1 Introdução

Dois grandes grupos de módulos compõem o *Open*BASE: módulos **básicos** e módulos **adicionais**.
Os **módulos básicos** OpenBASE são integrados por três sistemas:

1. Sistema de Definição de Banco de Dados (SDBD)
2. Rotinas de Manipulação de Banco de Dados (RMBD)
3. Sistema de Utilitários (SUBD)

Sistema de Definição de Banco de Dados

O Sistema de Definição de Banco de Dados (SDBD), denominado DEFINE, processa a descrição de um Banco de Dados (ESQUEMA) gerada através da sua Linguagem de Definição de Banco de Dados (LDBD).
Linguagem baseada no Modelo de Entidades e Relacionamentos (MER), disponibilizando a seus usuários um nível de detalhamento do Banco de Dados bastante amplo.
Através do DEFINE o Banco de Dados pode ser **criado** ou **modificado**.

Rotinas de Manipulação de Banco de Dados

As Rotinas de Manipulação de Banco de Dados (RMBD) compõem o núcleo do *Open*BASE. O *Open*BASE realiza suas funções através delas, porem, a nível interno, essas rotinas acessam outras rotinas responsáveis pelo método de acesso do *Open*BASE.
Todos os programas que constituem as formas possíveis de acessar um Banco de Dados, utilizam as RMDB. Para o usuário, a utilização da RMDB e totalmente transparente, a não ser que sejam utilizadas em programas do usuário, escritos em Linguagens Hospedeiras (COBOL, C, ..., FORTRAN).
Para a utilização das RMBD em programas escritos em Linguagens Hospedeiras, o *Open*BASE disponibiliza a biblioteca onde estão catalogadas as Rotinas (BDLIB), as quais estão disponíveis para uso irrestrito do usuário.

Sistema de Utilitários

O Sistema de Utilitários (SUBD) é um conjunto de programas voltados para a administração de Banco de Dados, os quais permitem:

- configurar o ambiente *Open*BASE.
- descarregar e recarregar um arquivo ou todo o Banco de Dados.
- adicionar dados de um arquivo em um arquivo do BD
- desfazer as transações não completadas.
- fornecer informações sobre a ocupação e utilização do BD
- verificar a consistência do Banco de Dados
- converter Bancos de Dados para o formato *Open*BASE.

Os **módulos adicionais** OpenBASE são integrados por três sistemas:

1. Sistema Interativo de Consulta e Atualização
2. Linguagem Estruturada de Consulta (SQL)
3. Linguagens de programação

Cada um desses componentes adicionais do OpenBASE possui seu próprio manual de documentação.

Sistema Interativo de Consulta e Atualização

O Sistema Interativo de Consulta e Atualização é um programa chamado **GERAL**, o qual oferece aos usuários uma interface interativa, bastante amigável, para consultas e atualizações em um Banco de Dados.
Possui linguagem não procedural, permitindo a utilização de telas de manipulação de dados e menus de opções. Como recurso adicional, dispõe de um poderoso gerador de relatórios.
Consulte nosso manual do software GERAL para obter maiores detalhes.

Linguagem Estruturada de Consulta

O Linguagem Estruturada de Consulta (SQL) baseia-se num ambiente chamado TSQL (Tecnocoop SQL), o qual implementa todos os recursos disponíveis da SQL ANSI sob o *Open*BASE. Consulte nosso manual do TSQL para obter maiores detalhes.

Linguagens de programação

As linguagens de programação da família OpenBASE se chamam **OPUS** e **OpusWin** , constituindo-se em poderosas ferramentas de programação de alto nível baseadas no dialeto Xbase com importantes extensões.
No que diz respeito aos comandos de manipulação de Banco de Dados, as linguagens **OPUS** e **OpusWin** oferecem ferramentas muito mais poderosas e de grande performance, pois estão plenamente integradas ao ambiente e estrutura do sistemas *Open*BASE.
Um programa escrito em OPUS ou **OpusWin**, quando submetido ao processo de compilação, é traduzido para a linguagem C, para então ser efetivamente compilado. Este processo oferece total portabilidade e alta performance nas

aplicações, além de permitir que o usuário possa agregar ao programa (no mesmo fonte e sem restrições), comandos da linguagem C.

2 Definição dos Bancos *Open*BASE

O processo da construção de um Banco de Dados OpenBASE é implementado através de duas etapas:

1. Elaboração do projeto lógico
2. Elaboração do projeto físico

A primeira etapa, ou seja, o Projeto Lógico (Modelo de Entidades e Relacionamentos do empreendimento), pode abranger, ainda, as fases de Normalização. A segunda etapa consiste na esquematização do Banco de Dados, ou seja, a elaboração do Projeto Físico, utilizando-se os recursos da Linguagem de Definição (LDBD) do *Open*BASE.
A definição de um Banco de Dados *Open*BASE consiste, basicamente, na elaboração de um esquema, que nada mais é do que a tradução do projeto lógico do Banco de Dados para a Linguagem de Definição de Banco de Dados (LDBD) do *Open*BASE.
O Esquema especifica para o sistema *Open*BASE todas as informações necessárias que farão parte do dicionário de dados, tais como:

