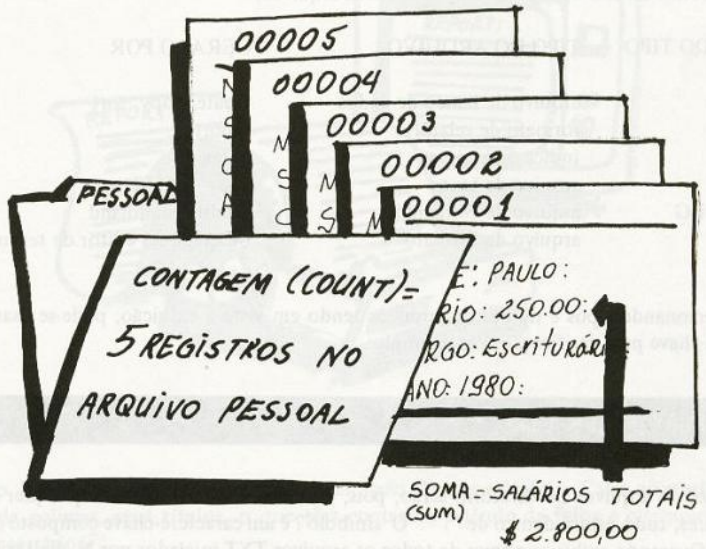
Os comandos COUNT e SUM

- COUNT

conta o número de registro no arquivo.

- SUM

totaliza o valor numérico em campos específicos.



O comando COUNT conta o número de registros no arquivo. Se este contiver 314 registros, então:

```
. COUNT
CONTAGEM = 00314
```

Lembre-se de que é possível obter uma contagem total atualizada de registros, através do comando DISPLAY STRUCTURE.

Para contar registros selecionados (através do COUNT), digitar:

```
. COUNT FOR PAIS = "USA"
```

Pode-se armazenar o resultado da contagem (feita mediante COUNT) em uma VARIÁVEL DE MEMÓRIA, através de:

```
. COUNT TO QUANTOS
CONTAGEM = 00314
```

No caso de se querer ver o que há em QUANTOS, digitar:

```
. ? QUANTOS
```

O comando SUM dá o total de até cinco campos numéricos, de uma só vez:

```
. SUM QUANTIDADE, CUSTO, QUANTIDADE * CUSTO
  36          246.50        6 721.30
```

Aqui, o * significa multiplicação.

Totalizam-se os registros selecionados (usando o comando SUM), através de:

```
. SUM SALARIO FOR SALARIO > 500
```

Os resultados obtidos com o comando SUM podem ser armazenados em variáveis de memória.

Veja o Capítulo 16 para detalhes de variáveis de memória.

```
. SUM QUANTIDADE, CUSTO, QUANTIDADE * CUSTO TO Q, C, QC
```

As variáveis de memória podem ser exibidas digitando-se:

```
. ? Q, C, QC
  36          246.50        6724.30
```

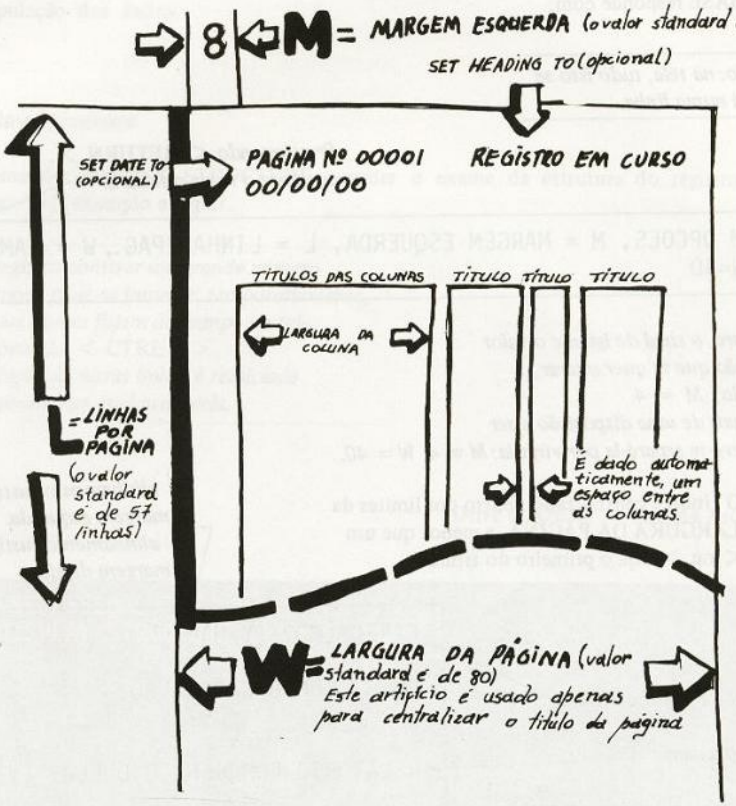
O comando REPORT

- REPORT

permite a definição e emissão de um relatório.

O comando REPORT permite a definição e emissão de um relatório. Cabe ao usuário definir o número de colunas, seus títulos, o que elas conterão, o título da folha e quaisquer totais ou subtotais necessários.

Forma de Relatório do dBASE



Sempre que se atribui um nome novo à FORMA (FORM), que vem acompanhando o comando REPORT, o usuário se verá frente a uma série de descrições que servirão para definir o relatório.

As descrições serão automaticamente preservadas em disco, em um arquivo do tipo REPORT FORM, de tal forma que da próxima vez em que se inserir REPORT FORM nome, o dBASE usará, também automaticamente, as definições relativas ao arquivo daquela FORMA. (O arquivo REPORT FORM aparecerá no disco, através da extensão .FRM.)

Por exemplo:

```
REPORT FORM NOVA
```


As regras que regem um nome de arquivo de FORMA (FORM) são as mesmas que regem os nomes dos arquivos de dados.

O dBASE responde com:

Observação: na tela, tudo isto se constituirá numa linha.

Pressionando < RETURN >, *aceita-se o valor standard.*

```
ENTRAR COM OPCOES, M = MARGEM ESQUERDA, L = LINHAS/PAG., W = TAMANHO PAG
M=4,L=50,W=40
```

Inserir a letra, o sinal de igual e o valor da disposição que se quer alterar, por exemplo, M = 4. Havendo mais de uma disposição a ser alterada, deve-se separá-la por vírgula: M = 4, W = 40.

O título é centralizado dentro dos limites da LARGURA DA PÁGINA, a menos que um < ou > seja o primeiro do título.

< alinhamento justificado à margem esquerda.
> alinhamento justificado à margem direita.

OPÇÕES
Respondendo Y, aqui é exibido.

A escolha do usuário.

Inserir Y (SIM), se desejar um total em qualquer das colunas.

Aqui, um Y(SIM) requer que se dê uma resposta às perguntas:

1. *Nome do campo.*
   Sempre que os dados deste campo forem alterados obtém-se um subtotal.
2. *Y (SIM) elimina todos os dados, exceto o campo de subtotais e os totais.*

```
CABECALHO? (Y/N)
ENTRE COM O CABECALHO:

RELATORIO COM ESPACO DUPLO? (Y/N)

DESEJA TOTAIS? (Y/N)

DESEJA RELATORIO RESUMIDO? (Y/N)

ENTRE COM CAMPO DE SUBTOTAL:


RELATORIO RESUMIDO? (Y/N)
```

IMPORTANTE
Em caso de erro ...

Não se pode alterar uma seleção uma vez que tenhamos avançado para o registro seguinte. Cometendo um erro e querendo iniciar novamente a descrição do relatório, pressione-se < ESC > para trazer o ponto de volta. Deletar a descrição da forma, digitando:

```
DELETE nome da forma.FRM
```

e começando de novo.

3. *Y (SIM) faz o papel saltar para a página seguinte após cada subtotal.*
4. *Opcional. Descreve a categoria dos dados no campo do subtotal.*

```
PULE A PAGINA APOS SUBTOTAL? (Y/N)
ENTRE COM CABECALHO NO SUBTOTAL:
COL     TAMANHO, CONTEUDO

001
ENTRE COM CABECALHO
```

O que está impresso na coluna

Largura da coluna

Usando < ou > como o primeiro caractere do título, faz-se o seu alinhamento em relação à margem esquerda ou direita.

Encerra-se a definição do relatório pressionando-se < RETURN >.
O relatório será exibido na tela.
(Veja a seguir.)


```
COL     TAMANHO, CONTEUDOS
008     ■
```

Normalmente, a largura da coluna será igual à do campo e os conteúdos incluirão o nome do campo a ser impresso. Por exemplo:

Observação: Tornando as larguras das colunas do RELATÓRIO maiores do que as dos campos, obtêm-se colunas impressas separadas entre si.

```
7,NOME
15,CARGO
```

Pode-se efetuar cálculos nas colunas:

```
COL          TAMANHO, CONTEUDO
005          10, PRECO - TOTPAGO
ENTRE COM CABECALHO: Balanco
COL          TAMANHO, CONTEUDO
005          10, QUANTIDADE * CUSTO
ENTRE COM CABECALHO: Custo Total
```

É possível fazer cálculos com os dados dos registros em relação a um total obtido fora do relatório. Por exemplo: suponhamos que se deseje imprimir uma porcentagem do salário de cada pessoa em relação ao salário médio. Calcula-se, em primeiro lugar, o salário médio:

```
. SUM SALARIO TO SALTOT
. COUNT TO CONTOT
. STORE SALTOT/CONTOT TO SALMED
```

A coluna do relatório terá o seguinte aspecto:

```
COL     TAMANHO, CONTEUDO
007     7,SALARIO/SALMED * 100
ENTRE COM CABECALHO: Percentual da Media
```

Se a largura da coluna for menor que a dos dados, os dados "C" serão levados para a linha seguinte. Porém, aparecerão asteriscos (***) impressos nas colunas "N".

As descrições do REPORT FORM poderão ser alteradas através do editor de texto do dBASE, desde que se tenha o cuidado de manter o alinhamento correto das linhas das descrições. Talvez seja melhor deletar o arquivo REPORT FORM e redescrever o relatório, se houver necessidade de modificação deste último.

Campo "C"

A	L	L	I	S	O	N

Campo "N"

1	8	7	6	3	4	.	2	1

COLUNA

A L L I
S O N
* * * *

Isto significa que um campo longo de texto poderá ser exibido sob a forma de um parágrafo.

É como o dBASE imprime os dados

COLUNA

É como o dBASE imprime os dados

É sempre útil imprimir o número do registro no relatório. Deve-se, então, reservar espaço suficiente para conter o maior dos números de registros disponíveis.

Combinando e reunindo os registros. O comando JOIN é muito útil na junção de dados visando um relatório (ver Capítulo 17).


```
COL     TAMANHO, CONTEUDO
001     3, #
```

Para imprimir o relatório, digitar:

```
. REPORT FORM NOVA TO PRINT
```

A seleção dos registros a serem impressos pode ser feita através de FOR:

```
. REPORT FORM NOVA TO PRINT FOR SALARIO>500
```

Começa-se a impressão a partir da metade do arquivo. Suponhamos que se deseje imprimir os arquivos compreendidos entre o nº 97 e o nº 115, inclusive:

```
. 97
. REPORT FORM NOVA NEXT 19 TO PRINT
```

Ao invés de se imprimir o relatório, pode-se criar uma cópia do disco a ser usada com um processador de palavras:

```
. SET ALTERNATE TO COPIADSK
. SET ALTERNATE ON
. REPORT FORM NOVA
```

Não se tendo certeza de que a impressora esteja ajustada de forma adequada para o início da página, pode-se avançar o papel para a parte superior da página seguinte, usando o próprio teclado do computador.

```
. SET PRINT ON
. EJECT
. SET PRINT OFF
```

Caso não se deseje que o dBASE inicie automaticamente o relatório na parte superior da página seguinte, pode-se alterar, também, esta condição digitando:

```
. SET EJECT OFF
```

Lembre-se de que, para alterar o título de cada relatório, é necessário digitar:

```
· SET HEADING TO titulo
```

Não há valor standard.

Para alterar a margem esquerda, inserir:

```
· SET MARGIN TO margem esquerda
```

O valor standard é 8
(ver página 126).

PARTE 4

MANTENDO OS ARQUIVOS

CAPÍTULO 11

FUNÇÕES UTILITÁRIAS E DE MANUTENÇÃO

Usando o Comando COPY

O comando COPY serve para criar cópias de reserva dos arquivos de dados (arquivos.DBF). Pode-se copiar um arquivo de um disco para outro ou fazer uma cópia do arquivo no mesmo disco, atribuindo-lhe um outro nome. Seguem as várias aplicações do comando COPY.

1. Trabalhando no drive A e querendo fazer uma cópia de reserva no drive B, digitar:

```
. USE PESSOAL
. COPY TO B:PESSOAL
```

2. Por outro lado, trabalhando no drive B e querendo copiar em A, digitar:

```
. USE PESSOAL
. COPY TO A:PESSOAL
```

3. Para copiar um arquivo de B para A, enquanto estiver no drive A, digitar:

```
. USE B:PESSOAL
. COPY TO PESSOAL
```

4. Pode-se atribuir um outro nome a um arquivo ao mesmo tempo em que ele está sendo copiado, com:


```
. USE PESSOAL
. COPY TO B:PESSOAS
```

5. Ao se copiar um arquivo em um mesmo disco, deve-se escolher um outro nome. Não se especificando um drive, o comando COPY produzirá uma cópia no mesmo disco.

```
. USE PESSOAL
. COPY TO PESSOAS
```

6. O dBASE permite que parte do arquivo seja copiada mediante a condição FOR:

```
. USE PESSOAL
. COPY TO FUNC FOR CARGO = 'ESCRITURARIO'
```

7. E copia-se, no novo arquivo, apenas os campos selecionados através da condição FIELD:

```
. USE PESSOAL
. COPY TO PESSOAS FIELD NOME,CARGO
```

No exemplo acima, o novo arquivo, de nome PESSOAS, conterá apenas dois campos: NOME e CARGO.

8. Pode-se combinar todas as possibilidades através de:

```
. COPY TO PESSOAS FIELD NOME,SALARIO FOR CARGO = 'ESCRITURARIO'
```

O comando COPY pode criar outros formatos de arquivos além do normal .DBF (ver Capítulo 19 – Trabalhando com Arquivos não-dBASE).

Copiando uma Estrutura de Registro Através do Comando COPY:

9. Pode-se copiar a estrutura de um arquivo sem copiar os dados nele contidos:

```
. USE PESSOAL
. COPY STRUCTURE TO PESSOAS
```

10. Pode-se, também, selecionar os campos desejados na nova estrutura:

A estrutura do novo arquivo denominado PESSOAS contém apenas NOME e SALÁRIO.

```
. USE PESSOAL
. COPY STRUCTURE TO PESSOAL FIELD NOME,SALARIO
```

Ao se especificar o nome de um arquivo já existente, a ser usado como arquivo TO, todos os dados contidos nele são destruídos. Isto vale para qualquer forma do comando COPY (COPY dados ou COPY estrutura).

Este uso do comando COPY é um modo rápido de se criarem novos arquivos sem ter que se passar por todas as descrições do comando CREATE.

Mudando os Nomes dos Arquivos

Desde que o arquivo não esteja aberto, pode-se atribuir-lhe um outro nome através do comando RENAME. Não se acrescentando um ponto e a indicação de outro tipo de arquivo, adotar-se-á o tipo normal do arquivo DBF (arquivo de banco de dados).

```
RENAME PESSOAL TO PESSOAS
```

Através do comando RENAME, pode-se atribuir outro nome a qualquer arquivo do disco incluindo um nome e designando o tipo de arquivo. Por exemplo:

Nome do arquivo da forma de relatório:

```
. RENAME REPORT1.FRM TO REPORT6.FRM
```

Nome do arquivo de comando:
(CP/M):
(MS/DOS):


```
. RENAME INPUT.CMD TO ORDENS.CMD
. RENAME INPUT.PRG TO ORDENS.PRG
```

Nome do arquivo de memória:

```
. RENAME AMANHA.MEM TO TOTAIS.MEM
```

Nome do arquivo de índices:

```
. RENAME NMNX.NDX TO NOMEINDX.NDX
```

Nome do arquivo de formato:

```
. RENAME TELAESP.FMT TO TELA1.FMT
```

Para maiores informações relativas aos sete tipos de arquivos gerados pelos comandos dBASE, ver o Sumário dos Tipos de Arquivos dBASE no Apêndice D.

Trocando Dados em Múltiplos Registros

Com o comando REPLACE pode-se trocar os dados em todos os registros ou em um número qualquer de registros selecionados. Por exemplo, para substituir todas as datas do arquivo pela data do sistema:

```
. REPLACE ALL DATADEHOJE WITH DATE*( )
```

Esquecendo de ALL e digitando:

```
. REPLACE DATADEHOJE WITH DATE*( )
```

altera-se DATADEHOJE apenas no registro em curso. Pode-se alterar vários campos com o mesmo comando, embora os dados de entrada em cada registro sejam os mesmos:

```
. REPLACE ALL DATADEHOJE WITH DATE*( ), COR WITH '*VERMELHO'
```

O comando REPLACE é extremamente útil quando se quer trocar registros intercalados por todo o arquivo. A inclusão de FOR ao comando proporciona a capacidade de "Busca e Substituição".

Notar as aspas em torno dos campos "C". Notar, também, que não é necessário ALL quando se usa a condição FOR.

```
. REPLACE COR WITH "VERDE" FOR COR = "GREEN"
. REPLACE PRECO WITH 5.95 FOR PRECO = 6.50
. REPLACE PROGRAM WITH C FOR CARGO ="PROGRAMADOR SENIOR"
```

No último exemplo, busca-se uma coisa substituindo-a por outra.