- Nome e senha do Banco de Dados
- Estratégias de segurança , bloqueio e integridade
- Estratégias de replicação e recuperação dos arquivos do Banco de Dados
- Especificação dos arquivos de dados (Entidade, Relacionamento, arquivo externo etc …)
- Especificação dos atributos dos arquivos, ou seja seus itens campos
- Estruturas das chaves e regras de integridade referencial

2.1 Construção do Banco de Dados

A criação do Banco de Dados, baseia-se na compilação bem sucedida do Esquema.
Nesta etapa o usuário submete o Esquema ao programa DEFINE para compilá-lo. Se tudo correr bem na compilação, o Banco de Dados é criado: são criados o dicionário de dados, os Arquivos de Dados e os Arquivos Índices que compõem o Banco de Dados.

2.2 Manutenção do Banco de Dados

Após concluída a criação do Banco de Dados, o usuário está apto a selecionar, incluir, alterar e suprimir dados no BD. Por Exemplo

- Utilizando um dos utilitários OpenBASE para adicionar registros em um arquivo do BD, a partir de um arquivo do usuário.
- Executando um programa do usuário que tenha sido desenvolvido em linguagens como C/C++, COBOL, FORTRAN, ... que utilizam as rotinas OpenBASE.
- Usando o ambiente interativo do software GERAL.
- Executando um programa do usuário desenvolvido em OPUS / **OpusWin**.

A manutenção de um Banco de Dados OpenBASE baseia-se na necessidade de modificar, reorganizar e recuperar Banco de Dados. Estas tarefas são executadas através dos Utilitários de Administração do Banco de Dados (SUBD).

2.3 O Esquema de um Banco OpenBASE

Através do esquema do Banco de Dados são definidos os aspectos físicos do modelo projetado, isto é, as características operacionais do Banco de dados, seus arquivos, registros, chaves, itens de dados e seus Relacionamentos. O arquivo texto que constitui o esquema deve ser escrito na Linguagem de Definição de Dados
O programa DEFINE compila o esquema do Banco de Dados, sendo que, terminada com sucesso a compilação do esquema, será criado o dicionário assim como todos os arquivos de dados e de índices que compõem o Banco de Dados.

2.3.1 Sintaxe geral do esquema

A seguir apresentamos a sintaxe geral de um esquema de Bancos de Dados OpenBASE que será comentada a seguir, passo a passo.

[<< <*comentário*> >>]
[$CONTROLE <*opção*>, ... , <*opção*>]
BANCO [<*percurso*>] <*nome_bd*> <*código_de_seguranca*>...
[{ARQRECUP | VARIAVEL | ARQREFAZ | DIARIO | DIAREC}]...
[{BLOQARQ | BLOQPAG | BLOQREG}]...

[ESQUEMA=[<*percurso_bd_origem*>]<*nome_bd_origem*>...
<*código_de_seguranca_bd_origem*>...
[<*palavra_de_nível_bd_origem*>]]
[NIVEIS: <*numero_nivel*> <*palavra_nivel*>
<*numero_nivel*> <*palavra_nivel*>]
[RELACOES:]
NOME:~[<*percurso*>] <*nome_arquivo*> <*tipo_arquivo*>...
[ESQUEMA = [<*percurso_bd_origem*>] <*nome_bd_origem*>...
<*codigo_de_seguranca_bd_origem*>...
[<*palavra_de_nivel_bd_origem*>]]
[REGISTRO:]
<*nome_item*> [{[<*rep*>] / (<*ligações*>) / (<*caminho*>)/(0)}]...
<*tipo*>:<*tamanho*>[,<*num_decimais*>] ... | |
[(<*num_nivel_ler*>,<*num_nivel_grav*>)] ...
[{POS <*nome_item_rd*>[+ <*deslocamento*>] /...
VIRTUAL (<*nome_item_part*>, ..., <*nome_item_part*>)}] ...
[UNICA]
...
...

Dentro do esquema, podem ser incluídos comentários, podendo estar em qualquer lugar e ocupar várias linhas.

Sintaxe **[<< <comentário> >>]**

Onde: << Marca o inicio de um comentário e >> Marca o fim de um comentário

Exemplo

<< Isto e um comentário: primeira linha de um comentário de várias linhas >>
<< Isto e um comentário: segunda linha de um comentário de várias linhas >>

2.3.2 Declaração das Opções de Controle

O comando **$CONTROLE** do Esquema é opcional, podendo ser usado pelo projetista do Banco de Dados para especificar opções de controle válidas para todo o processamento do Esquema fonte pelo programa DEFINE. O comando $CONTROLE deve ser especificado no início do fonte do Esquema (primeiro registro do Esquema).

Sintaxe $CONTROLE <opção>, ..., <opção>

onde <opção> é uma das opções de controle relacionadas a seguir.

A lista de opções de controle é bastante extensa. Para não complicar demasiado, somente algumas opções de controles serão abordadas no momento. As demais opções serão abordadas detalhadamente mais adiante.

ARQUIVO (default)

- Cria os arquivos de dados caso eles não existam.

ARQUIVOS = <n>

- Informa o numero de arquivos que serão definidos na Base de Dados (arquivos de dados e arquivos de índices).
- O valor "default" de n é de 1000 arquivos

Observação:

As opções de controle podem ser definidas com validade para todos os componentes de um Banco de Dados, sendo que para isso basta declará-las imediatamente após o comando $CONTROLE. As opções de controle podem ser definidas de modo a serem válidas apenas para determinados arquivos, sendo que para isto devem ser declaradas apenas para os arquivos onde se deseja que a opção atue. Vale lembrar que só faz sentido usar nos arquivos para desligar a opção ligada no comando CONTROLE ou para documentação.

2.3.3 Sintaxe

NOME: <nom_arq> <tipo> [<esquema>] [LISTA | NÃOLISTA]

CRIARQUIVO

- Os arquivos de dados são criados. Se já existirem, esses arquivos serão reinicializados.

DICIONÁRIO (default)

- Determina a criação do dicionário de dados após a compilação do esquema.

ERROS = <n>

- Se esta opção for especificada, após alcançado o número de erros estipulado em <n>, será interrompida a compilação do Esquema pelo DEFINE.
- O valor default de n e 99999

ITENS = <n>

- Informa o número de itens de dados que serão definidos no Banco de Dados.
- O valor "default" de n e de 1800

LIGACOES = <n>

- Informa o número de ligações (Relacionamentos declarados) que pode conter o Banco de Dados.

▪ O valor "default" de n e de 500 ligações

LIGAT

▪ Permite que um arquivo tipo E (Entidade) ou R (Relacionamento) referencie um arquivo tipo T (Tabela) de outro Banco de Dados.

LISTA (default)

▪ Libera a listagem do Esquema na unidade de saída especificada.

NÃOCOMPCAB

▪ Indica a não compressão de cabeçalho dos arquivos de índice.

NÃOLISTA

▪ Suprime a liberação da listagem do Esquema.

NOMEBANCO

▪ Liga o dispositivo de segurança do *Open*BASE, encarregado de evitar colisões de nomes idênticos de arquivos de Bancos de Dados diferentes, evitando assim que haja superposição de arquivos e a destruição dos mesmos.

NOMEIGUAL

▪ Indica que num Banco de Dados podem existir itens com nomes iguais, desde que estejam em arquivos diferentes.

PRIMBIN

Deve-se utilizar esta opção de controle nos seguintes casos:

- o primeiro item ter tipo I, B, D ou M (binário).
- o arquivo não ter chaves .
- ter a primeira chave única mais a opção de controle
- COMPCAB (DEFAULT)

Caso não seja utilizada esta opção, possibilitando que o banco de dados seja gerado, a seguinte mensagem será enviada durante a definição do banco de dados:

"Primeiro item não pode ser binário"

Outra forma de solucionar este problema é a declaração da cláusula NAOCOMPCAB para o arquivo.

REDEFU

▪ Permite que um item de tipo Universal possa ser composto de itens de tipo interno binário.

SEMARQUIVO

▪ Suprime a criação de arquivos de dados.

SEMDICION

▪ Suprime a criação do dicionário de dados após a compilação.

SEMINTEGR

▪ A mensagem "número de ligações maior que esquema" não é mais emitida para arquivos tipo T com números de ligações maior que na entidade no esquema.

```
<<banco "a" definido em a.e>>
        BANCO a 1 ARQRECUP
        NOME: ae    e
                a (0)     u03
                b         u05
<<banco "b" definido em b.e>>
        $CONTROLE LIGAT, SEMINTEGR
        BANCO b 1 arqrecup
        NOME: be
                bc (0)    u05
                ba (ae)  u03
```

Observação:

Esta opção de controle (**SEMINTEGR)** só deve ser utilizada quando o usuário tiver certeza de que é a solução do seu problema pois, esta opção utilizada de forma indevida, poderá acarretar sérios problemas de INTEGRIDADE REFERENCIAL no BANCO DE DADOS.

Por exemplo, um registro do arquivo "ae" pode ser excluído no banco "a" mesmo havendo registros no arquivo "be" do banco "b" ligados a "ae", o que viola as regras de integridade referencial no banco "b".

SEMTABELA

Suprime a impressão da tabela sumário.

SEQIDX = 1

Indica que os nomes dos arquivos índices serão gravados no diretório do disco com um número seqüencial após ao nome.

Se diferente de 1 (default), grava um número seqüencial na frente do nome do arquivo de índice.

TABELA (default)

Quando a compilação é completada com êxito, o DEFINE imprime uma tabela sumário a respeito dos arquivos de dados contendo:

- Nome dos Arquivos

- Tipos dos Arquivos
- Quantidade de itens por arquivo
- tamanho dos registros
- tamanho dos cabeçalhos
- numero de ligações
- quantidade de chaves por arquivo.