O comando REPLACE também é indicado para aumentar o espaço destinado às descrições, após ter-se modificado a estrutura do registro (ver Capítulo 13). Por exemplo:

```
. REPLACE QUALIDADE WITH "MEDIANAMENTE BOA" FOR QUALIDADE ="MDBA"
```

E pode-se trocar um número específico de registros, por exemplo: do número 432 ao 441.

```
. USE NOME DO ARQUIVO
. 432
. REPLACE NEXT 10 JURISDICAO WITH 'VALE DO PARAIBA'
```

Um Modo Opcional de se Editarem os Dados

O comando REPLACE troca automaticamente os dados em cada registro selecionado. Querendo rever os dados antes de trocá-los ou ainda se as modificações não forem uniformes, usa-se o comando CHANGE no lugar do REPLACE.

O comando CHANGE é útil quando se quer alterar registros localizados ao longo do arquivo. Para se alterar um grupo de registros adjacentes, torna-se mais conveniente usar o comando BROWSE.

Através do CHANGE ordena-se, em dBASE, que sejam apresentados um ou mais campos em cada registro selecionado. Pode-se alterar os dados no campo ou passar para o campo seguinte, prosseguindo, então, para o próximo registro.

Por exemplo, suponhamos que se deseje conceder aumento salarial a todos os empregados que vêm trabalhando na companhia há um determinado tempo, mas quer-se examinar cada pessoa individualmente. Então:

O dBASE representará o primeiro registro que atenda à condição FOR.

Pressionando <RETURN> *desloca-se para o campo seguinte.*

Pressionando <RETURN> *novamente, desloca-se para o campo do salário.*

Introduzindo aqui um novo salário, ele será alterado. Porém, pressionando < RETURN >, *desloca-se para o próximo registro, pois o último dos campos pedidos terá sido atingido.*

```
. USE PESSOAL
. CHANGE FIELD NOME, ANO, SALARIO FOR ANO  < 1975

REGISTRO 00002

NOME:     Luiz
CHANGE:

ANO:      1972
TO:

SALARIO:  650.00
TO:
```

Os campos "C" apresentam três opções: (1) < RETURN >, *para saltá-lo; (2) entrar com os dados a fim de executar uma busca e substituição ou (3) entrar com* < CTRL Y >, *para permitir a escrita dos novos dados sobre os antigos. Estas opções são o mesmo que usar o comando EDIT fora do modo full-screen (descrito na página 94).*

Os campos "N" e "L" apresentam duas opções: (1) < RETURN > *para saltá-la, ou (2) entrar com os dados necessários à alteração.*

Com o comando CHANGE, necessita-se da expressão FIELD, ainda que se peça apenas um campo:

```
. CHANGE FIELD TERRITORIO FOR TERRITORIO = "SETENTRIONAL"
```

Outro Modo de se Localizar os Registros

Desejando alterar registros espalhados por todo o arquivo através do processo de edição em full-screen, usa-se o comando LOCATE para selecionar o registro e, depois aplica-se o comando EDIT.

Como no exemplo relativo ao comando CHANGE, tente entrar com:

```
. USE PESSOAL
. LOCATE FOR ANO <1975
RECORD:  00002
```

O dBASE responderá com o número do primeiro registro que atenda à condição FOR.

Edita-se, então, o registro através do comando EDIT normal:

```
. EDIT 2
```

Terminada a edição daquele registro, digitar:

```
. CONTINUE
REGISTRO: 00004
```

e o dBASE localizará o próximo registro.

Querendo trabalhar com um conjunto específico de registros, por exemplo, do 101 ao 150, digitar:

```
. 101
. LOCATE NEXT 50 FOR ANO <1975
```

Inserindo um Registro no Meio de um Arquivo

Pode-se acrescentar um novo registro ao arquivo, em qualquer lugar desejado, mediante o comando INSERT, devendo, primeiramente, posicionar o ponteiro de registro. Caso se pretenda introduzir um novo registro entre os de números 3 e 4, digita-se:


```
. 3
. INSERT
```

A tela de entrada de dados será a mesma que a obtida com APPEND podendo-se, então, entrar com os dados. O comando INSERT permite apenas a entrada de um registro. Querendo acrescentar mais registros depois do que já foi inserido, deve-se entrar novamente com o comando INSERT.

Também, para inserir um registro antes do registro em curso, digitar:

```
. INSERT BEFORE
```

Querendo inserir um registro no arquivo, mas de forma a preenchê-lo posteriormente com os dados, cria-se um registro em branco e salta-se a tela de entrada de dados, digitando:

```
. INSERT BLANK
```

ou

```
. INSERT BLANK BEFORE
```

Um Comando UPDATE para Lotes de Dados

Quando os analistas de sistemas projetam os sistemas de informação, dois são os tipos de arquivo: o de cadastro e o de movimento. O registro de cadastro contém dados descritivos, tais como: nomes, endereços, cargos e categorias. Contém, igualmente, informações resumidas como: cifras relativas ao período de um ano até a presente data e quantidades disponíveis. O registro do cadastro deve englobar tudo o que é necessário ao conhecimento de determinado assunto. Os registros de cadastro destinam-se a clientes, vendedores, produtos, componentes, pessoas etc. Até uma lista das pessoas a quem serão enviados cartões de Natal constitui exemplo de um cadastro.

O segundo tipo de arquivo, o de movimento, contém um registro para cada evento, atividade ou alteração ocorrida. O registro de movimento contém a data e os dados que descrevem pedidos recebidos, cheques pagos, aumento de estoque; essencialmente, qualquer alteração que reflita, também, no cadastro. Os registros de movimento representam um evento simples, porém de muita importância, pois fornecem os elementos indicadores necessários a uma auditagem e que permitem que se determinem e examinem todas as mudanças que concorreram para a situação atual do cadastro.

As alterações podem ser de dois tipos: as que são aritmeticamente acrescentadas aos dados existentes e as que irão substituí-los. Caso se decida reunir todas essas alterações em um arquivo de movimento separado, pode-se atualizar o cadastro através do comando UPDATE.

Para tanto, ambos os registros – de cadastro e de movimento – deverão conter um campo com dados correspondentes e de nomes idênticos. Tal campo, chamado CAMPO-CHAVE é, em geral, um nome ou número que identifica, de forma única, um registro do outro (por exemplo, o número de uma pessoa, o número do CPF ou o número de um produto).

O comando UPDATE trabalha de dois modos, dependendo da seqüência dos arquivos de cadastro e de movimento

Acesso Seqüencial: O comando UPDATE assume o método de Acesso Seqüencial e requer que ambos os arquivos estejam na mesma ordem de seqüência. O arquivo de movimento deve estar ordenado de acordo com a seqüência do campo-chave. O arquivo de cadastro, porém, tanto poderá estar ordenado quanto indexado pelo campo-chave.

Acesso Aleatório: No modo do Acesso Aleatório, o arquivo de movimento pode estar em qualquer seqüência, enquanto o de cadastro precisa estar indexado pelo campo-chave.

Um exemplo ajudará a esclarecer o assunto. Suponhamos que se queira atualizar um cadastro chamado PRODUTOS a partir de um arquivo de movimento, de nome TROCAS. Ambos os arquivos contêm um campo-chave denominado PRODNÚMERO e um campo chamado DISPONÍVEL. O campo DISPONÍVEL do registro do cadastro flutua constantemente, enquanto o seu homônimo no registro de movimento representa um acréscimo ou um decréscimo. Efetuam-se os decréscimos através da introdução de um número negativo no campo DISPONÍVEL do registro de movimento. Abre-se o cadastro e trabalha-se a partir do arquivo de movimento:

```
. USE PRODUTOS
. UPDATE FROM TROCAS ON PRODNUMERO  ADD DISPONIVEL
```

Escolhendo o método Aleatório:

```
. UPDATE FROM TROCAS ON PRODNUMERO ADD DISPONIVEL RANDOM
```

Suponhamos que se deseje, no registro do cadastro, algo sobre a última transação que deu origem a uma atualização. Admitindo um campo, em ambos os registros, denominado CLIENTE, poder-se-ia atualizar o cadastro da seguinte forma:

```
. UPDATE FROM TROCAS ON PRODNUMERO ADD DISPONIVEL REPLACE;
CLIENTE
```


Se o campo que se pretende substituir no cadastro não possuir o mesmo nome do campo de movimento, pode-se executar o comando REPLACE tradicional, da seguinte forma:

```
UPDATE FROM TROCAS ON PRODNUMERO ADD  DISPONIVEL REPLACE;
ULTICLIE WITH CLIE
```

É possível atualizar vários campos, separando-se seus nomes por vírgulas:

```
ADD  campo1, campo2, campo3;
REPLACE   campo1, campo2 WITH campo3, campo6
```

O dBASE Necessita de Alguma Assistência

ATENÇÃO:
Informação importante sobre o comando UPDATE.

Tal como na versão 2.4, o dBASE não acusa erro se o comando UPDATE, partindo de um registro de movimento, não encontrar um registro de cadastro correspondente. Portanto, é preciso ter certeza de que os campos-chave estão corretos e de que encontrarão um registro correspondente no cadastro. E isto é conseguido ao se executar o seguinte arquivo de comando que verificará o movimento em relação ao cadastro e que acusará os movimentos não correspondidos: e tem-se acesso ao editor de texto, com MODIFY COMMAND, designando-se este programa por VERIFICA (ou por qualquer outro nome) e com as linhas de codificação indicadas abaixo. Usa-se <CTRL W> a fim de se preservar o arquivo de comando após o término da operação. Executa-se o programa com o uso de DO VERIFICA que perguntará pelos nomes dos arquivos.

Este programa requer que o cadastro seja indexado pelo campo-chave.

O arquivo de movimento pode estar em qualquer seqüência. Os comentários à direita não deverão ser incluídos, pois constam apenas para efeito de informação.

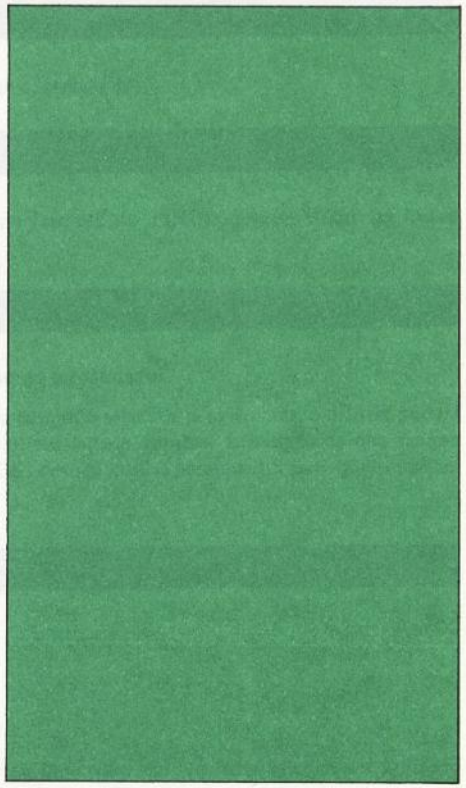
```
SET TALK OFF              desligar as mensagens interativas.
CLEAR
ERASE
ACCEPT "INSERIR O NOME DO CADASTRO" TO CADAS
                          entrar com o nome do cadastro.
ACCEPT "INSERIR O NOME DO ARQUIVO DE INDICES DO CADASTRO" TO IND
                          entrar com o nome do arquivo de índices.
ACCEPT "INSERIR O NOME DO ARQUIVO DE MOVIMENTO"
TO MOVIM                  entrar com o nome do arquivo de movimento.
ACCEPT "INSERIR O NOME DO CAMPO CHAVE" TO CHAVE
                          entrar com o nome do campo-chave.
?
USE &MOVIM                abrir o arquivo de movimento cujo nome foi armazenado em
                          MOVIM.
SELECT SECONDARY          selecionar o cadastro.
USE &CADAS INDEX &IND     abrir nomes de cadastros armazenados em CADAS e IND.
SELECT PRIMARY            selecionar arquivo de movimento.
DO WHILE . NOT . EOF      faça o seguinte até o final do arquivo.
STORE &CHAVE TO GUARDA    armazenar temporariamente o campo-chave do movimento.
SELECT SECONDARY          selecionar o cadastro.
FIND &GUARDA              buscar registro de cadastro correspondente.
IF # = 0                  o número de registro ϕ significa não haver correspondente.
? GUARDA                  exibir chave do movimento
? ? "NAO ENCONTRADO REG #"    exibir mensagem de não correspondência.
SELECT PRIMARY            selecionar arquivo de movimento.
                          mostrar o número do regitro do arquivo de movimento.
? ? #
ENDIF                     fim da condição IF.
SELECT PRIMARY            selecionar arquivo de movimento.
SKIP                      obter próximo registro de movimento
```



```
ENDDO                     Voltar ao início de loop DO.
SET TALK ON               ligar as mensagens interativas.
CLEAR
RETURN
```

PARTE 5

IMPORTANTES FUNÇÕES ADICIONAIS



CAPÍTULO 12

OBTENDO TOTAIS RESUMIDOS

O comando TOTAL resume dados numéricos em grupos de registros afins. Os resultados são armazenados em um arquivo-resumo separado.

Os registros afins são identificados pelo campo que contém a característica para a qual estamos gerando um resumo. Este é o campo CHAVE que engloba quaisquer dados repetidos em mais de um registro, tais como número de contas correntes, categorias, tipos etc.

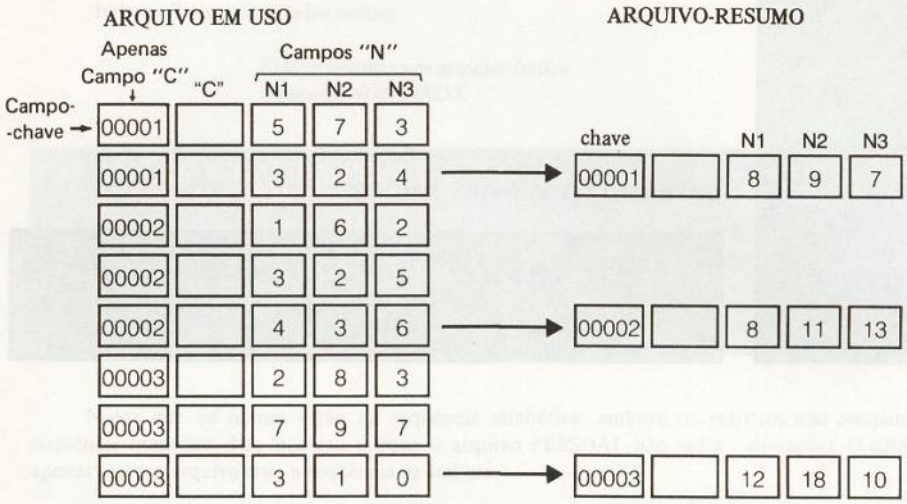
O arquivo deve ser organizado em uma seqüência de chave, segundo um critério de ORDENAÇÃO ou de INDEXAÇÃO. O comando TOTAL começa pelo registro no 1 e soma os campos numéricos de todos os registros seguintes que apresentam a mesma chave. Quando esta é alterada, um registro-resumo é emitido para aquele grupo, após o que a soma é reiniciada para o próximo grupo. O comando TOTAL não gera um total geral.

Suponhamos que se tenha um arquivo de pagamentos, com seqüência igual à dos números das contas correntes com vários pagamentos para cada uma delas. O resumo dos pagamentos relativos a cada conta corrente poderia ser feito digitando-se:

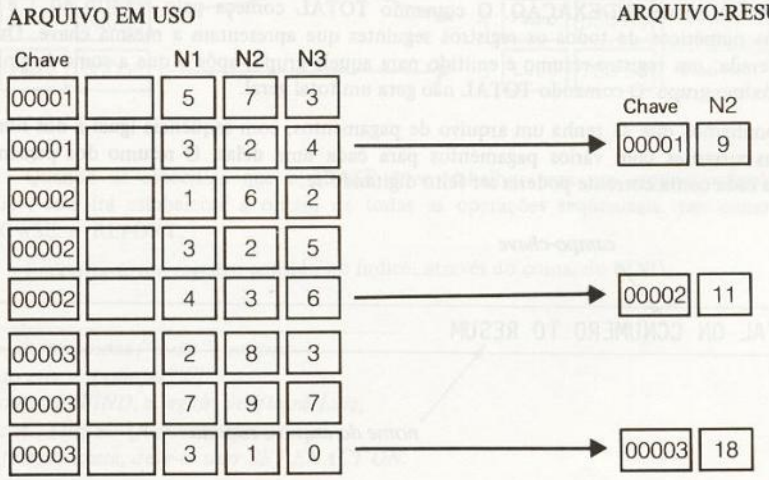
campo-chave

```
TOTAL ON CCNUMERO TO RESUM
```

nome do arquivo resumo



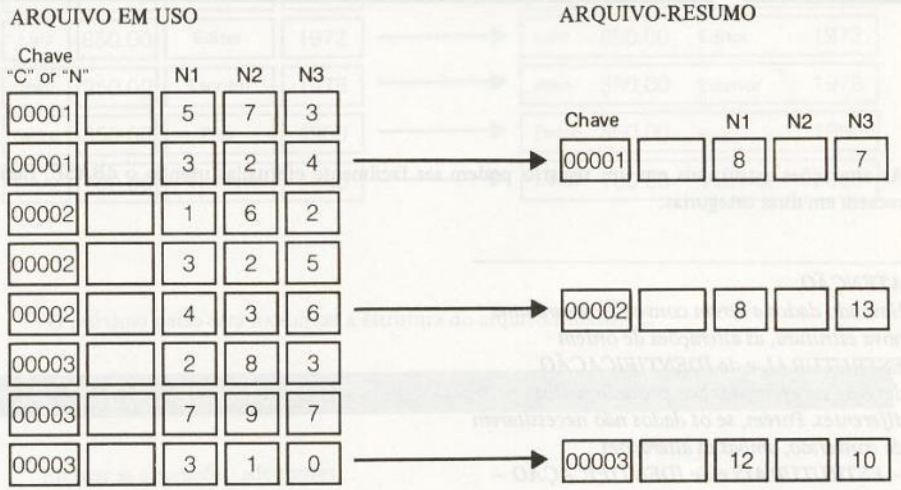
Já existindo o arquivo-resumo, ele determinará quais os campos totalizados. Vamos supor que já se disponha de um arquivo-resumo preexistente do qual constem apenas o campo-chave e um campo numérico:



Não se desejando a totalização de todos os campos numéricos, pode-se especificar quais os campos desejados:

```
TOTAL ON CCNUMERICO TO RESUM FIELDS N1,N3
```

Se forem necessários apenas totais impressos, pode-se usar as opções SUBTOTAL e SUMMARY e REPORT, com o comando REPORT, que também dará um total global.