Resumindo, o DEFINE opera com as seguintes opções de controle por "default":

- ARQUIVO
- ARQUIVOS=1000
- DICIONÁRIO
- ERROS=99999
- 2000 (dois mil/itens)
- LIGACOES=500
- LISTA
- TABELA

2.3.4 Exemplo

$CONTROLE ARQUIVOS=200, TABELA, ERROS=10, ITENS=500

Nesse caso, as opções de controle geradas são:

- ARQUIVO
- ARQUIVOS=200
- DICIONÁRIO
- ERROS=10
- ITENS=500
- LIGACOES=500
- LISTA
- TABELA

Notar que a opção TABELA poderia ser omitida por ser tratar de uma opção "default".

2.4 Declaração do Banco de Dados

Constitui-se num conjunto de declarações que devem ser especificadas no Esquema de forma ordenada. Estas declarações estão apresentadas nos parágrafos que seguem.

2.4.1 Sintaxe

BANCO [<percurso>] <nome_bd> <codigo_de_seguranca> [BITS64] ...
[{ARQRECUP | VARIAVEL | ARQREFAZ | DIARIO | DIAREC | AUTOREC}] ...
[{BLOQARQ | BLOQPAG | BLOQREG}] ...
[ESQUEMA = [<percurso_bd_origem>]<nome_bd_origem> ...
<codigo_de_seguranca_bd_origem> ...
[<palavra_de_nivel_bd_origem>]]
[DISTRIB = [<servidor>]

2.4.2 Descrição

O comando **BANCO** marca o início da declaração de um dicionário de dados OpenBASE.
<percurso> indica o diretório completo onde será criado o dicionário de dados (Default: /usr/tsgbd/tsdic)
No ambiente **Windows** o nome completo do percurso inclui, obviamente, a letra do drive, que deve preceder o nome do diretório (ou pasta) onde reside (ou vai residir) o banco de dados OpenBASE. Caso não seja especificada a letra do Drive, serão aplicadas as seguintes regras:

- Na definição de um Banco de Dados, não havendo especificação da letra do drive, nos comandos BANCO e/ou NOME:, será assumida a letra do "current drive".
- Na utilização de um Banco de Dados, ou seja na abertura do Banco, é aceita a letra do Drive conforme especificado nas informações de Configuração do Registry (antigo arquivo FACPRINT).
- Pode ser utilizado o utilitário **Wincnfg.exe** (versão Windows do bdcnfg) para consultar ou modificar o percurso dos Bancos de Dados OpenBASE.
- Caso não haja especificação da letra do Drive dentro das informações do Registry, é assumido o Drive C como Drive default.

Se um banco for mudado de drive deve ser executado o define para colocar o novo drive no dicionário.
O operando **<nome_bd>** especifica o nome do dicionário de dados: consiste de uma cadeia com até 12 caracteres, com as seguintes restrições:

- pode conter letras, números, ponto (.) e sublinhado (_).
- tem que iniciar por letra.
- não pode ter brancos.

- deve ser único no diretório.

No MS-DOS <nome_arquivo> não pode ultrapassar 8 caracteres.
O operando **<codigo_de_seguranca>** especifica o código de segurança do Banco de Dados que consiste de um número inteiro entre 1 e 4.294.836.225 (inclusive). Este número é o escolhido pelo projetista do Banco de Dados como uma senha.

BITS64

Se houver a possibilidade de um ou mais arquivos de um Banco de Dados exceder o tamanho de 4GB, a opção <BITS64> indica que, nos índices, os endereços dos registros terão 64 BITS, ou seja, 8 Bytes.
Para que esta opção funcione corretamente, é necessário que o sistema operacional, o FileSystem e as bibliotecas C/C++ suportem os sistemas LFS (Large File System).
No ambiente Windows, esta opção está disponível apenas para arquivos NTFS, pois o FAT32 não suporta LFS.

ARQRECUP

Determina que seja criado um ou mais arquivos, destinados a armazenar os dados das transações antes deles serem modificados.
Permitindo assim que uma ou mais transações não completadas possam ser desfeitas a qualquer momento (rollback / undo).

VARIAVEL

A opção <VARIAVEL> indica que o arquivo de recuperação terá registros de tamanho variável, diminuindo, em princípio, o seu tamanho.

ARQREFAZ

A opção <ARQREFAZ> indica que será criado um arquivo auxiliar com o nome ***<nome>*.Z** que conterá as atualizações realizadas no Banco de Dados de forma a permitir o reinicio automático de uma transação. O arquivo ***<nome>*.Z** é associado a cada processo e contém uma lógica (apenas as informações modificadas) das atualizações realizadas.

DIÁRIO

Determina que seja utilizado o arquivo DIÁRIO (criado pelo programa BDSGBD) para armazenar todos os dados das transações depois deles serem modificados. Permitindo assim que uma ou mais transações completadas possam ser refeitas, a qualquer momento, a partir de uma determinada data/hora ou a partir de uma determinada transação identificada pelo seu numero.
Esta opção implica na utilização automática da opção ARQRECUP.

DIAREC

Determina que seja utilizado o arquivo DIÁRIO (criado pelo programa BDSGBD) para armazenar todos os dados das transações antes deles serem modificados e depois deles serem modificados.
Permitindo assim que uma ou mais transações completadas possam ser desfeitas ou refeitas, a qualquer momento, a partir de uma determinada data/hora ou a partir de uma determinada transação identificada pelo seu número.
Esta opção implica na utilização automática da opção ARQRECUP.

AUTOREC

A opção AUTOREC indica que ao ser aberto um banco, se o arquivo de recuperação contiver uma transação não completada, a recuperação será feita automaticamente sem solicitar a execução do utilitário bdrecu, desde que não exista nenhum outro processo ativo utilizando o banco e que o bloqueio seja do tipo banco.

BLOQARQ

Determina, para todos os processos de atualização do Banco de Dados que a estratégia de bloqueio será baseada em bloqueio de Arquivos.

BLOQPAG

Estabelece para todos os processos de atualização do Banco de Dados que a estratégia de bloqueio será baseada em bloqueio de páginas.

BLOQREG

Determina para todos os processos de atualização do Banco de Dados que a estratégia de bloqueio será baseada em bloqueio de registros.

ESQUEMA =

Indica que dicionário de dados referenciado em <nome.bd> conterá a estrutura de um sub-esquema, ou seja, a estrutura parcial de um Banco de Dados já existente.
Esta cláusula determina que, na compilação do sub-esquema, seja aberto o Banco de Dados que dará origem ao sub-esquema (BDO), com a finalidade de consistir as estruturas definidas em ambos.
Assim sendo:

- Todos os arquivos declarados no sub-esquema devem existir no BDO
- Todos os itens declarados em um arquivo no sub-esquema devem existir no arquivo que esta sendo referenciado no BDO e em conseqüência os tipos e tamanhos dos itens devem ser idênticos.

A definição de um arquivo no sub-esquema deve contemplar todos os itens chaves do arquivo que esta sendo referenciado no BDO, na ordem em que foram declarados no BDO.
<percurso_bd_origem> Percurso onde se encontra o dicionário de dados do Banco de Dados que será referenciado no sub-esquema.
<nome_bd_origem> Nome do dicionário de dados do Banco de Dados que será referenciado no sub-esquema.

<codigo_de_seguranca_bd_origem> Código de segurança do Banco de Dados que será referenciado no sub-esquema.
<palavra_de_nivel_bd_origem> Palavra de nível do Banco de Dados que será referenciado no sub-esquema.
Em casos de sub-esquema não é necessário declarar as cláusulas de controle ARQRECUP, DIÁRIO, BLOQARQ | BLOQPAG | BLOQREG etc … pois esses controles estão subordinados ao Banco de Dados de origem ao qual o sub-esquema se referencia.

2.4.3 Exemplos

BANCO nota_fiscl 33 AQRRECUP DIÁRIO
BANCO /usr/apl/bds/folha_pg 33 ESQUEMA=bd_rh 21 gerent
banco teste 1

Em caso de Bancos Distribuídos:

<banco> <seg> ... [distrib = <servidor>]

A opção **distrib** indica que o banco de dados é distribuído e o servidor mestre.

nome: <arquivo> <tipo> ... [servidor = <servidor>]

O operando **<servidor>** especifica o nome do Host servidor onde reside o Banco distribuído, por Exemplo.

banco abc 1 distrib = TS1
nome: arq1 e servidor = TS2
c1(1) u03
nome: arq2 r servidor = TS3
c2(arq1) u03

2.5 Declaração dos Níveis de Acesso

Em um Banco de Dados, são estabelecidos níveis de acesso a informações para os diversos tipos de usuários. Como por exemplo, estes níveis de acesso podem ser proporcionais aos níveis da hierarquia funcional dos usuários que manipulam o Banco de Dados, em uma organização qualquer.

2.5.1 Sintaxe

NÍVEIS: <numero_nivel> <palavra_nivel>
<numero_nivel> <palavra_nivel>]

2.5.2 Descrição

Define os Níveis de Privacidade. O *Open*BASE possibilita a atribuição de níveis de leitura e gravação, para cada item de dados.
Cada nível de privacidade e definido associando-se um numero de nível com sua respectiva palavra de nível.
<numero_nivel> <palavra_nivel>
Os números de níveis são representados por inteiros de 1 a 15 e que devem ser especificados em ordem crescente na elaboração do Esquema do Banco de Dados. As palavras de níveis são constituídas por ate 6 caracteres, sendo o primeiro caracter obrigatoriamente uma letra.
A declaração dos níveis é opcional. Caso seja declarado algum nível, e necessário declarar também o nível máximo, ou seja, associar uma palavra de nível ao nível 15.

2.5.3 Exemplo

NIVEL:01 todos
05 superv
10 gerent
12 diret
15 dba

2.6 Declaração dos Arquivos de Dados

2.6.1 Sintaxe

[RELACOES:]
[ESQUEMA = [<percurso_bd_origem>]<nome_bd_origem>...
<codigo_de_seguranca_bd_origem>...
[<palavra_de_nivel_bd_origem>]]...