Usando a expressão FIELDS, o campo-chave tanto poderá ser um campo "N", como "C".

CAPÍTULO 13

MODIFICANDO A ESTRUTURA DO REGISTRO

As alterações estruturais em um registro podem ser facilmente efetuadas usando o dBASE. Elas recaem em duas categorias:

ATENÇÃO:
Havendo dados a serem convertidos para uma nova estrutura, as alterações de ordem ESTRUTURAL e de IDENTIFICAÇÃO deverão ser efetuadas por procedimentos diferentes. Porém, se os dados não necessitarem de conversão, ambas as alterações – ESTRUTURAIS e de IDENTIFICAÇÃO – poderão ser realizadas simultaneamente, através do comando MODIFY STRUCTURE.

Alterações Estruturais

1. Anexação de novos campos (até um máximo de 32).
2. Eliminação de campos.
3. Aumento ou diminuição do tamanho de um campo.

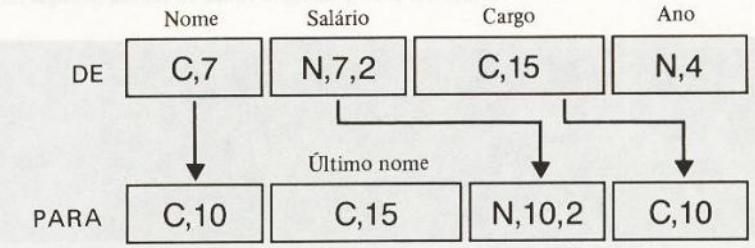
Alterações na Identificação

1. Mudança no nome do campo.
2. Mudança no tipo do campo.

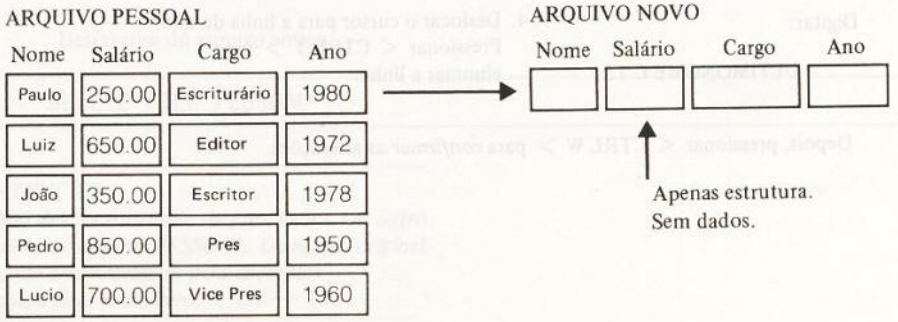
Alterações Estruturais

Como exemplo, alteremos a estrutura de PESSOAL, como segue:

1. AUMENTAR "NOME", DE 7 PARA 10 BYTES.
2. AUMENTAR "SALÁRIO", DE 7 PARA 10 BYTES
3. ANEXAR UM CAMPO "ULTIMONOME"
4. ELIMINAR O CAMPO "ANO"



A primeira coisa a fazer é copiar através do comando COPY a estrutura original para um novo arquivo.



Digitar:

```
. USE PESSOAL
. COPY STRUCTURE TO NOVO
```

Em seguida, usando o comando MODIFY, modificar o novo arquivo:

```
. USE NOVO
. MODIFY STRUCTURE
MODIFY APAGARA TODOS OS DADOS DOS REGISTROS... PROCEDER? (Y/N) Y
```

Responder SIM.
Lembrar de que se está apagando um arquivo vazio.

Realizar as quatro alterações:

```
          NOME          TIPO       TAMANHO      DEC
CAMPO 01  :NOME          C          007          000
CAMPO 02  :SALARIO       N          007          002
CAMPO 03  :CARGO         C          015          000
CAMPO 04  :ANO           N          004          000
```

3. Deslocar o cursor para a linha do salário. Pressionar < CTRL N >.

1. Inserir 10

2. Inserir 10

4. Deslocar o cursor para a linha do ano. Pressionar < CTRL T > para eliminar a linha.

Digitar:

ULTIMONOME C 15

Depois, pressionar < CTRL W > para *confirmar* as alterações.

Digitar, agora:

```
. DISPLAY STRUCTURE
ESTRUTURA PARA ARQUIVO: NOVO.DBF
NUMERO DE REGISTROS:   00000
DATA DA ULTIMA ATUALIZACAO:  D/M/A
BANCO DE DADOS DE USO PRIMARIO
CAMPO     NOME          TIPO          TAMANHO          DEC
001       NOME           C             010
002       ULTIMONOME     C             015
003       SALARIO        N             010             002
004       CARGO          C             015
** TOTAL **                           00051
```

A nova estrutura deverá ter este aspecto.

Em seguida, anexar os dados originais à nova estrutura:

```
. APPEND FROM PESSOAL
```

Observar.

```
. DISPLAY ALL
00001   Paulo                          250.00  Escriturario
00002   Luiz                           650.00  Editor
00003   Joao                           350.00  Escritor
00004   Pedro                          850.00  Pres
00005   Lucio                          700.00  Vice Pres
00006
```

Desfazer-se do arquivo antigo:

```
. DELETE FILE PESSOAL
```

ATENÇÃO
Não deletar e atribuir, de fato, agora, um outro nome ao arquivo PESSOAL. O arquivo original ainda será necessário para exemplos posteriores neste livro.

Atribuir um outro nome ao arquivo novo, através do comando RENAME.

```
. USE
. RENAME NOVO TO PESSOAL
```

Isto fechará o arquivo NOVO. Não se pode atribuir um outro nome (RENAME) a um arquivo aberto.

Uma modificação estrutural acabou de ser completada.

Procedimento para Alteração Estrutural

Certificar-se de não perder o Passo 3 (USE < arquivonovo >) e, então, responder Y (sim) à pergunta proposta. Fecha < arquivonovo > para que se lhe atribua um novo nome.

```
1) . USE <arqvelho>
2) . COPY STRUCTURE TO <arqnovo>
3) . USE <arqnovo>
4) . MODIFY STRUCTURE
5) . APPEND FROM <arqvelho>
6) . DELETE FILE <arqvelho>
7) . USE
8) . RENAME<arqnovo> TO <arqvelho>
```

Alterar a estrutura através dos comandos de Edição:
< CTRL N > *abre uma linha*
< CTRL T > *elimina uma linha*
< CTRL Y > *limpa a linha*
< CTRL W > *preserva as alterações*
< CTRL Q > *abandona as alterações.*

Alterações na Identificação:

Como exemplo, serão trocados todos os nomes dos campos do arquivo PESSOAL. O primeiro passo é converter, temporariamente, o arquivo em um arquivo-texto que não contenha uma identificação de arquivo de banco de dados (DBF):

```
. COPY TO TEMP SDF
```

Nome	Salário	Cargo	Ano
Paulo	250.00	Escriturário	1980
Luiz	650.00	Editor	1972
João	350.00	Escritor	1978
Pedro	850.00	Pres	1950
Lucio	700.00	Vice Pres	1960

Paulo	250.00	Escriturário	1980
Luiz	650.00	Editor	1972
João	350.00	Escritor	1978
Pedro	850.00	Pres	1950
Lucio	700.00	Vice Pres	1960

O próximo passo será modificar a estrutura do arquivo PESSOAL:

```
. MODIFY STRUCTURE
```

Efetuar as alterações adequadas:

PESSOAL.DBF

Quem	Quanto	Ocupação	Quando

Converter o arquivo-texto de volta à estrutura modificada do arquivo dBASE, através de:

```
. APPEND FROM TEMP SDF
```

TEMP.TXT

Paulo	250.00	Escriturário	1980
Luiz	650.00	Editor	1972
João	350.00	Escritor	1978
Pedro	850.00	Pres	1950
Lucio	700.00	Vice Pres	1960

PESSOAL.DBF

Quem	Quando	Ocupação	Quando
Paulo	250.00	Escriturário	1980
Luiz	650.00	Editor	1972
João	350.00	Escritor	1978
Pedro	850.00	Pres	1950
Lucio	700.00	Vice Pres	1960

Procedimento para Alteração na Identificação

```
1) . USE <ARQUIVO A SER ALTERADO>
2) . COPY TO TEMP SDF
3) . MODIFY STRUCTURE

4) . APPEND FROM TEMP SDF
5) . DELETE FILE TEMP.TXT
```

Responder Y (Sim) à indagação. Alterar apenas os nomes dos campos e/ou tipos de campo. Não anexar ou deletar campos, nem alterar qualquer tamanho de campo. Pressionar < CTRL W > *para preservar as alterações.*

Altere, agora, PESSOAL, levando-o de volta à sua forma original.

Lembre-se de que, se não houver dados a converter, ambas as alterações – ESTRUTURAL e de IDENTIFICAÇÃO – poderão ser realizadas simultaneamente, mediante o comando MODIFY STRUCTURE. Não há necessidade de seguir os procedimentos deste capítulo, a menos que se deseje converter em registros modificados os dados já existentes.

CAPÍTULO 14

A INDEXAÇÃO DE DADOS VISANDO A UMA RÁPIDA RECUPERAÇÃO

Encontrando Registros em um Arquivo

Quando se introduz um comando DISPLAY FOR, o dBASE lê para a memória do computador cada um dos registros do arquivo, comparando-os com os dados desejados. O DISPLAY FOR permite que se indague sobre os dados em qualquer combinação de campos no registro. Além disso, pode-se perguntar por conjuntos de dados, tais como quantias superiores ou inferiores a determinadas cifras. O DISPLAY FOR proporciona uma flexibilidade completa em relação às perguntas.

Suponhamos, porém, que se disponha de um grande arquivo de registros de clientes e que pretendamos procurar um registro para atender certo cliente que telefona. Sendo o arquivo muito grande e utilizando DISPLAY FOR CLIENTES = "COMPANHIA DE PROGRAMAS PARA COMPUTADOR", o tempo de espera até a ordem ser processada corresponderá a alguns minutos. Já um método alternativo, denominado INDEXAÇÃO, levará apenas poucos segundos para executar a mesma tarefa.

O índice contém os dados procurados e a localização do registro no disco. Em vez de procurar o arquivo na forma seqüencial, o dBASE poderá percorrer rapidamente o índice e localizar, de modo preciso, o registro desejado.

Vejamos o que acontece quando se faz a indexação de um arquivo. Observar, primeiramente, um arquivo não indexado:

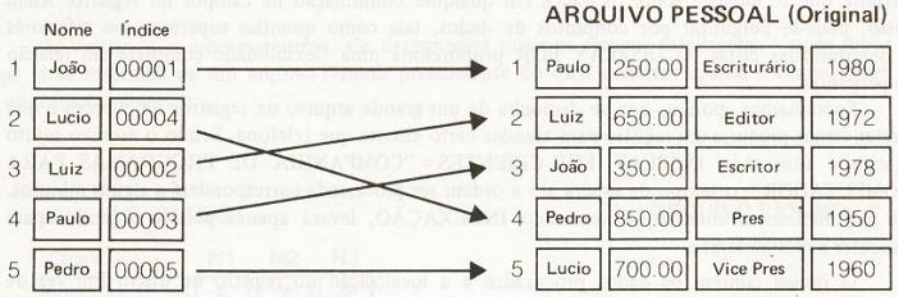
```
. USE PESSOAL
. DISPLAY ALL NOME
00001  Paulo
00002  Luiz
00003  Joao
00004  Pedro
00005  Lucio
```

Indexando-o, agora, pelos nomes:

Está-se criando um arquivo-índice chamado NOMEINDX.

```
. INDEX ON NOME TO NOMEINDX
. DISPLAY ALL NOME
00003  Joao
00005  Lucio
00002  Luiz
00001  Paulo
00004  Pedro
```

Notar que os nomes estão na seqüência alfabética, embora os registros não estejam na seqüência numérica. Isto ocorreu porque o arquivo PESSOAL não sofre alterações. O dBASE apenas exibe o arquivo com a seqüência de índices.



Quando se especifica que o dBASE deve trabalhar com um arquivo valendo-se de um índice, este irá estabelecer a ordem de todas as operações seqüenciais, tais como DISPLAY, BROWSE e REPORT.

Encontra-se um registro contido no índice, através do comando FIND:

Não usamos aspas ("Luiz"), mesmo sendo este um campo "C". O comando FIND, a seguir, verificará Luiz, Luiz A., Luiz B. Querendo apenas uma verificação exata, deve-se usar SET EXACT ON.


```
. FIND Luiz
```

Não encontrando o registro desejado, o dBASE envia uma mensagem NO FIND. Encontrando-o, porém, apenas se obtém o ponto; o registro não é automaticamente exibido. Para trazê-lo à tela, usa-se DISPLAY ou BROWSE.

Múltiplos Índices em um Mesmo Arquivo

É possível criar-se um número qualquer de índices para um arquivo, embora apenas um de cada vez possa ser aberto com ele. Cria-se um outro índice para o arquivo-exemplo, digitando:

```
. INDEX ON CARGO TO CARGINDX
```

Pode-se, agora, encontrar um cargo no arquivo, através de:

```
. FIND Vice Pres
```

Desejando encontrar novamente um nome (através do FIND), deve-se voltar ao índice de nomes, mediante:

```
. SET INDEX TO NOMEINDX
```

Alterando os dados, os índices também deverão ser alterados.

Adicionando novos registros ao arquivo ou editando registros já existentes, o dBASE poderá ser instruído para atualizar até sete índices relativos àquele arquivo. Isto significa que sempre que se usar APPEND, EDIT, REPLACE ou PACK, deve-se cuidar para que todos os índices sejam incluídos. Isto é possível digitando:

```
. SET INDEX TO NOMEINDX,CARGINDX
```

Pode-se especificar até sete índices separados por vírgulas.

O primeiro índice citado refere-se sempre ao índice em uso, em condições de ser utilizado para que se encontrem os registros no arquivo.

Abrindo os Arquivos

Na próxima vez que o arquivo for usado, deve-se lembrar novamente de se especificar o índice. Para tanto, usa-se um comando SET INDEX TO ou abre-se o arquivo mencionando-se o índice:

```
. USE PESSOAL INDEX NOMEINDX
```

Os nomes dos arquivos de dados e de índices podem ser os mesmos, desde que os tipos dos arquivos sejam diferentes.

```
. USE PESSOAL INDEX PESSOAL
```

PESSOAL.DBF PESSOAL.NDX

O registro nº 1 não é, necessariamente, o primeiro registro.

Quando o arquivo for indexado, o registro nº 1 poderá ocupar qualquer lugar do arquivo. Para se chegar ao primeiro registro na seqüência indexada, digitar:

```
. GO TOP
. DISPLAY
```

Exibe o primeiro registro do índice.

ou

```
. GO BOTTOM
. DISPLAY
```

Apresenta o último registro do índice.

Esquecendo de Atualizar um Índice

Havendo mais de sete índices no arquivo ou caso se esqueça de incluir o índice ao se alterarem os dados no arquivo, este poderá ser reindexado, digitando-se:


```
. USE PESSOAL INDEX NOMEINDX
. REINDEX
```

Este procedimento deverá ser aplicado a cada arquivo de índice não mantido em uso.

A Indexação Pode Acelerar a Ordenação para os Relatórios

Não sendo necessário reter uma cópia separada do arquivo ordenado, pode-se indexar o arquivo e imprimir um relatório a partir do arquivo indexado. A indexação é, em geral, muito mais rápida do que a ordenação, especialmente quando se lida com grandes arquivos. A ordenação é realizada em apenas um campo de cada vez, enquanto um único índice pode ser criado, simultaneamente, utilizando vários campos.

A Indexação em Múltiplos Campos

Funciona apenas em campos "C".
Os campos numéricos devem ser convertidos mediante a função STR.

Pode-se criar um índice em dois ou mais campos de um registro. Isto é muito útil na geração de relatório cuja seqüência deva obedecer à de um campo contido em outro campo. Usa-se o símbolo + para unir tais campos.