2.6.2 Descrição

Declaração dos arquivos de Dados.
Opcionalmente, para documentar o início das declarações dos arquivos de dados, utiliza-se a declaração da palavra RELAÇÕES seguida de dois pontos (:).
NOME: Marca o início da declaração de cada arquivo de dados.
<percurso> Especifica o percurso onde será criado o arquivo de dados. Em caso de omissão, será assumido o mesmo percurso utilizado para o dicionário de dados.

<nome_arquivo> Especifica o nome do arquivo de dados e consiste de uma cadeia com a té 10 caracteres com as seguintes restrições:

- pode conter letras, números, ponto (.) e sublinhado (_)
- tem que iniciar por letra
- não pode ter brancos
- deve ser único no diretório

No MS-DOS, <nome_arquivo> não pode ultrapassar 8 caracteres.

<tipo_arquivo>

Especifica o tipo do arquivo de dados. E designado por uma letra.

Tipos Básicos do MER: E e R

E —> Entidade

R —> Relacionamento

Referências Externas aos tipos básicos: T e C

T —> Tabela

C —> Consulta

Tipos documentacionais: A, L e F

A —> Entidade Associativa

L —> Entidade de Ligação

F —> Entidade Fraca

Arquivos do tipo Tabela

Arquivos Tabela se referem a arquivos do tipo Entidade pertencentes a outros Banco de Dados e arquivos do tipo Consulta, se referem a arquivos Relacionamentos pertencentes a outros Banco de Dados.

Arquivos tipo T ou C

Arquivos tipo T ou C também podem participar de sub-esquemas. A definição de um arquivo tipo T ou C deve contemplar todos os itens chaves do arquivo que esta sendo referenciado (E ou R).
Os tipos dos arquivos documentacionais são convertidos pelo DEFINE para os tipos básicos. Assim sendo, Entidade Associativa ou Entidade de Ligação e convertida para Entidade, assim como, Entidade Fraca e convertida para Relacionamento.
ESQUEMA =
Esta cláusula só será utilizada no caso de interseção de Banco de Dados, ou seja, se o tipo do arquivo for T (tabela) ou C (consulta).
Em função de um arquivo tipo T ou C se referir a um arquivo de um outro Banco de Dados, esta clausula determina que na compilação do Esquema ou sub-esquema, seja aberto o Banco de Dados, ao qual o arquivo tipo T ou C se refere (BDR), com a finalidade de consistir as estruturas de ambos os arquivos: o que se refere com o que está sendo referenciado.
Assim sendo, todos os itens declarados no arquivo tipo T ou C devem existir no arquivo tipo E ou R que está sendo referenciado no BDR e em conseqüência os tipos e tamanhos dos itens devem ser idênticos.
<percurso_bd_origem> Percurso onde se encontra o dicionário de dados do Banco de Dados ao qual o arquivo pertence efetivamente.
<nome_bd_origem> Nome do dicionário de dados do Banco de Dados ao qual o arquivo pertence efetivamente.
<codigo_de_seguranca_bd_origem> Código de segurança do Banco de Dados ao qual o arquivo pertence efetivamente.
<palavra_de_nivel_bd_origem> Palavra de nível do Banco de Dados ao qual o arquivo pertence efetivamente.

2.6.3 Exemplo

<< BANCO exemplo1 33 ** Banco hipotético ** >>
NOME: arq_dpto E << Entidade Departamentos >>
NOME: arq_func A << Entidade Funcionários>>
<< BANCO exemplo2 44 ** Banco hipotético ** >>
NOME: arq_func T ESQUEMA=exemplo1 33
NOME: arq_proj E << Entidade Projetos >>
NOME: arq_part R << Relacionamento Participação >>

2.7 Declaração dos Itens de Dados

A declaração dos itens de dados e feita logo a seguir a declaração do arquivo de dados cujos itens de dados fazem parte do seu registro.

2.7.1 Sintaxe

[REGISTRO:]
<nome_item> [{[<rep>]/(<ligações>)/(<caminho>)/(0)}]

... <tipo>:<tamanho>[,<num_decimais>]
... [(<num_nivel_ler>,<num_nivel_grav>)]
... [{POS <nome_item_rd>[+<deslocamento>] /
... VIRTUAL (<nome_item_part>, ..., <nome_item_part>)}]
... [UNICA]

2.7.2 Descrição

Na declaração dos itens de dados, opcionalmente, para documentar o inicio das declarações dos itens de dados que compõem o registro de um arquivo de dados, utiliza-se a declaração da palavra REGISTRO seguida de dois pontos (:).
<nome_item> É o nome de um dos itens que compõem o registro do arquivo de dados o qual compõe-se de uma cadeia com até 12 caracteres com as seguintes restrições:

- pode conter letras, números, ponto (.) e sublinhado (_)
- tem que iniciar por letra
- não pode ter brancos
- deve ser único no diretório.