Por exemplo, suponhamos que se queira um relatório sobre o último nome, por região. Considerando ambos os casos como de caracteres, pode-se entrar com:

```
INDEX ON REGIAO + ULTIMONOME
```

O comando FIND poderá ser utilizado na localização de um registro indexado por vários campos, porém, é preciso muito cuidado. Como ilustração, suponhamos que REGIÃO seja um campo "C" de 10 bytes; então, ao se usar o comando FIND, deve-se inserir os 10 bytes. Por exemplo:

Se a região fosse um campo numérico de 5 bytes dever-se-ia, primeiramente, converter o campo numérico em um de caracteres através da função STR (Capítulo 18).

```
INDEX ON STR (REGIAO,5) + ULTIMONOME
```

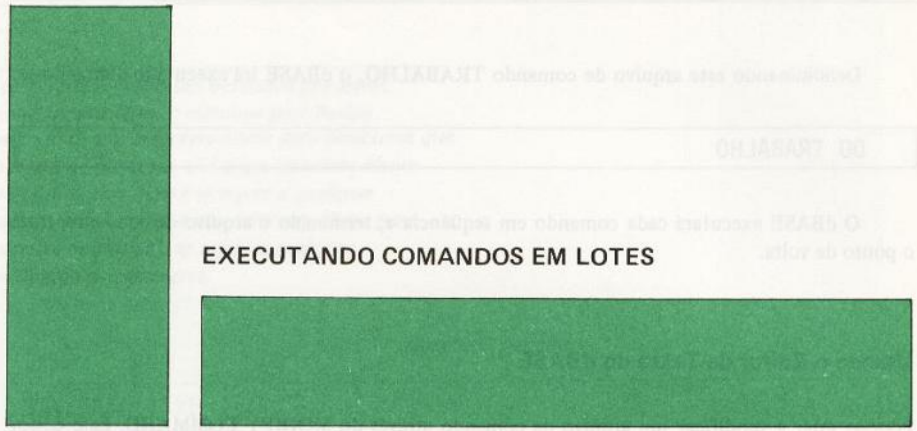
```
FIND NORTE      JONES
```

Cinco espaços em branco

Os Índices Aumentam o Tempo de Processamento

Se, por um lado, os índices podem economizar tempo, por outro, podem aumentar o tempo de processamento sempre que se anexem ou alterem registros em um arquivo. Deve-se ter em mente que os índices também necessitam de atualização. Este dilema vem preocupando os projetistas de sistemas há anos. Sendo possível esperar cinco minutos duas vezes ao dia, não se deve, então, retardar todo o resto do mecanismo. Se, porém, os arquivos contiverem milhares de registros, os índices tornar-se-ão uma necessidade para um acesso rápido.

CAPÍTULO 15



EXECUTANDO COMANDOS EM LOTES

O dBASE executará um lote de comandos, em seqüência, se eles forem escritos em um ARQUIVO DE COMANDO. Este poderá ser gerado através de um editor de texto ou pelo editor embutido do dBASE.

Se o sistema operacional for o MS-DOS, o tipo de arquivo de comando será o .PRG.

Os arquivos de comando devem apresentar um tipo de arquivo .CMD, a fim de serem identificados como tais pelo dBASE.

Suponhamos que se deseje abrir o arquivo PESSOAL, ordenar o seu campo ANO e preparar um relatório chamado QTOTEMPO.

Por hipótese, a forma de relatório QTOTEMPO já foi criada de modo a poder gerar um arquivo de comando com o seguinte aspecto:

```
. USE PESSOAL
. SORT ON ANO TO ARQANO
. USE ARQANO
. REPORT FORM QTOTEMPO
```

Ver MODIFY COMMAND, a seguir.

Denominando este arquivo de comando TRABALHO, o dBASE irá executá-lo digitando-se:

```
DO TRABALHO
```

O dBASE executará cada comando em seqüência e, terminado o arquivo de comando, trará o ponto de volta.

Usando o Editor de Texto do dBASE

Pode-se criar e modificar um arquivo de comando através do MODIFY COMMAND. Este editor embutido é muito útil na criação de tais arquivos.

Nota: o editor do dBASE será restritivo se o arquivo de comando for maior que 4000 bytes. Isto poderá ocorrer na programação no NÍVEL 2.

Para criar um arquivo de comando de nome TRABALHO, digitar:

```
. MODIFY COMMAND TRABALHO
```

Se o TRABALHO ainda não existir, o dBASE gerará um novo arquivo de comando chamado TRABALHO.CMD e apresentará uma tela em branco para ser preenchida. Insere-se cada comando em linhas separadas. As linhas não podem conter mais de 75 caracteres.

Dividir as longas linhas de comando. Se a linha de comando ultrapassar os 75 caracteres, dividi-la através de ponto e vírgula. Ver exemplos na página 45.

Terminando de escrever o arquivo de comando, usa-se < CTRL W > para preservar, no disco, o arquivo recém-criado.

O cursor e os códigos de edição funcionam da mesma maneira que no modo EDIT, com o acréscimo de:

< CTRL T >	Elimina a linha inteira no cursor
< CTRL N >	Insere uma linha em branco antes do cursor
< CTRL R >	Puxa a linha para cima
< CTRL C >	Puxa a página para baixo
< CTRL W >	Preserva o arquivo de comando
< CTRL Q >	Abandona as alterações no arquivo de comando

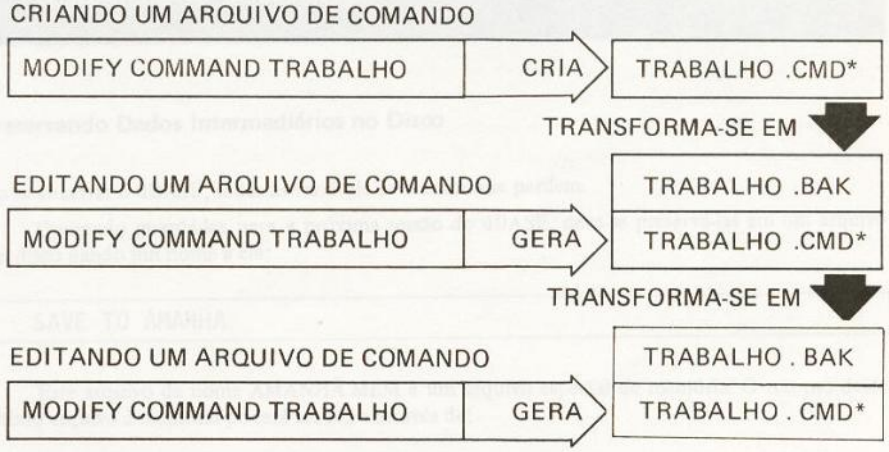
Da próxima vez que se digitar:

```
MODIFY COMMAND TRABALHO
```

obter-se-á o arquivo de comando na tela.

Na primeira vez em que se chamar um arquivo de comando anteriormente criado e se encerrar a sessão de edição através de < CTRL W > (preservar alterações), o dBASE criará, de modo automático, um arquivo de reserva. O dBASE caracterizará o arquivo de reserva no disco por um tipo de arquivo .BAK. (BACKUP).

O TRABALHO . CMD passa a ser TRABALHO . BAK. Da próxima vez em diante, a nova versão transforma-se no (.CMD) atual. A versão anterior transforma-se em reserva (.BAK), e a reserva anterior é eliminada. Conseqüentemente, sempre se tem, no disco, as versões atual e anterior do arquivo de comando.



* Se o sistema operacional for o PC DOS ou o MS DOS, esta extensão será .PRG.

Usando o *Wordstar*® na Criação de Arquivos de Comando

O Wordstar e outros programas processadores de palavras inserem códigos de impressão e de formato (tanto visíveis, como invisíveis) no arquivo. Assim, ao se escrever um arquivo de comando através de um processador de palavras, deve-se usar o modo "não-documento". No Wordstar, ele equivale à opção "N".

Pode-se, também, "limpar" um arquivo de comando escrito por um processador de palavras. No CP/M, tal tarefa é realizada através da função "Z" do programa de cópia PIP.

```
A > PIP A:=B TRABALHO.CMD [Z]
```

CAPÍTULO 16

CALCULANDO E ARMAZENANDO DADOS TEMPORÁRIOS

Usando ? no Cálculo de Dados

O dBASE poderá ser utilizado como uma calculadora convencional sempre que o ponto aparecer. Desejando obter a resposta de 2 + 2?, usa-se "?" para dar saída ao resultado:

```
. ? 2+2
4
```

Pode-se efetuar vários cálculos de uma só vez:

```
. ? 2+2, 5*5, 4-3, 500/25, 400+50+35
4        25        1        20        485
```

```
. ? 500 + 34 +  1 * 3 + 456 * 2
1449
```

Quando vários cálculos são necessários à obtenção de um único resultado, o dBASE executa, em primeiro lugar, as multiplicações e divisões e, a seguir, as adições e subtrações.

No exemplo acima, as operações 1 * 3 e 456 * 2 são realizadas em primeiro lugar. Para se somar as três primeiras parcelas e, em seguida, efetuar a multiplicação, será necessário o uso de parênteses. Notar a diferença:

```
. ? (500 + 34 + 1) * 3 + 456 * 2
2517
```

A colocação dos parênteses reúne o grupo de números. Eles são calculados como um único valor.

Pode-se, também, extrair alguns valores do arquivo e submetê-los a outros cálculos:

```
. USE PESSOAL
. SUM SALARIO
2800.00
. COUNT
COUNT = 00005
. ?   2800/5
560
```

As variáveis de memória (a serem discutidas posteriormente neste capítulo) poderão ser combinadas com valores reais:

```
. USE PESSOAL
. SUM SALARIO
2800.00
. COUNT TO QUANTOS
COUNT = 00005
. ? 2800/QUANTOS
560
```

```
. ? SALARIO * 2
. USE PESSOAL
. SUM SALARIO TO SALTOT
. 3
. ? SALARIO/SALTOT
```

O salário no registro nº 3 é dividido pelo salário total do arquivo.

Usando a Memória como um Rascunho

No decorrer da operação com o dBASE, pode ser necessário anotar alguns fatos e cifras extraídos dos arquivos. O dBASE permite que se atribua um nome a estes dados e que eles sejam armazenados na memória, visando uma referência posterior.

A título de exemplo, suponhamos ter somado um campo numérico e que desejamos saber como este total se comporta em relação aos outros totais do arquivo. Através do SUM SALÁRIO do arquivo PESSOAL, o dBASE apresenta o total daquele campo:

```
. SUM SALARIO
2800.00
```

```
. SUM SALARIO TO SALTOT
2800.00
```

O valor 280.00 ficou armazenado em SALTOT para ser usado sempre que necessário. O SALTOT permanecerá na memória como uma variável de memória, até que seja eliminado ou até que o dBASE seja encerrado.

Variável é um outro nome para o campo.

Além do mais, a variável de memória pode ser utilizada de vários modos. Apenas querendo ver o que ela contém, digitar:

O dBASE exibirá

```
. ? SALTOT
2800.00
```

Para saber a média dos salários do arquivo:

```
. COUNT TO TOTPESSOAL
COUNT = 00005
```

Em seguida, efetua-se o cálculo da média dividindo o salário total pelo número total de pessoas:

```
. ? SALTOT/TOTPESSOAL
560.00
```

É exatamente assim que os programadores escrevem os cálculos a serem executados em seus programas. O dBASE, tal como as linguagens de programação, guarda o local onde os dados estão armazenados e permite que a referência a eles seja feita por seus nomes, ao invés de ser feita por suas localizações reais na memória. Esta é a função primária de uma linguagem de programação.

Pode-se somar (através do comando SUM) o salário total de um arquivo diferente e determinar a porcentagem de um total em relação ao outro.

O salário total representa 43% do salário total no segundo arquivo.

```
. USE ARQ2
. SUM SALARIO TO SALTOT2
6457.00
. ? SALTOT/SALTOT2
0.43
```

Mediante o comando STORE, pode-se criar uma variável de memória ou alterar o seu conteúdo:

```
. STORE 50000 TO SALTOT
50000
. STORE SALTOT TO SALTOT2
50000
. ? SALTOT, SALTOT2
50000     50000
```

e calcular, também, a variável de memória.

```
. STORE SALTOT * 2 TO SALTOT
100000
```

O dBASE permite o armazenamento pelos três tipos de dados em uma variável de memória:

```
. STORE " COMPUTADOR" TO GUARDA
. STORE Y TO VERDADE
. STORE 850 TO SALTOT
```


Tente reproduzir os exemplos acima e apresentar todas as variáveis de memória, digitando:

```
. DISPLAY MEMORY
GUARDA          (C) COMPUTADOR
VERDADE         (L) .T.
SALTOT          (N) 850
** TOTAL **  03 VARIABLES USED 0018 BYTES USED
```

Pode-se armazenar até sessenta e quatro variáveis de memória de um só vez. Para se contar com um número maior de variáveis deve-se, em primeiro lugar, preservar, em disco, as que estão em uso corrente.

Pode-se remover e armazenar dados em um arquivo.

O nome no registro nº 4 está armazenado com GUARDANOME.

```
. USE PESSOAL
. 4
. STORE NOME TO GUARDANOME
Pedro
```

Preservando Dados Intermediários no Disco

Ao se encerrar o dBASE, todas as variáveis de memória se perdem.

Querendo guardá-las para a próxima sessão do dBASE, deve-se preservá-las em um arquivo em disco dando um nome a ele:

```
SAVE TO AMANHA
```

Este arquivo de nome AMANHÃ.MEM é um arquivo especial de memória. O retorno deste último arquivo à memória poderá ser feito através de:

```
RESTORE FROM AMANHA
```

Depois de preservadas em disco, as variáveis de memória ainda permanecem na memória para serem usadas até que se encerre o dBASE (por meio do QUIT). Tendo utilizado todas as sessenta e quatro variações de memória, deve-se liberar as que não são mais necessárias (pelo comando RELEASE), a fim de dar lugar a outras.

Cancela todas as variáveis de memória.

Cancela a variável de memória à qual foi atribuído o nome.

Cancela as variáveis às quais foram atribuídos os nomes.

```
. RELEASE ALL
. RELEASE variavel
. RELEASE variavel1,variavel2,variavel3
```

Quando se introduz um comando RESTORE, todas as variáveis de memória que se encontram armazenadas são liberadas (pelo RELEASE) antes que o arquivo (de memória) seja novamente colocado na memória. A partir da versão 2.4 é possível fazer com que as variáveis armazenadas no arquivo de memória sejam acrescentadas às variáveis em uso que se encontram na memória, acrescentando ADDITIVE ao comando:

```
. RESTORE FROM AMANHA ADDITIVE
```

O ADDITIVE mantém intactas as variáveis de memória em curso e acrescenta à memória tantas variáveis de memória contidas no arquivo em disco quantas forem possível, até um total de sessenta e quatro.

O comando CLEAR liberará todas as variáveis de memória. Também fechará ao mesmo tempo todos os arquivos que se encontrarem abertos.

```
. CLEAR
```

A partir da versão 2.4, pode-se preservar (SAVE) e liberar (RELEASE) variáveis de memória com caracteres comuns em seus nomes.

*Os caracteres comuns são indicados por letras comuns ou por letras e números partilhados e pelo uso de um caractere-chave para caracteres que não sejam de interesse. O ? é um caractere-chave de um único caractere e se refere a qualquer caractere simples. O * é um caractere-chave para caracteres múltiplos e se refere a qualquer combinação de caracteres.*

caracteres comuns

```
. SAVE ALL LIKE
. RELEASE ALL LIKE
```

caracteres comuns

e pode-se liberar (RELEASE) variáveis de memória que não contenham caracteres comuns:

```
. RELEASE ALL EXCEPT
```

caracteres comuns

Por exemplo, as variáveis de memória do tipo SALTOT, QUANTOT e CONTOT seriam designadas por:

```
. SAVE ALL LIKE *TOT
```

TOT é comum, enquanto o resto não o é.

As variáveis de memória do tipo T3QTD, J3TOTAL, Z3QTD seriam designadas por:

```
. RELEASE ALL LIKE ?3*
```

O 3 na segunda posição é comum, mas o único caractere que antecede o 3 não o é, assim como não o são os caracteres que vêm depois do 3.

CAPÍTULO 17

TRABALHANDO COM MÚLTIPLOS ARQUIVOS

Mantendo Dois Arquivos Abertos Simultaneamente

É possível trabalhar com dois arquivos simultaneamente, chamando um deles de PRIMÁRIO e o outro, SECUNDÁRIO.

Pode-se ir de um ao outro, entre os dois arquivos, sem que eles tenham de ser abertos todas as vezes. Além disso, o dBASE guarda sua posição em ambos os arquivos, de tal modo que o ponteiro de registro permaneça exatamente no mesmo local onde é deixado, quando for necessário ir-se de um arquivo ao outro.

Trabalhar com dois arquivos é muito fácil. Abre-se um arquivo pelo processo costumeiro e, depois, seleciona-se o segundo arquivo como se segue:

```
. USE PESSOAL
. SELECT SECONDARY
. USE LUGARES
```

No exemplo acima, abriu-se o arquivo PESSOAL. Selecionando LUGARES para arquivo secundário, o PESSOAL passa a ser, automaticamente, o arquivo PRIMÁRIO. Permanecer em LUGARES até se digitar:

```
. SELECT PRIMARY
```

após o que retorna-se novamente ao PESSOAL

Para se andar de um para outro, entre os dois arquivos, entra-se com SELECT PRIMARY ou SELECT SECONDARY. O único ponto a ser lembrado é: qual o arquivo PRIMÁRIO e qual o SECUNDÁRIO.