[<rep>] É o fator de repetição de um item. A LDBD não permite que seja implementado itens multivalorados. Para tanto, o usuário pode definir que um item seja repetido tantas vezes quanto for determinado em <rep>. **<rep>** consiste de um valor numérico inteiro contido entre colchetes.
Este recurso quando utilizado faz com que o DEFINE associe aos nomes do itens repetitivos, um número seqüencial que varia de 1 a <rep>.
Assim sendo, se o nome do item for telefone e o fator de repetição for 3, três itens serão criados com os respectivos nomes: telefone1, telefone2 e telefone3.
Observações:

- Um item repetitivo não pode ser chave e nem ser redefinido.
- Tratar itens repetitivos como sendo itens comuns, ou seja, não confundir um item repetitivo com vetor.

(<ligações>)
Especifica o número de ligações que a Entidade mantém com outras Entidades, ou seja, e o número de vezes em que a chave primária da Entidade (Atributo Determinante) e referenciada como chave estrangeira em outros arquivos do Banco de Dados. Assim sendo, <ligações> e um número que pode variar de 0 a n.
Este argumento deve ser declarado somente em arquivos tipo Entidade e ao lado do item que corresponde ao Atributo determinante da Entidade.
Desta forma, o item que associa o número de ligações passa a ser a chave primária do arquivo Entidade.
Observações:

- O número de ligações tem que ser declarado entre parênteses, pois é esta notação sintática "()" que especifica para o DEFINE a declaração de um item chave porém esta notação "()", não determina a categoria da chave (primária ou secundária).
- Para que o DEFINE possa reconhecer que a chave declarada corresponde a chave primaria do Arquivo tipo Entidade, a mesma tem que ser a primeira chave declarada no registro. Caso contrário a chave não será reconhecida como primária e sim como secundária.
- Se o número de ligações não corresponder exatamente com o número de vezes que o Atributo Determinante é referenciado como chave estrangeira, o DEFINE acusa erro na compilação do Esquema. Assim sendo, o número de ligações tem que corresponder a quantidade de caminhos que fazem referencia a Entidade.

Numa relação entidade (tipo E ou T) o número de ligações pode ser especificado como *, o que será automaticamente substituído pelo número de referências nos arquivos posteriormente definidos, por Exemplo

nome: PESSOA e
NOMEP (*) u30 << (*) será substituído por (1) >>
IDADE n2
nome: FILHOS r
NOMEP1 (PESSOA) U30
IDADEF n2

<caminho>)
É o nome do arquivo Entidade com o qual o item de dados ira fazer associação (procedência). Indica que o item e uma chave estrangeira (Atributo Associativo / Item Associativo).
Este argumento define de forma declarada o Relacionamento entre as Entidades e assim sendo, determina para o *Open*BASE a obrigatoriedade do controle da Integridade Referencial.

Observações:

- Pode ser declarado em arquivo tipo Entidade porém não pode ser o primeiro item do arquivo Entidade.
- Um arquivo Entidade ou Relacionamento pode ter em seu registro mais de uma chave estrangeira. O limite de chaves por registro e de até 127.
- Chave estrangeira está na categoria de chave secundária.

(0)
Indica que o item é uma chave alternativa, porém e for a 1a chave de um arquivo tipo Entidade, define que a Entidade não possui ligações mas assume o papel de chave primária. Caso contrário corresponde a uma chave alternativa que pode ser declarada tanto em arquivos tipo Entidade quanto em arquivos tipo relacionamento.
Observações:

- Todos os Itens podem ser chaves, salvo os Itens repetitivos.
- Um arquivo tipo Entidade ou Relacionamento pode ter em seu registro mais de uma chave alternativa.
- limite de chaves por registro é de até 127.
- Chave alternativa está na categoria de chave secundária.

<tipo>:<tamanho>[,<num_decimais>
Define o tipo e o tamanho do Item, assim como o número de casas decimais no caso do item suportar valores numéricos.

2.7.3 Tipo do Item

O tipo do item a ser declarado em <tipo>, consiste de uma letra, no qual pode ser:

U	**Universal**
N	**Numérico sem sinal**
S	**Numérico com sinal**
P	**Numérico compactado sem sinal**
C	**Numérico compactado com sinal**
I	**Numérico Binário (positivo)**
B	**Numérico Binário (positivo/negativo)**
F	**Numérico Ponto Flutuante (positivo ou negativo)**
D,D2,D4	**Data**
L	**Lógico**
M	**Multimídia (Objeto Binário Longo)**