Em caso de esquecimento, recorre-se ao DISPLAY STRUCTURE ou DISPLAY STATUS (na versão 2.4) para essa informação.

Pode-se usar arquivos adicionais utilizando simplesmente os arquivos através do USE. O arquivo recém-aberto transforma-se em PRIMÁRIO ou SECUNDÁRIO dependendo do modo adotado quando de sua abertura. Poder-se-ia continuar trabalhando em um arquivo principal, designado PRIMÁRIO, e seguir explorando as condições em vários arquivos SECUNDÁRIOS, sem "se perder a posição" no arquivo principal.

Conectando Dois Arquivos para Exibição e Relatório

Partindo de dois arquivos, as operações de impressão e de exibição tornam-se simultaneamente possíveis através da opção SET LINKAGE.

O que se deve fazer primeiro é:

```
SET LINKAGE ON
```

A seguir, abrem-se os dois arquivos conforme o indicado a seguir.

Se ambos os arquivos tiverem nomes de campo idênticos, poderão ser distinguidos por um prefixo P. ou S. Por exemplo: se os dois arquivos usarem COR como nome de campo, pode-se apresentá-lo por:

```
. USE ARQV1
. SELECT SECONDARY
. USE ARQV2
```

Executa-se, então, a listagem ou o relatório.

```
. DISPLAY ALL P.COR S.COR
```

Na realidade, o prefixo é necessário apenas para os dados não selecionados. No arquivo PRIMÁRIO, por exemplo, dispensam-se os prefixos P., enquanto os S. não são usados apenas em casos de nomes idênticos. Estando no arquivo SECUNDÁRIO, os únicos prefixos a serem usados serão os P., para distinguir os campos de arquivo de dados PRIMÁRIO que possuam o mesmo nome.

Terminando a listagem ou o relatório
não se esqueça de usar o SET LINKAGE OFF.

ATENÇÃO:

Para que este DISPLAY ligado funcione a contento, ambos os arquivos deverão conter o mesmo número de registros.

Além disso, tal opção destina-se a apenas uma passagem pelos arquivos, não servindo para o uso contínuo de percorrimento entre eles. Querendo um segundo relatório ou listagens, os arquivos deverão ser reabertos, a partir do início.

Usando o Comando Relacional JOIN para Combinar Arquivos

O dBASE é um sistema relacional de banco de dados, ou seja, os dados podem ser processados segundo formas diferentes que não precisam ser previamente determinadas.

O comando JOIN é a conhecida capacidade relacional que torna possível a verificação dos dados de um arquivo em relação aos de outro e também a criação de um registro quando as condições são atendidas. O JOIN é muito útil na combinação de dados visando relatórios e listagens. Também pode ser usado para catalogar e classificar dados.

Fazendo uso do arquivo PESSOAL para ilustrar o comando JOIN, suponha que se queira conceder um aumento salarial a todos que ganhem menos que $400 e que esta notícia será dada por telefone. Examine os dados contidos no seguinte arquivo denominado TELEFONE.

NOME	TELNO
Luiz	343-2323
João	832-3991
Paulo	593-2393
Pedro	892-0972
Lucio	439-2476

Através do JOIN, pode-se gerar um arquivo chamado AUMENTO que contém o nome e o número do telefone de todos os que atendem às condições salariais:

NOME (extraído de PESSOAL)	TELNO (extraído de TELEFONE)
Paulo	593-2393
João	832-3991

Para tanto, deve-se, primeiramente, abrir o arquivo PESSOAL:

```
. USE PESSOAL
```

Em seguida, abre-se o arquivo TELEFONE que atuará como arquivo SECUNDÁRIO:

```
. SELECT SECONDARY
. USE TELEFONE
```

Agora, devolve-se o controle ao arquivo PESSOAL (arquivo PRIMÁRIO):

```
. SELECT PRIMARY
```

O comando JOIN tem o seguinte aspecto:

```
. JOIN TO AUMENTO FOR SALARIO < 400  .AND. NOME=S.NOME FIELD NOME;
  TELNO
```

Registro Primário — Registro Secundário

Examinando este comando mais atentamente:

```
. JOIN TO AUMENTO
```

Esta parte especifica o nome do arquivo de saída, que neste caso é o AUMENTO.

```
FOR SALARIO < 400   .AND. NOME = S.NOME
```

Esta parte é a condição de verificação. Cada registro no arquivo PRIMÁRIO (PESSOAL) deverá atender à condição de o salário ser inferior a 400 e também encontrar, pelo nome, um registro no arquivo SECUNDÁRIO (TELEFONE). Cada registro no arquivo PRIMÁRIO será comparado a todos os registros no arquivo SECUNDÁRIO. Ocorrendo uma verificação, cria-se um novo registro para saída.

```
FIELD NOME,TELNO
```

Este trecho especifica os campos no registro de saída. Não se dispondo de uma expressão FIELD, o arquivo de saída conterá todos os campos possíveis dos arquivos PRIMÁRIO e SECUNDÁRIO, até um total de trinta e dois campos.

Para um outro exemplo do comando JOIN, examinar os dados do seguinte arquivo de móveis, que contém o peso de cada um de seus componentes:

Este arquivo é chamado MOBÍLIA:

PEÇA	PESO
Almofada	2
Piano	375
Secretária	75
Cadeira	12
Banco	9

Para se classificar essas peças do mobiliário, pode-se construir um arquivo para as categorias de variação de peso, examinando os dados no seguinte arquivo de categorias de peso:

Este arquivo é denominado ARQCAT.

MIN	MAX	CATEGORIA
1	25	Leve
26	75	Médio
76	150	Pesado
151	500	Muito pesado

Os dois arquivos poderiam ser unidos (pelo JOIN), produzindo o seguinte arquivo:

Este arquivo é denominado RESULTADO:

PEÇA	PESO	CATEGORIA
Almofada	2	Leve
Piano	375	Muito pesado
Secretária	75	Médio
Cadeira	12	Leve
Banco	9	Leve

Os passos para se executar o comando JOIN são:


```
. USE MOBILIA
. SELECT SECONDARY
. USE ARQCAT
. SELECT PRIMARY
. JOIN TO RESULT FOR PESO>MIN .AND. PESO<MAX FIELD PECA,;
PESO,CATEGORIA
```

Observar o uso do ponto e vírgula na interrupção da linha de comando.

Examinemos o comando JOIN:

Este trecho especifica o nome do arquivo de saída; neste caso, RESULT

Este trecho representa a condição de verificação: cada registro no arquivo PRIMÁRIO (MOBÍLIA) é comparado a todos do arquivo SECUNDÁRIO (ARQCAT). Ocorrendo uma verificação, cria-se um novo registro para a saída.

```
. JOIN TO RESULT FOR PESO>MIN .AND. PESO < MAX;
FIELD PECA,PESO,CATEGORIA
```

Este trecho especifica os campos no registro de saída.

Tal como no exemplo do telefone, usado para ilustrar o comando JOIN, deve-se lembrar que a referência aos campos do arquivo SECUNDÁRIO que apresentam nomes idênticos é feita através de um prefixo S. Assim:

```
. JOIN TO SAIDA FOR NOPECA =S.NOPECA
```

Como cada registro no arquivo PRIMÁRIO é comparado a todos os registros do arquivo SECUNDÁRIO, o comando JOIN é muito eficiente quando os arquivos não são muito grandes.

ATENÇÃO

O número máximo de registros que o comando JOIN poderia em princípio gerar corresponde ao total de registros do arquivo PRIMÁRIO multiplicado pelo total de registros do arquivo SECUNDÁRIO. Este valor poderá ultrapassar o máximo de 65535 registros permitidos em um arquivo dBASE.

CAPÍTULO 18

OPÇÕES PARA EXIBIÇÃO E FORMATAÇÃO ESPECIAIS

Extraindo Dados do Meio de um Campo

A possibilidade de se extrair dados do meio de um campo é, em geral, muito útil. Por exemplo, é possível querer exibir apenas o ano de uma data armazenada em um formato mês-dia-ano. Tecnicamente, esta função é chamada *substringsearch* – busca subseqüencial (string significa uma série de caractere –, e tal qual sua prima, a Varredura de Campos (discutida no Capítulo 10, junto com DISPLAY), trabalha apenas com os dados "C".

O formato é:

$ (dado, 1ª posição, número de posições)

Por exemplo, suponhamos que a data 12-11-42 esteja armazenada em um campo de oito caracteres chamado DATA. Pode-se retirar o ano de DATA, através de:

```
. STORE $(DATA,7,2) TO ANO
```

Ou o mês:

```
STORE $(DATA,1,2) TO MES
```

Querendo exibir estes dados em relatório, é preciso apenas de:

$ (DATA,7,2,) ou $ (DATA,1,2,)

como conteúdo das colunas.

Para fins de ordenação e de busca, as datas deverão ser armazenadas, no registro, segundo o formato ano-mês-dia. Porém, para facilitar a leitura nos relatórios o $ será provavelmente utilizado em datas com maior freqüência.

Pelo uso do sinal + como uma função de conexão (discutida neste capítulo), pode-se trocar uma data do tipo ano/mês/dia por outra, tipo dia/mês/ano, do seguinte modo:

```
STORE $(DATA,7,2)+'/'+$(DATA,4,2)+'/'+$(DATA,1,2) TO NOVADATA
```

A expressão acima, sem o STORE e o TO, funcionaria igualmente bem em uma coluna na definição de uma forma de relatório:

```
$(DATA,7,2)+'/'+$(DATA,4,2)+'/'+$(DATA,1,2)
```

Usa-se o $ apenas quando se quer ver uma pequena parte de um grande campo. Supondo que se deseje olhar os vinte primeiros caracteres em um campo de comentários muito maior:

```
. DISPLAY $(COMENTARIO,1,20)
```

Usando o $ para múltiplos campos, estes deverão ser separados por vírgulas:

```
. DISPLAY $(CAMPOA,1,10),$(CAMPOB,2,6)
```

Campos separados por vírgulas.

Conectando os Campos

Os dados do tipo caractere podem ser conectados para atender a vários modos de exibição e de relatórios. Tecnicamente, esta operação recebe o nome de concatenação.

A simbologia empregada na conexão (concatenação) é

+ conecta dois campos
– conecta dois campos e elimina quaisquer espaços em branco entre eles.

O símbolo – elimina os espaços em branco entre ambos os campos. Elimina os espaços em branco no final do primeiro campo e, também, do início do segundo. É a chamada "concatenação com branco espremido".

A função TRIM, que tem por finalidade eliminar os brancos residuais (*trailing blanks*) de um campo de caracteres também é, em geral, muito usada. Por exemplo, examine os três campos seguintes:

		Tamanho do Campo
CIDADE:	Resende	10 bytes
ESTADO:	RJ	2 bytes
CEP:	20000	5 bytes

Utilizando a função TRIM e o símbolo +, todos estes campos podem ser conectados em uma única coluna de relatório, fornecendo os seguintes resultados:

Resende, RJ 20000

Para tanto, deve-se entrar com os seguintes dados na coluna do relatório:

```
TRIM(CIDADE)+' , '+ESTADO+' '+CEP
```

Se o CEP fosse um campo numérico, os dados numéricos teriam de ser primeiramente convertidos em caracteres mediante a função STR (a ser discutida mais tarde neste capítulo), do seguinte modo:

```
TRIM(CIDADE)+' , '+ESTADO+' '+STR(CEP,5)
```

Os campos de primeiro e último nomes poderiam ser conectados através de:

```
TRIM(PRIMNOME)+' '+ULTNOME
```

ou

```
TRIM(ULTNOME)+' , '+PRIMNOME
```


Poder-se-ia conectar campos separados de datas, tais como:

```
MES+' - '+DIA+' - '+ANO
```

Não esquecer de que a concatenação exige campo "C".

Convertendo Dados Numéricos em Dados do Tipo Caractere

Conectar vários campos pode ser muito útil quando se pretende exibir ou imprimir os dados. A função de concatenação trabalha apenas com dados "C" no dBASE. Então, querendo exibir dados de caracteres concatenados com dados numéricos, surge um problema que pode ser resolvido através da função STR (vem de *STR*ing = seqüência de caracteres que pode ser manipulada como uma única unidade). O formato desta função é:

STR (dado numérico, tamanho, número opcional de decimais).

Ela funciona assim: havendo no registro um campo chamado PERCENT e desejando concatená-lo a um símbolo de porcentagem, como 58%, digita-se:

```
DISPLAY ALL STR(PORCENT,2)+'%'
```

ou STR (PORCENT,2)+"%" em uma coluna de relatório.

É muito mais fácil digitar:

```
DISPLAY ALL PORCENT '%'
```

ou fazer duas colunas no relatório: uma para PORCENTAGEM e outra para "%". Porém, o símbolo do percentual e o número ficarão afastados de um espaço.

Outra exigência incomum a que podem ficar sujeitas estas funções ocorre quando se torna necessário indexar dois campos de um arquivo segundo um mesmo índice. Aqui, outra vez a função + deverá ser usada e, se ambos os campos não forem "C" surgirá o mesmo problema. Suponha que se tenha um campo de caracteres chamado COR e um campo numérico, de um dígito, denominado MATIZ. Para criar, simultaneamente, um índice em ambos, digita-se:

```
INDEX ON COR + STR(MATIZ,1) TO nome do indice
```

Convertendo caixa-baixa em CAIXA-ALTA

Já se deve ter percebido que digitando:

```
. DISPLAY FOR CARGO = 'EDITOR'
```

verificam-se apenas os cargos escritos com as letras E,D,I,T,O e R em caixa-alta. Analogamente, "editor" e Editor" não são verificados.

Ao se misturar, nos registros, dados em caixa alta e baixa: (1) pode-se facilmente convertê-los em caixa-alta mediante função caixa-alta; ou (2) a função caixa-alta poderá ser usada cada vez que se buscar um dado.

O formato da função caixa-alta é: !(nome do campo)

Para se atender à disposição acima, de tal modo que EDITOR, editor, Editor ou até mesmo eDitoR, sejam verificados, deve-se digitar:

```
. DISPLAY FOR !(CARGO) = 'EDITOR'
```

Um ponto muito importante é que, usando a função caixa-alta, os dados que se inserem entre aspas, EDITOR neste caso, devem estar em caixa-alta. O que se faz através de !(CARGO) é informar ao dBASE para, primeiramente, passar cargo para caixa-alta antes de submetê-lo a uma verificação.

Os cargos do arquivo inteiro também poderão ser convertidos através de:

```
. REPLACE ALL CARGO WITH !(CARGO)
```

Enviando Códigos Especiais ao Terminal ou à Impressora

Tanto a tela do terminal quanto a impressora proporcionam vários modos de operação. Por exemplo, a tela pode apresentar vídeo reverso (preto e branco) ou diferentes intensidades de brilho.

Algumas impressoras imprimem através de tipos com formatos diferentes ou à alta velocidade valendo-se de "qualidade de máquina" ou à baixa velocidade com a impressão de qualidade da carta.

Estes modos são ativados por um ou mais caracteres que atuam como um código especial, quando transmitidos à tela ou à impressora. Os códigos são de três formatos: decimal, hexadecimal e ASCII – o que, na verdade, constitui três modos distintos de se apresentar a mesma coisa.

Deve-se usar apenas códigos decimais que são números variando de 0 a 255. Quaisquer códigos que incluam letras são, definitivamente, não-decimais. Os números sozinhos, porém, poderiam ser insuficientes para indicar com qual tipo de código se está trabalhando (de 0 a 99 pode ser decimal ou hexadecimal). Usa-se, geralmente, um código de escape (decimal 27) para se iniciar a seqüência. Assim, o código 27 seguido de outros números constitui uma boa indicação para um código decimal. Se os códigos forem hexadecimais, ou ASCII, será necessário consultar uma tabela de conversão de hexadecimal em decimal ou de ASCII em decimal.

Para dar saída a um caractere no terminal, emprega-se o comando ? e a função CHR (de caractere).

```
. ? CHR(27)
```

A saída é o decimal 27.

Sendo necessário dar saída a uma série de caracteres, pode-se concatená-los através do símbolo +:

```
. ? CHR(27) + CHR(42) + CHR(45)
```

Querendo controlar a impressora, introduz-se primeiramente o SET PRINT ON:

```
. SET PRINT ON
. ? CHR(27) + CHR(01)
```

Projetando as Telas Personalizadas para Entrada e Edição de Dados

As telas personalizadas para a entrada e a edição de dados podem solucionar os seguintes problemas:

1. Os nomes dos campos não descrevem os conteúdos dos campos de forma adequada.

2. A seqüência dos dados no documento-fonte não corresponde à dos campos no registro com que se está trabalhando.

3. Os registros são criados, mas apenas parte dos dados é introduzida exigindo, assim, muitos saltos ao longo do registro.

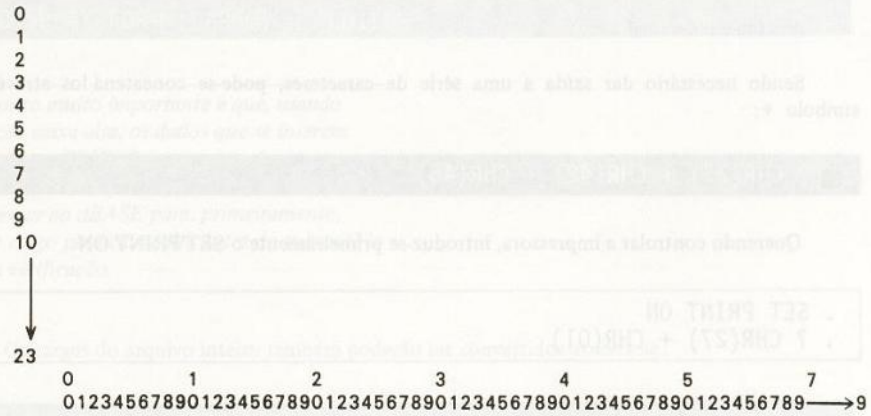
4. Deseja-se impedir algumas pessoas de lidarem com *todos* os dados, através da confecção de telas distintas para diferentes usuários, embora esta não seja uma medida de segurança garantida.

Pode-se projetar uma tela personalizada, capaz de exibir os dados na ordem desejada junto com as descrições de campo sobre qualquer assunto, sendo apresentados apenas os campos de interesse, ao invés do registro inteiro.

Usando o MODIFY COMMAND ou qualquer editor de texto, cria-se um arquivo dBASE especial, denominado arquivo de formato (tipo de arquivo .FMT), constituído de localização na tela, descrições e nomes de campo.

Uma localização na tela fica definida pelo número de sua linha e de sua coluna. A tela típica contém 24 linhas de 80 colunas. As linhas começam com o número 0 e terminam no 23, na parte superior e inferior da tela, respectivamente.

As colunas começam com 0 à esquerda e terminam com 79 à direita.



Cada linha do arquivo de formato começa por @ linha, coluna seguidos por um SAY ou GET.

Através do SAY exibe-se uma descrição do campo ou título da tela. A fim de se exibir "ENTRADA DE PEDIDOS" na linha 5, a partir da coluna 19, a linha teria o seguinte aspecto:

```
@ 5,19 SAY "ENTRADA DE PEDIDOS"
```


Querendo exibir um campo vazio destinado à entrada de dados, ou um campo de dados para edição, usa-se o GET:

```
@ 12,0 GET PRIMNOME
```

Para combinar a descrição do campo com o campo de dados, coloca-se o GET na mesma linha do SAY:

```
@ 12,0 SAY "Entre com o primeiro nome e a inicial do meio"
GET PRIMNOME
```

A tela exibiria, a partir da linha 12 e coluna 0:

```
Entre com o primeiro nome e a inicial do meio :
```

1. Não usar a linha 0.
2. O arquivo deverá conter o tipo de arquivo .FMT.
3. O arquivo deverá conter apenas linhas de @ linha, coluna, comandos SAY ou GET.

Tendo criado o arquivo de formato, deve-se usá-lo digitando:

```
. SET FORMAT TO nome de arquivo
```

A partir da versão 2.4, sempre que se tiver acesso aos modos APPEND ou EDIT, exibe-se o formato individual da tela, em vez de o formato normal do dBASE.

Mediante o comando READ, o arquivo de formato poderá ser usado para exibir um registro. Tendo introduzido SET FORMAT na tela personalizada de arquivo, pressiona-se:

```
. READ
```

e o registro em andamento será apresentado através da tela personalizada. Aqui, é preciso ter cuidado, pois a capacidade de edição ainda estará presente.

A seguir, temos um exemplo do formato de tela e do arquivo que lhe deu origem:

```
      0         1         2         3         4         5         6
      01234567890123456789012345678901234567890123456789012345678901234567890123456789
0
1     TELA DE ENTRADA DE DADOS              Registro # 00000
2
3
4     Entre com o primeiro nome :           :
5
6
7                        Entre com o ano de ingresso na companhia :      :
8
9
10    Qual o salario da pessoa? :           :          Entre com o cargo :
11
```

Conteúdos do arquivo de formato:

```
@ 1,0 SAY "TELA DE ENTRADA DE DADOS"
@ 1,34 SAY "Registro # "
@ 1,43 SAY #
@ 4,0 SAY "Entre com o primeiro nome " GET NOME
@ 7,13 SAY "Entre com o ano de ingresso na companhia " GET ANO
@ 10,0 SAY "Qual o salario da pessoa?" GET SALARIO
@ 10,42 SAY "Entre com o cargo " GET CARGO
```

CAPÍTULO 19

TRABALHANDO COM ARQUIVOS NÃO dBASE

Transformando os Dados do dBASE em um Relatório de Processador de Palavras

Pode-se converter um arquivo de dados dBASE (formato .dBF) em um arquivo-texto através do comando COPY e do SDF (System Data Format – Formato de Dados do Sistema). Por exemplo:

```
. USE PESSOAL
. COPY TO PESSOAS SDF
```

O arquivo PESSOAS contém apenas dados de caracteres, sem qualquer identificação especial do dBASE. O arquivo contém o tipo de arquivo .TXT (PESSOAS.TXT) e pode ser lido e editado por qualquer editor de texto ou processador de palavras que trabalhe com arquivos-texto padrão ASCII. Os arquivos TXT também podem ser lidos e processados por programas que tratem os registros como uma seqüência de caracteres contíguos.

Convertendo os arquivos dBASE em Outros Arquivos

Os arquivos de dados gerados por linguagens não dBASE (como o BASIC) formatam os registros colocando vírgulas entre os campos. Algumas vezes os campos de caracteres vêm entre aspas, ou um outro caractere é usado para delimitá-los.

Pode-se converter um arquivo dBASE neste formato usando comando COPY e o formato DELIMITED. Há três opções para o tipo de formato DELIMITED:

1. DELIMITED WITH, = as vírgulas separam todos os campos.
2. DELIMITED WITH x = as vírgulas separam todos os campos e o delimitador escolhido (x) envolve os campos de caracteres.

3. DELIMITED = as vírgulas separam todos os campos e as aspas simples envolvem todos os campos de caracteres.

Os formatos DELIMITED são criados mediante o comando COPY:

```
. USE PESSOAL
. COPY TO PESSOAS DELIMITED WITH ,
```

ou

```
. COPY TO PESSOAS DELIMITED
```

Estes arquivos contêm também o tipo de arquivo .TXT e podem ser lidos e editados por qualquer editor de texto ou processador de palavras.

Convertendo Outros Arquivos em Arquivos dBASE

Querendo converter os arquivos TXT novamente em arquivos dBASE, deve-se usar o comando APPEND FROM e um dos formatos SDF ou DELIMITED, como segue:

```
. USE PESSOAL
. APPEND FROM PESSOAS SDF
```

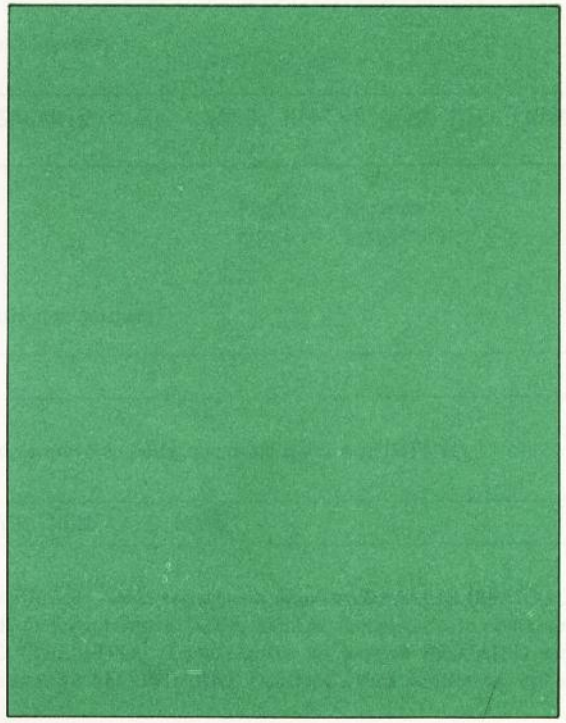
ou

```
. USE PESSOAL
. APPEND FROM PESSOAS DELIMITED
```

O exemplo acima constitui a forma única do comando APPEND DELIMITED. O arquivo aceita todos os formatos DELIMITED gerados pelo comando COPY DELIMITED. Porém, se os campos de caracteres forem delimitados por símbolos diferentes de aspas, os delimitadores aparecerão como dados nos campos de dados.

PARTE 6

UM GOSTINHO DA PROGRAMAÇÃO EM dBASE



CAPÍTULO 20

CONSTRUÇÃO DE UM PROGRAMA EM dBASE

Além dos comandos poderosos que apresenta no Nível 1, o dBASE é também uma linguagem estruturada de programação. Ao se escrever um programa em dBASE deve-se primeiro estruturá-lo logicamente para, depois, codificá-lo nesta linguagem.

As linguagens estruturadas diferem de outras como BASIC e COBOL. Com BASIC, por exemplo, o programa é escrito instantaneamente. O usuário apenas tem que se sentar ao terminal e começar a digitar. O mesmo não ocorre com o dBASE, a menos que o programa seja relativamente simples. A vantagem de uma linguagem estruturada como o dBASE surge mais tarde, quando se é necessário alterar o programa. A lógica de um programa estruturado é mais fácil de ser seguida. Em termos simples, sua lógica é mais lógica.

Os programas em dBASE são construídos por meio de loops (comando DO WHILE). Um loop é uma condição que continua a ser repetida até surgir uma razão que a faça parar. Por exemplo, imprimindo um arquivo de registros, prossegue-se a impressão até que os registros se esgotem. Executando alterações em um arquivo, continua-se a executá-las até que não haja mais alterações a serem feitas. Esta ação é o loop maior ou principal em um programa. Em geral, o loop principal realiza a entrada dos dados, processa-os e lhes dá saída vezes e vezes consecutivas.

Dentro do loop do programa principal há um ou mais loops menores. Todas as instruções que puderem ser repetidas várias vezes formarão um loop. O exemplo seguinte é um programa de impressão de fatura. O loop principal é "Imprimir Fatura" e o menor (loop aninhado) é "Linha do Artigo".


```
LOOP PRINCIPAL
|
|  FATURA
|  DADOS SOBRE O CLIENTE                    DADOS SOBRE O TRANSPORTE
|  XXXXXXXXXXXXXXXXXXXX                     XXXXXXXXXXXXXXXXXXXXXXXX
|  XXXXXXXXXXXXXXXXXXXX                     XXXXXXXXXXXXXXXXXXXXXXXX
|  XXXXXXXXXXXXXXXXXXXX                     XXXXXXXXXXXXXXXXXXXXXXXX
|
|  PEDIDO Nº. xxxxxxxx      COMO FOI TRANSPORTADO xxxxxxxxx
|
|  QUANTIDADE  Nº DO PRODUTO  DESCRIÇÃO  PREÇO  TOTAL
|  xxxxxxxxx   xxxxxxxxxxx    xxxxxxxxx  xxxxx  xxxxx
|  xxxxxxxxx   xxxxxxxxxxx    xxxxxxxxx  xxxxx  xxxxx
|  xxxxxxxxx   xxxxxxxxxxx    xxxxxxxxx  xxxxx  xxxxx
|
|  Imprimir dados sobre o cliente e o transporte
|
|  Imprimir linha
|
|  LOOP ANINHADO
|  |
|  |  Calcular e imprimir linha do artigo
|  |
|
|  Calcular/Imprimir o total
|
|                                                $ xxxxxxxxxxxxxx
```

A lógica do programa é constituída por loops contidos em loops, denominados loops "aninhados". O que a programação no Nível 2 do dBASE faz, na realidade, é aninhar os loops DO.

Depois de se executar alguns destes loops aninhados é que se dá início à apreciação da lógica embutida no Nível 1.

Exemplo de Programa de Mala Direta

Este programa é apresentado como um exemplo da programação em dBASE. Para o seu próprio uso, o disco do dBASE II fornece programas mais sofisticados para etiquetas.

Apresentamos, a seguir, um exemplo de Programa para Mala Direta que imprime uma etiqueta de cada vez. Ele não testa os campos em branco; para um cargo ou endereço em branco, a linha de impressão também é mantida em branco. Todos os campos do registro são "C", incluindo o Código Postal. Ao se executar este programa todos os comentários deverão ser retirados.

```
SET TALK OFF                          Desligar mensagem interativa
SET PRINT ON                          Ligar impressora
USE ARQNOME                           Abrir arquivo
DO WHILE .NOT. EOF                    Executar DO ate chegar ao final de arquivo
  ? NOME                              Imprimir nome
  ? CARGO                                cargo
  ? COMPANHIA                            companhia
  ? ENDERECO                             endereco
  ? CIDADE+' '+ESTADO+' 'CEP             cidade,estado,cep
  ?
  ?                                   Saltar linhas
  ?                                      para
  ?                                   proxima etiqueta
  STORE 1+CONTADOR TO CONTADOR        Contar etiquetas
  SKIP                                Obter o proximo registro
ENDDO                                 Ir para o inicio do loop
SET PRINT OFF                         Desligar a impressora
ERASE                                 Limpar a tela
? ' TOTAL DE ETIQUETAS IMPRESSAS'     Exibir a contagem das
?? CONTADOR                           etiquetas impressas
SET TALK ON                           Ligar, novamente, as mensagens interativas
CLEAR                                 Fechar arquivos e liberar variaveis de memoria
RETURN                                Fazer retornar o ponto
```

Exemplo de Programas de Entrada de Dados

O programa a seguir, relativo à entrada de dados, cria um registro para o arquivo PESSOAL, valida os dados segundo critérios abaixo:

1. O salário deve variar de 100.00 a 200.00.
2. O ano deve variar de 1950 a 1989.
3. Os cargos devem ser Editor, Escritor, Vice Pres, Pres ou Escriturário.

Gera-se a seguinte tela de entrada de dados para este programa:


```
PROGRAMA DE ENTRADA DE DADOS PARA PESSOAL

Entrar com os seguintes dados:

Nome:              :  Salario:        :  Ano:    :  Cargo:

Salario errado          Ano errado          Cargo errado
Este e o ultimo registro? (S/N)  :  :
```

O programa para a entrada de dados contém dois loops. O principal continua com o processamento até o usuário entrar com um S em resposta a: Este é o último registro? O segundo loop é de erro, que continua a obter e a validar a entrada até não haver mais erros.

Programa dBASE de Entrada de Dados

```
SET TALK OFF
USE PESSOAL
ERASE
@ 1,0 SAY  'PROGRAMA DE ENTRADA DE DADOS PARA PESSOAL'
@ 3,0 SAY  'Entrar com os seguintes dados: '
@ 5,0 SAY  'Nome           Salario          Ano          Cargo
@ 11,0 SAY 'Este e o ultimo registro (S/N)'
STORE '  ' TO FIM                                  = Definir indicador de termino
    DO WHILE FIM <>'S'                             = Executar o Do ate o usuario indicar o termino
    STORE '  ' TO INNOME, INSALARIO                = Criar limites para a entrada
    STORE '   ' TO INANO
    STORE '      ' TO INCARGO
    STORE '  ' TO FIM                              = Definir indicador de erro
    STORE Y TO ERRO                                = Executar o Do ate a entrada estar correta
      DO WHILE ERRO                                = Reajustar indicador de erro
      STORE N TO ERRO                              = Definir
      @ 5,5 GET INNOME                                localizacoes
      @ 5,24 GET INSALARIO                             de entrada
      @ 5,41 GET INANO                                  na tela
      @ 5,56 GET INCARGO
      @ 11,33 GET FINISHED                         = Ativar a tela
      READ                                         = Apagar a linha de mensagem de erro
       @ 8,0                                       = Converter a entrada em numerico
       STORE VAL (INANO) TO ANOAUX                 = Testar o salario correto
       IF SALARIOAUX < 100.00 .OR. SALARIO > 2000
       STORE Y TO ERRO
       @ 8,0 SAY 'Salario errado'.
       ENDIF
       STORE VAL (INSALARIO) TO SALARIOAUX         = Converter a entrada em numerico
       IF ANOAUX<1950 .OR. ANOAUX >1984            = Testar o ano correto
       STORE Y TO ERRO
       @ 8,23 SAY 'Ano Errado'.
       ENDIF
       IF INCARGO<>'Escritor'.AND.INCARGO<>;       = Testar o cargo correto
       'Editor' .AND. INCARGO <>'Escriturario'.deAND.;
       INCARGO<>'Pres'  .AND. INCARGO<>'Vice Pres'
       STORE Y TO ERRO
       @ 8,42 SAY 'Cargo errado'
       ENDIF
       ENDDO                                       = Voltar ao inicio do loop de erro.
      APPEND BLANK                                 = Criar novo registro em branco
      REPLACE NOME WITH INNOME, CARGO WITH INCARGO = Preencher registro em branco
      REPLACE SALARIO WITH SALARIOAUX, ANO WITH ANOAUX    com dados validos
      ENDDO                                        = Voltar ao inicio do loop principal
SET TALK ON                                        = Ligar, novamente, as mensagens interativas
ERASE                                              = Limpar a tela
RETURN                                             = Fazer voltar o ponto
```

Sumário dos Comandos do NÍVEL 2

A apresentação resumida dos comandos e funções do Nível 2 que vem a seguir deverá ser cuidadosamente examinada por aqueles que desejarem tornar-se programadores em dBASE. Tais programadores costumam escrever seus programas combinando vários comandos do Nível 1 com as informações que se seguem.

Para explicações mais detalhadas sobre o uso destes comandos, deve-se consultar o manual de documentação do dBASE.

COMANDO	PERMITE
? ?	Obter dados de saída ou resultados na mesma linha da saída anterior.

@ linha, coluna SAY
@ linha, coluna GET
	Definir, através de linhas e colunas, a localização da exibição de mensagens na tela e receber entradas feitas pelo usuário. Também é usado para criar saídas impressas personalizadas.
	
	SAY dá saída a mensagens e GET define uma variável de memória ou campo de dados. READ ativa imediatamente todos os GETs anteriores e permite a entrada/ edição na tela. Por exemplo:
	
	`@ 3,0 SAY "Somarei 2 numeros para voce"`
	`@ 5,0 SAY "Entrar com Primeiro Numero" GET PRIMEIRO`
	`@ 7,0 SAY "Entrar com Segundo Numero" GET SEGUNDO`
	`READ`
	`@ 9,0 SAY "Sua Resposta e"`
	`@ 9,15 SAY PRIMEIRO + SEGUNDO`
	
	(Quando o READ é executado, o cursor se desloca para o primeiro GET na tela. O READ é concluído quando o cursor passa pelo último GET na tela ou quando se pressiona <RETURN> no último GET.)

ACCEPT	Exibir mensagem na tela e obter entrada feita pelo usuário. O ACCEPT aceita qualquer entrada e determina automaticamente seu tipo. Por exemplo: ACCEPT "Entrar com dado" TO ENTRADA

APPEND BLANK	Anexar um registro em branco ao final do arquivo. A entrada de dados é usualmente feita através dos comandos GET e READ. Depois de validada, ela entra, em geral, em um registro em branco (criado pelo APPEND BLANK) através do comando REPLACE.
CALL	Desvia para a localização real da memória, definida no comando SET CALL.
CANCEL	Cancelar o arquivo de comando em execução.
CLEAR GETS	Limpar todos os GETs pendentes.

DO CASE	Testar várias condições. Por exemplo:
	
	`DO CASE`
	`CASE COR= 'VERMELHO' .AND. MATIZ= 'ESCURO'`
	`? 'Quente'`
	`CASE COR= 'AZUL' .AND. MATIZ= 'CLARO'`
	`? 'Frio'`
	`OTHERWISE`
	`? 'Nenhum Vermelho Escuro,Nenhum Azul Claro'`
	`ENDCASE`

DO WHILE	Definir um loop DO. O loop DO é a estrutura lógica fundamental de um programa dBASE. Há sempre um loop principal em um programa que, geralmente, engloba uma série de loops menores. O exemplo seguinte é o de um loop principal e único em um programa simples.
	
	`USE CORES`
	`DO WHILE .NOT. EOF`
	`IF COR = 'VERMELHO'`
	`STORE 1 + VERMELHO TO VERMELHO`
	`IF COR = 'AZUL'`
	`STORE 1 + AZULTOTAL TO AZULTOTAL`
	`ELSE`
	`STORE 1 + OUTROTOT TO OUTROTOT`
	`ENDIF`
	`ENDIF`
	`SKIP`
	`ENDDO`

ELSE Definir as instruções a serem tomadas se as condições IF não forem verdadeiras.

ENDCASE Definir o final de CASE.

ENDDO Definir o final do DO.

ENDIF Definir o final do IF.

GO ou GOTO Deslocar o ponteiro para um registro específico.

Esta não é uma instrução de loop, de ramificação ou de salto, como em BASIC ou COBOL. Em dBASE, os loops são realizados através do comando DO WHILE, ou chamando-se outros arquivos de comando como rotinas menores com o programa.

IF Testar uma ou mais condições. Por exemplo:

```
IF COR='VERMELHO' .AND. MATIZ='ESCURO'
   ? 'Quente'
   IF COR='AZUL' .AND. MATIZ='CLARO'
   ? 'Frio'
    ELSE
   ? 'Nenhum Vermelho Escuro,Nenhum Azul Claro'
   ENDIF
ENDIF
```

INPUT Exibir mensagens na tela e receber entradas feitas pelo usuário. O INPUT exige que os dados de caracteres venham entre aspas. Por exemplo:

INPUT "Entrar com dados alfabéticos entre aspas" TO DADOS.

LOAD Carregar um programa de linguagem de máquina contido em disco.

LOOP Forçar um retorno ao início do loop DO.

NOTE (ou *) Definir uma linha de comentários do programa.

OTHERWISE Definir as instruções a serem tomadas se as condições CASE não forem verdadeiras.

PEEK address Exibir valor decimal de números binários sem sinal, armazenado em um byte específico.

POKE address, byte Armazenar dados na memória.
byte

READ Ativar entrada de dados.

REMARK Dar saída, na tela, a quaisquer caracteres.

RETURN Definir o fim de um arquivo de comando dBASE (programa).

SET BELL (ON) OFF Tocar campainha ao inserir com dados não válidos. A campainha poderá ser desligada, se assim for desejado.

SET COLON (ON) OFF Usar dois pontos para limitar os campos na tela, para entrada de terminal especificada por um GET.

SET CONSOLE (ON) Ativar tela do terminal. A saída pela tela poderá ser desligada, se assim se desejar.

SET DEBUG ON (OFF) Exibir todos os comandos de um arquivo de comando (usado para depurar um programa).

SET ECHO ON (OFF) Ligar as saídas ECHO e STEP à impressora.

SET ESCAPE (ON) OFF — Cancelar um arquivo de comando através da tecla ESC. O modo cancelar poderá ser desligado a fim de não se cancelar, acidentalmente, um programa dBASE em execução.

SET INTENSITY (ON) OFF — Exibir intensidade dupla para o modo full-screen. A intensidade dupla poderá ser desligada, se assim desejar.

SET FORMAT TO (SCREEN) PRINTER — Enviar a saída do comando SAY para a tela ou impressora.

SET STEP ON (OFF) — Pára depois que cada comando for executado (usado quando se está depurando um programa).

SET RAW ON (OFF) — Exibir dados sem deixar espaço entre os campos. As exibições normais anexam um espaço em branco entre os campos para tornar a leitura mais fácil. Poderá ser desligado, caso assim se desejar.

SET TALK (ON) OFF — Estabelecer respostas interativas. Em geral, as respostas interativas são desligadas em um programa dBASE.

SKIP — Deslocar o ponteiro de registro em um arquivo. Por exemplo:

SKIP, sozinho, avança 1 registro.
SKIP + 12 avança 12 registros.
SKIP – 3 retrocede 3 registros.

TEXT — Dar saída ao texto que o segue. Por exemplo:

```
TEXT
    Esta é uma forma adequada de se exibir ou imprimir longas
    mensagens em dBASE.
END TEXT
```


WAIT — Parar temporariamente o programa e receber, do usuário, um caractere de entrada. Por exemplo:

```
WAIT TO OQUEMAIS
```

Sumário das Funções do Nível 2

SÍMBOLO	FUNÇÃO	PERMITE
&	Substituição (Macro)	Referir-se a dados ou comandos armazenados em quaisquer outros lugares. Por exemplo, recebe-se, do usuário, um nome de arquivo que é armazenado em NOMEARQ. O programa reporta-se a &NOMEARQ e o dBASE faz a substituição. Exemplos: USE &NOMEARQ COPY TO &NOMEARQ
@	Busca Subseqüencial	Localizar onde os caracteres se encontram. O Formato é @ (o quê, onde). Por exemplo: ? @("xy", "abxyz") dá saída a 3. XY está localizado na posição 3.
EOF	Fim do arquivo	Testar se o arquivo está no fim. Por exemplo: DO WHILE .NOT. EOF
FILE	Teste de arquivo	Testar a existência de um arquivo em disco. Por exemplo: ? FILE ("PESSOAL") dá saída a .T ou .F.
INT	Inteiro	Retirar as decimais de um número. Por exemplo: ? INT (454.33) dá saída a 454 (número inteiro).

LEN	Comprimento	Determinar o comprimento de uma seqüência de caracteres. Por exemplo: ? LEN ("COMPUTADOR") dá saída a 10.
RANK	Valorização de uma ordem	Determinar o valor numérico de um caractere. Por exemplo: ? RANK ("A") dá saída a 65. A é o binário 65, B é 66, C é 67 etc.
TRIM	Suprimir espaços em branco	Eliminar os espaços em branco "residuais de uma seqüência de caracteres". Por exemplo: ? TRIM ("RIO) dá saída a "RIO".
TYPE	Teste do tipo	Determina o tipo de dados. Por exemplo: ? TYPE (1800.45) dá saída a um N. As saídas são C, N e L e U, para os não determináveis.
VAL	Conversão de caracteres para numérico	Converter dados de caracteres em dados numéricos. O formato é VAL (dados de caracteres). ? VAL ("454.42") dá saída a 454.42.

APÊNDICES

APÊNDICE A

Sumário das Especificações do dBASE

Arquivos de Dados

Número máximo de registros em um arquivo	65 535
Número máximo de caracteres em um registro	1 000
Número máximo de campos em um registro	32
Número máximo de caracteres em um campo	254
Número máximo de arquivos abertos de uma só vez	2

Validações dos Tipos de Dados

Dados de caracteres (todos os caracteres do teclado que podem ser impressos, incluíndo símbolos, números inteiros e espaços).
Dados numéricos (dígitos, ponto decimal e sinal negativo).
Dados lógicos (T,t,F,f,Y,y,N,n).

Índices

Número máximo de índices que podem ser atualizados por arquivo	7
Número máximo de caracteres em uma chave de índice	99

Relatórios

Número máximo de caracteres em título de relatório	254
Número máximo de campos em um relatório	24

Números

Precisão dos campos numéricos	16 dígitos
Maior número positivo	$1{,}8 \times 10^{63}$, aproximadamente
Menor número negativo	$1{,}0 \times 10^{63}$, aproximadamente
Número máximo de campos que podem ser somados através do comando SUM	5

Outros

Número máximo de caracteres por linha de comando	254
Número máximo de todos os tipos de arquivos que podem estar abertos ao mesmo tempo	16

Tipos de Arquivos Gerados pelo dBASE

DBF – arquivo Data Base (arquivo de Banco de Dados)
FRM – arquivo de relatório
CMD ou PRG – arquivo de comando
NDX – arquivo de índice
MEM – arquivo de memória
TXT – arquivo de texto
FMT – arquivo de formação

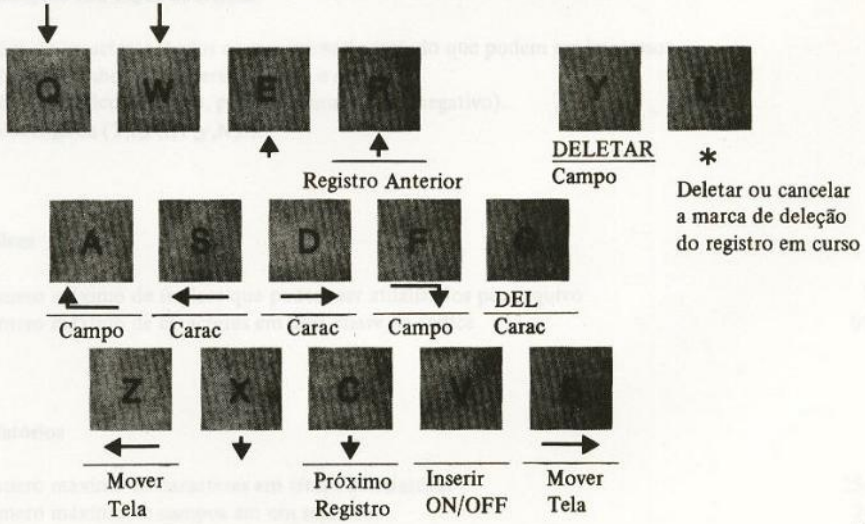
APÊNDICE B

Sumário dos Comandos das Teclas de Controle para Entrada e Edição de Dados

Modos APPEND, INSERT, EDIT e BROWSE

Pressionar Ctrl com as seguintes teclas:

Abandonar ou Preservar alterações no registro em curso e trazer o ponto de volta



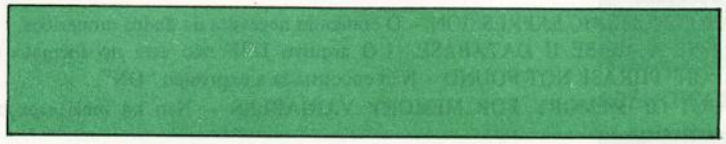
Estas teclas controladoras de comando requerem operações em full-screen. Como a maioria dos computadores permite tal tipo de operação e o funcionamento normal do dBASE é em FULL SCREEN ON, não há, em geral, necessidade de qualquer cuidado a esse respeito. Se, porém, as operações em full-screen tiverem sido desligadas, religá-las através de:

```
SET SCREEN ON
```

antes de se ter acesso aos modos APPEND, EDIT, INSERT ou BROWSE.

APÊNDICE C

Sumário das Mensagens de Erro em dBASE



BAD DECIMAL WIDTH FIELD – Número de casas decimais especificado de forma incorreta.
BADFILENAME – Nome de arquivo não especificado corretamente.
BAD TYPE FIELD – Tipo de campo diferente de C, N, ou L.
BAD WIDTH FIELD – Erro na largura do campo (deve ser de 1 a 255).
** BEYOND STRING – A busca subseqüencial refere-se a caracteres inexistentes.
CANNOT INSERT – THERE ARE NO RECORDS IN DATABASE FILE – Não se pode inserir = Não há registros no arquivo DATABASE.
CANNOT OPEN FILE – O tipo de arquivo no disco não está correto.
COMMAND FILE CANNOT BE FOUND – O arquivo de comando não pode ser encontrado.
DATA ITEM NOT FOUND – Dados não encontrados.
DATABASE IN USE IS NOT INDEXED – O banco de dados em uso não está indexado.
DIRECTORY IS FULL – Não há espaço, no disco, para mais títulos de arquivos.
DISK IS FULL – Não há mais espaço no disco.
END OF FILE FOUND UNEXPECTEDLY – Erro na seqüência do arquivo.
"FIELD" PHRASE NOT FOUND – Não encontrada a expressão "FIELD".
FILE ALREADY EXISTS – O nome do arquivo já foi criado.
FILE DOES NOT EXIST – Não existe o arquivo.
FILE IS CURRENTLY OPEN – O comando não pode funcionar se o arquivo estiver aberto.
FORMAT FILE CANNOT BE OPENED .FMT – Arquivo não encontrado.
FORMAT FILE HAS NOT BEEN SET – O arquivo de formato não foi estabelecido.
ILLEGAL DATA TYPE – Tipo de campo incorreto.
ILLEGAL GOTO VALUE – Número não válido de registro.
ILLEGAL VARIABLE NAME – Nome não válido.
INDEX DOES NOT MATCH DATABASE – Especificação incorreta do índice.
INDEX FILE CANNOT BE OPENED – Arquivo NDX não encontrado.
JOIN ATTEMPTED TO GENERATE MORE THAN 65534 RECORDS – A união, através do comando JOIN, tentou gerar mais do que 65534 registros.
KEYS ARE NOT THE SAME LENGTH – Campos-chave com comprimentos diferentes nos dois arquivos.

MACRO IS NOT A CHARACTER STRING – & trabalha apenas com dados de caracteres.
MORE THAN 5 FIELDS TO SUM – Comando-Soma limitado a 5 campos.
MORE THAN 7 INDEX FILE SELECTED – O dBASE trabalha com um máximo de 7 índices.
NESTING LIMIT VIOLATION EXCEEDED – Ultrapassado o máximo de arquivos abertos.
NO EXPRESSION TO SUM – Falta parte do comando SUM.
NO "FOR" PHRASE – Sem a expressão "FOR".
NO "FROM" PHRASE – Sem a expressão "FROM".
NO FIND – Sem se encontrar um registro pelo comando FIND, o número do registro passa a ser zero.
NON-NUMERIC EXPRESSION – O comando necessita de dados numéricos.
NOT A dBASE II DATABASE – O arquivo DBF não está no formato correto do dBASE.
"ON" PHRASE NOT FOUND – Não encontrada a expressão "ON".
OUT OF MEMORY FOR MEMORY VARIABLES – Não há mais espaço na memória para variáveis.
RECORD LENGTH EXCEEDS MAXIMUM SIZE (of 1000) – Comprimento de registro excede o tamanho máximo (1000).
RECORD NOT IN INDEX – Índice não atualizado pelo arquivo de dados.
RECORD OUT OF RANGE – O número do registro não existe no arquivo.
SORTER INTERNAL ERROR, NOTIFY SCDP – Erro interno. Entrar em contato com a Datalógica.
SOURCE AND DESTINATION DATA TYPES ARE DIFFERENT – Tipos distintos de dados.
*** SYNTAX ERROR – Confusão na linha de comando ou grupo de sintaxe.
SYNTAX ERROR IN FORMAT SPECIFICATION – Erro no comando @ linha, coluna.
SYNTAX ERROR, RE-ENTER – Entrada incorreta.
"TO" PHRASE NOT FOUND – Não encontrada a expressão "TO".
TOO MANY CHARACTERS – Dados de entrada muito longos.
TOO MANY FILES ARE OPEN – Ultrapassa o nível máximo de 16 arquivos.
TOO MANY MEMORY VARIABLES – Ultrapassado o máximo de 64 variáveis de memória.
TOO MANY RETURNS ENCOUNTERED – Erro na estrutura do arquivo de comando.
"WITH" PHRASE NOT FOUND – Não encontrada a expressão "WITH".
UNASSIGNED FILE NUMBER – Erro interno. Entrar em contacto com a Datalógica.
*** UNKNOWN COMMAND – Comando não válido.
VARIABLE CANNOT BE FOUND – A variável não pode ser encontrada.
*** ZERO DIVIDE – Não se pode dividir por zero.

APÊNDICE D

Sumário dos Tipos de Arquivos dBASE

Os comandos dBASE geram os sete seguintes tipos de arquivos:

DBF 1. O ARQUIVO DATABASE (DATABASE FILE) – nome do arquivo .DBF

O arquivo DBF é o arquivo de dados do próprio dBASE, gerado quando se tem acesso a um dos seguintes comandos: CREATE, COPY ou SORT. É o único arquivo em que se pode aplicar o comando USE e é o tipo de arquivo adotado ao se executar um APPEND FROM.

FRM 2. O ARQUIVO DA FORMA DO RELATÓRIO (REPORT FORM FILE) – nome do arquivo .FRM

O arquivo FRM é automaticamente criado quando se descreve um RELATÓRIO para o dBASE. Depois, o comando REPORT usará todas as descrições contidas neste arquivo.

NDX 3. O ARQUIVO DE ÍNDICE (INDEX FILE) – nome do arquivo .NDX

Criado quando se quer indexar um arquivo. Pode-se atribuir ao arquivo de índice o mesmo nome do arquivo de banco de dados pois, embora com nomes idênticos, os tipos de arquivos serão diferentes. Por exemplo:

USE PESSOAL INDEX PESSOAL

O comando USE refere-se ao PESSOAL.DBF, enquanto INDEX refere-se a PESSOAL.NDX.

CMD 4. O ARQUIVO DE COMANDO (COMMAND FILE) – nome do arquivo .CMD. ou nome do arquivo .PRG

Usando em dBASE, o comando MODIFY COMMAND para construir um arquivo de comando, o tipo de arquivo CMD (ou PRG para o MS-DOS) é automaticamente anexado ao nome do arquivo de comando. Ao se executar um arquivo de comando através do DO, o dBASE procurará o tipo específico de arquivo. *Observação:* deve-se anexar o tipo adequado de arquivo aos arquivos de comando quando eles são construídos através de editores de texto não dBASE.

MEM 5. ARQUIVO DE MEMÓRIA (MEMORY FILE) – nome do arquivo .MEM

Pode-se armazenar dados na memória e preservá-los em um arquivo em disco. O arquivo especial criado por um comando SAVE é o arquivo .MEM. Quando se usa RESTORE FROM nome do arquivo, está-se recuperando o arquivo .MEM anteriormente preservado.

TXT 6. O ARQUIVO-TEXTO (TEXT FILE) – nome do arquivo .TXT

Todas as saídas que vão para a tela também podem ser enviadas para o disco, bastando, para tanto, criar um arquivo substituto através do comando SET ALTERNATE TO. Gera-se, para este arquivo alternativo de saída, um tipo de arquivo. TXT., que também é gerado quando se criam arquivos não dBASE, tais como os obtidos pelos comandos COPY SDF ou COPY DELIMITED (ver Capítulo 19 – Trabalhando com Arquivos não dBASE).

FMT 7. O ARQUIVO DE FORMATO (FORMAT FILE) – nome do arquivo .FMT

O arquivo de formato é utilizado para se desenvolver telas personalizadas de entrada e de edição de dados. Os arquivos de formato contêm apenas localizações de tela, descrições e campos de dados. Eles são criados através de um editor de texto e permitem que se projete uma tela do modo desejado (ver Capítulo 18).

APÊNDICE E

Sumário da Sintaxe do Nível 1

Este apêndice apresenta exemplos de todas as variações de sintaxe dos comandos do Nível 1. As palavras em maiúsculo pertencem à linguagem dBASE. As palavras em minúsculo são nomes de arquivos, campos, número de registros ou dados.

Explicações das referências:

arquivo	nome do arquivo de dados (.DBF).
campo	nome do campo; campos separados por vírgulas, exceto para o DISPLAY.
memvar	nome da variável de memória; memvar separado por vírgulas.
condição	condição de verificação.
#	número do registro.
valor	campo ou dados.
[ON]	no funcionamento normal do dBASE está ligado.
[OFF]	no funcionamento normal do dBASE está desligado.

? campo(s)	exibir campos de dados no registro em curso.
? memvar(s)	exibir variáveis de memória.
? *	exibir situação deletada do registro em curso.
? #	exibir o número do registro em curso.
? DATE ( )	exibir a data do sistema.
? CHR (n)	exibir o caractere equivalente do número.
? resultados	exibir os resultados dos cálculos.

APPEND	anexar novos registros ao arquivo; introduzir o modo de entrada de dados.
APPEND FROM arquivo APPEND FROM arquivo FOR condição	anexar um arquivo de dados a outro arquivo de dados.
APPEND FROM arquivo TXT SDF APPEND FROM arquivo TXT DELIMITED APPEND FROM arquivo TXT SDF FOR condição APPEND FROM arquivo TXT DELIMITED FOR condição	converter e anexar o arquivo não dBASE (tipo de arquivo TXT) ao arquivo dBASE.
BROWSE BROWSE FIELD campo(s)	entrar com a combinação dos modos exibir/editar
CHANGE FIELD campo(s) FOR condição	rever/editar registros selecionados
CLEAR	fechar arquivos e deletar todas as variáveis de memória.
CONTINUE	continuar a busca do próximo registro utilizando o comando LOCATE
COPY TO arquivo COPY TO arquivo FOR condição COPY TO arquivo NEXT número COPY TO arquivo NEXT número FOR condição	criar cópia de arquivo
COPY TO arquivo SDF FOR condição COPY TO arquivo DELIMITED FOR condição COPY TO arquivo DELIMITED WITH, FOR condição COPY TO arquivo NEXT número SDF FOR condição COPY TO arquivo NEXT número DELIMITED FOR condição COPY TO arquivo NEXT número DELIMITED WITH, FOR condição	criar cópia não dBASE de arquivo
COPY STRUCTURE TO arquivo COPY STRUCTURE TO arquivo FIELD campo(s)	definir um novo arquivo a partir de um já existente
COUNT COUNT FOR condição COUNT TO memvar COUNT TO memvar FOR condição	contar registros
CREATE arquivo	definir novo arquivo de dados

DELETE FILE arquivo	deletar arquivo de dados
DELETE FILE qualquer arquivo.tipo	deletar qualquer arquivo.
DISPLAY (registro em curso) DISPLAY campo(s) (em registro em curso) DISPLAY RECORD # DISPLAY RECORD # campo(s) DISPLAY ALL DISPLAY ALL campo(s) DISPLAY FOR condição DISPLAY campo(s) FOR condição DISPLAY NEXT número DISPLAY NEXT número campo(s) DISPLAY NEXT número FOR condição DISPLAY NEXT número campo(s) FOR condição	exibir registro simples ou registros selecionados; pode-se anexar OFF a quaisquer destes formatos para desligar o nº do registro
DISPLAY FILES DISPLAY FILES ON drive	exibir os nomes dos arquivos de dados (arquivos .DBF)
DISPLAY FILES LIKE qualquer arquivo-tipo DISPLAY FILES ON drive LIKE qualquer arquivo-tipo	exibir os nomes dos arquivos selecionados
DISPLAY MEMORY	exibir todas as variáveis de memória
DISPLAY STATUS	exibir situação de funcionamento do dBASE
DISPLAY STRUCTURE	exibir estrutura do registro
DO arquivo CMD	executar um arquivo de comando CP/M
DO arquivo PRG	executar um arquivo de comando MS-DOS
EDIT # EDIT #, campo, dados	exibir registro para edição modo não full-screen: editar campos de forma simples, sem exibição
EDIT	entrar com edição de campo, de forma simples
ERASE	limpar a tela
HELP	exibir mensagens de auxílio
INSERT INSERT BEFORE	criar e inserir novo registro introduzir o modo de entrada de dados

INSERT BLANK INSERT BLANK BEFORE	criar e inserir novo registro saltar o modo de entrada de dados
JOIN TO arquivo FOR condição JOIN TO arquivo FOR condição FIELD campo(s)	reunir dois arquivos, formando um terceiro
LIST	o mesmo que DISPLAY, exceto que todos os registros são exibidos *default*
LOCATE FOR condição LOCATE NEXT número FOR CONDIÇÃO	busca seqüencial em exibição
MODIFY COMMAND arquivo MODIFY COMMAND arquivo-tipo	entrar com o editor de texto do dBASE e criar ou editar arquivo
MODIFY STRUCTURE	entrar com o modo de modificação da estrutura do registro
QUIT QUIT TO "programa" QUIT TO "programa", "programa"	encerrar o dBASE encerrar o dBASE e executar programas (CP/M-80)
RENAME arquivo TO arquivo	atribuir outro nome ao arquivo de dados
RENAME qualquer arquivo.tipo TO qualquer arquivo.tipo	atribuir outro nome a qualquer arquivo
REPLACE ALL campo WITH valor REPLACE campo WITH valor FOR condição REPLACE NEXT número campo WITH valor REPLACE NEXT número campo WITH valor FOR condição	alterar dados em múltiplos registros múltiplos campos WITH (com) valores separados por vírgulas: c W v, c W v
RESET	informar ao sistema sobre a troca de disco (CP/M)
SELECT PRIMARY SELECT SECONDARY	selecionar arquivo primário selecionar arquivo secundário
SET ALTERNATE TO arquivo TXT	definir nome de arquivo substituto de saída (.TXT)
SET ALTERNATE ON [OFF]	copiar saída em tela para arquivo substituto

SET CARRY ON [OFF]	transportar dados para o resumo seguinte no modo de entrada de dados
SET CONFIRM ON [OFF]	solicitar que RETURN confirme a entrada no modo de entrada de dados
SET DATE TO data	entrar com data do sistema
SET DEFAULT TO drive	definir drive *default* do disco para o dBASE
SET EXACT ON [OFF]	requerer verificação exata no campo completo
SET LINKAGE ON [OFF]	conectar dois arquivos para exibição ou relatório
SET PRINT ON [OFF]	imprimir o que está exibido na tela
SET SCREEN [ON] OFF	ligar o modo full-screen
SORT ON campo-chave TO arquivo SORT ON campo-chave TO arquivo DESCENDING	criar novo arquivo em ordem seqüencial
SUM (até 5) campos SUM (até 5) campos FOR condição SUM (até 5) campos TO memvar(s) SUM (até 5) campos TO memvar(s) FOR condição	somar os campos
TOTAL ON campo-chave TO arquivo TOTAL ON campo-chave TO arquivo FIELD campo(s) TOTAL ON campo-chave TO arquivo FOR condição TOTAL ON campo-chave TO arquivo FIELD campo(s) FOR condição	criar registro de resumo
UPDATE FROM arquivo ON campo-chave ADD campo(s) (RANDOM) UPDATE FROM arquivo ON campo-chave REPLACE campo(s) UPDATE FROM arquivo ON campo-chave REPLACE campo WITH campo	atualizar cadastro a partir do arquivo de transação
UPDATE FROM arquivo ON campo-chave ADD campo(s) REPLACE campo(s)	múltiplos campos WITH, campo separado por vírgulas c W c, c W c
USE arquivo	abrir arquivo
USE	fechar arquivo

Comandos que Deletam e Cancelam a Marca de Deleção de Registros

DELETE RECORD # DELETE ALL DELETE FOR condição DELETE NEXT número DELETE NEXT número FOR condição	marcar os registros a serem deletados
PACK	remover registros marcados de um arquivo
RECALL RECORD # RECALL ALL RECALL FOR condição RECALL NEXT número RECALL NEXT número FOR condição	cancelar a marca de deleção dos registros
SET DELETED ON [OFF]	tornar invisíveis os registros deletados

Comandos que Trabalham com Relatórios

EJECT	saltar para a próxima página na impressora
REPORT FORM arquivo REPORT FORM arquivo TO PRINT REPORT FORM arquivo TO PRINT FOR condição REPORT FORM arquivo NEXT número REPORT FORM arquivo NEXT número TO PRINT REPORT FORM arquivo NEXT número TO PRINT FOR condição	criar e executar relatório ou executar relatório
SET EJECT [ON] OFF	começar relatório em nova página
SET HEADING TO título	entrar com o título do relatório em curso
SET MARGIN TO número (1-254)	entrar com a margem esquerda para relatório

Comandos que Trabalham com Índices

FIND dados	encontrar registro através de índice, sem usar dados entre aspas
GO TOP	ir para o primeiro registro do índice
GO BOTTOM	ir para o último registro do índice
INDEX ON campo-chave To índice	criar um índice (tipo de arquivo .NDX)
REINDEX	atualizar índice
SET INDEX TO índice(s)	identificar até 7 índices para se trabalhar com arquivo de dados
USE arquivo INDEX índice	abrir arquivo de dados com índices

Comandos que Trabalham com Variáveis de Memória

COUNT TO memvar COUNT TO memvar FOR condição	contar registros e armazenar a contagem em variável de memória
RELEASE memvar(s) RELEASE ALL RELEASE ALL LIKE caracteres comuns em memvar RELEASE ALL EXCEPT caracteres comums em memvar	deletar variáveis de memória
RESTORE FROM MEM arquivo RESTORE FROM MEM arquivo ADDITIVE	recuperar variáveis de memória a partir de disco (tipo de arquivo .MEM)
STORE dado TO memvar STORE campo TO memvar	criar ou alterar variável de memória
SUM (até 5) campos TO memvar(s) SUM (até 5) campos TO memvar(s) FOR condição	somar campos e armazená-los em variáveis de memória

Comandos que Trabalham com Arquivos de Formato

@ linha, coluna SAY "mensagem de tela"	exibir mensagem
@ linha, coluna GET campo	exibir dados
@ linha, coluna SAY "mensagem de tela" GET campo	exibir mensagem e dados
SET FORMAT TO arquivo FMT	ativar formato de tela

APÊNDICE F

Índice dos Comandos e Funções do Nível 1

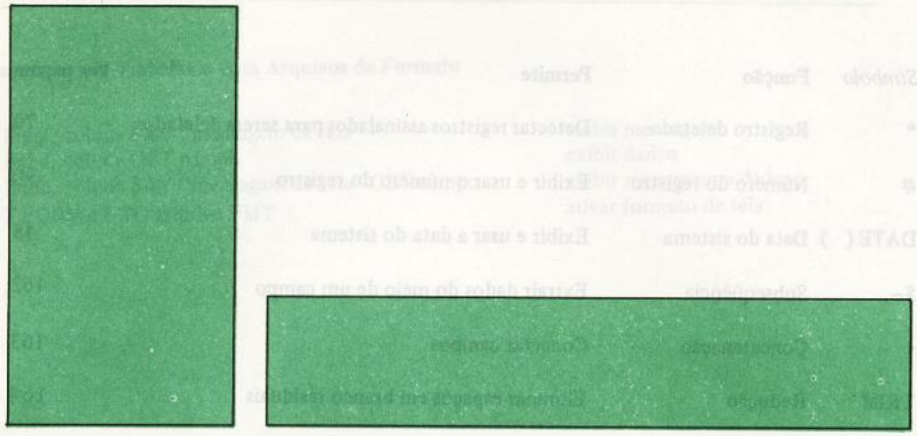
Comando	Permite	Ver página
?	Exibir dados ou resultados	151
APPEND	Introduzir o modo de entrada de dados	62
APPEND FROM	Anexar um arquivo a outro arquivo	67
BROWSE	Entrar com a combinação dos modos exibir/editar	79
CHANGE	Rever e editar registros selecionados	121
CLEAR	Fechar todos os arquivos e deletar todas as variáveis de memória	156
CONTINUE	Continuar a busca do próximo registro através do comando LOCATE	123
COPY	Criar cópias dos arquivos de dados	117
COPY STRUCTURE	Criar cópias das estruturas dos registros	118
COUNT	Contar registros	104
CREATE	Definir novo arquivo de dados	58
DELETE	Deletar registros ou arquivos	69

DISPLAY	Exibir dados, estruturas de registros, nomes de arquivos, variáveis de memória e funcionamento do dBASE	86
DISPLAY FOR	Exibir registros selecionados	86
DO	Executar um arquivo de comandos	178
EDIT	Editar um registro	73
EJECT	Saltar para a próxima página na impressora	112
ERASE	Limpar a tela	35
FIND	Encontrar um registro através de índice	142
GO TOP	Ir para o primeiro registro no índice	144
GO BOTTOM	Ir para o último registro no índice	144
HELP	Exibir mensagens de auxílio	47
INDEX	Criar um índice	85
INSERT	Criar novo registro no meio de um arquivo	123
JOIN	Reunir dois arquivos formando um terceiro arquivo	160
LIST	Semelhante a DISPLAY (todos os registros são exibidos à revelia)	97
LOCATE	Busca seqüencial sem exibição	123
MODIFY COMMAND	Entrar com o editor de texto do dBASE	126
MODIFY STRUCTURE	Alterar uma estrutura de um registro	134
PACK	Remover os registros marcados para deleção em um arquivo	69
QUIT	Encerrar o dBASE	61
RECALL	Remover a marca de deleção de registros	70

Nos comandos SET seguintes, os [ON] e [OFF] entre parênteses indicam os standards do dBASE.

Símbolo	**Função**	**Permite**	**Ver página**
*	Registro deletado	Detectar registros assinalados para serem deletados	70
#	Número do registro	Exibir e usar o número do registro	90
DATE ( )	Data do sistema	Exibir e usar a data do sistema	48
$	Subseqüência	Extrair dados do meio de um campo	162
+ –	Concatenação	Conectar campos	163
TRIM	Redução	Eliminar espaços em branco residuais	164
CHR	Caractere	Dar saída a caracteres especiais	167
!	Caixa-Alta	Converter caixa-baixa em caixa-alta	166
STR	Seqüência Numérica	Converter dados "N" em dados "C"	165

ÍNDICE ANALÍTICO



OUTROS LIVROS NA ÁREA

0-07-450167-4