No OpenBASE um item de dado do **tipo D2**, só aceita datas compreendidas entre 01/01/1901 e 04/06/2080, inclusive, na conversão de uma variável de memória para o item D2 do banco. Caso a data esteja fora deste intervalo, é emitida a mensagem "OPUS(varite) => Estouro na conversão numérica". Para datas com tipo D4 esta conversão pode ser feita para datas dentro ou fora do intervalo mencionado anteriormente.
O item D4 indica uma data no formato aaaa/mm/dd onde aaaa ocupa 2 bytes, mm 1 byte e dd 1 byte. O dia 1 é considerado 01/01/0001. A data 31/12/1900 ainda é considerada como data nula para se manter compatibilidade com o tipo D2.
Para sua utilização na OPUS em conjunto com as datas tipo D2 foram feitas as seguintes alterações:
DTOC
Para itens do banco do tipo D2 retorna uma cadeia de caracteres no formato dd/mm/aa se set century off e 1901 <= ano <= 1999. Caso contrário retorna uma cadeia no formato dd/mm/aaaa.
Para itens do banco do tipo D4, retorna uma cadeia no formato dd/mm/aaaa, mesmo para datas menores do que 1901 e datas maiores do que 1999.
CTOD
Considera uma cadeia no formato dd/mm/aa como dd/mm/19aa.
Conversão de variável para item do banco.
Se tipo D2, 01/01/1901 <= data <= 04/06/2080, caso contrário é emitida a mensagem OPUS (varite) => Estouro na conversão numérica.
Conversão do tipo do item D2 para D4.
Os itens tipo D2 de um banco de dados podem ser descarregados e carregados no mesmo banco de dados com os itens alterados de tipo D2 para tipo D4.

2.7.4 Tamanho do Item

O tamanho do item a ser declarado em **<tamanho>**, consiste de um valor numérico que varia em função do tipo do item. Este tamanho nem sempre representa o número de bytes para armazenar o conteúdo do item.
Para melhor compreensão, consulte a Tabela a seguir.

Tabela de Tipos e Tamanhos				
Tipo item	**Tamanho Declarar**	**Bytes (tb)**	**Valores Suportados**	**Tipo Interno**
U	1 a 2048	1 a 2048	Caracter (qualquer)	Cadeia
N	1 a 18	1 a 18	Numérico de 1 a 18 dígitos	Cadeia
S	1 a 18	1 a 18	Numérico de 1 a 18 dígitos mais sinal	Cadeia
P	2 a 18 (par)	1 a 9	Numérico de 1 a 18 dígitos	Compactado
C	1 a 17 (impar)	1 a 9	Numérico de 1 a 18 dígitos mais sinal	Compactado
I	2 ou 4	2 ou 4	Numérico entre 0 e 2	Binário

F	4 a 8	4 a 8	Numérico de n dígitos. Se tb=4 então n=7 senão n=15.	Ponto flutuante
B	1 ou 7	1 ou 7	Numérico entre –2(8*tb-1) e 2(8*tb-1)-1.	Binário
D	2	2	Data no formato dd/mm/aa entre 00/00/00 e 31/12/99.	Binário
D2			Data no formato dd/mm/aaaa entre 1/01/1901 e 04/06/2080.	Binário
D4				Binário
L	1	1	0 -> falso 1 -> verdadeiro	Binário
M	4 (**)	1 a 4 Gb	String de Bits	Binário

(**) Itens tipo Multimídia (M4) são implementados pelo *Open*BASE como arquivos em disco de tamanho variável. Caso o Arquivo de Dados possua este tipo de item, no diretório onde se encontra o arquivo de dados é criado um subdiretório que será composto por outros subdiretórios, tantos quantos a quantidade de itens Multimídia declarados no Arquivo de Dados.
A aplicação deste tipo de item destina-se a armazenar textos longos, imagens ou sons.

2.7.5 Número de Casas Decimais

Se o item suportar valores numéricos é possuir casas decimais, o número de casas decimais e informado em <num_decimais>, que consiste de um valor numérico entre 1 a 18. Neste caso, o separador entre o tamanho do item e as casas decimais é obrigatoriamente a vírgula (,).
As casas decimais são virtuais, assim sendo, internamente, o valor contido em <tamanho>, corresponde ao conteúdo do item com suas respectivas casas decimais: internamente não apresenta separador entre a parte inteira e as casas decimais.

(<num_nivel_ler>,<num_nivel_grav>)
Especifica um par de números inteiros que declaram o nível de privacidade para leitura e gravação do item de dados.
<num_nivel_ler>
É o número do nível de leitura que consiste de um número de nível declarado anteriormente em nível, tem que ser menor ou igual ao nível de gravação.
<num_nivel_grav>
É o número do nível de gravação que consiste de um número de nível declarado anteriormente em nível, tem que ser maior ou igual ao nível de leitura.
Os níveis de leitura e gravação estão associados às suas respectivas palavras de níveis e funcionam da maneira seguinte: Na abertura do Banco de Dados, o usuário declara a sua palavra de nível e o *Open*BASE converte esta palavra para o número do nível associado à palavra (Numero de Nível do usuário - NNU). Desse modo, permite consistir se o usuário tem ou não autorização para ler ou gravar (atualizar) o item. Assim sendo, se NNU for maior ou igual ao nível de leitura do item, o usuário pode ler o item. Se NNU for maior ou igual ao nível de gravação, o usuário pode gravar o item.
Caso o usuário, num procedimento de leitura não seja autorizado a acessar o item, o valor de retorno do item é mascarado em função do seu tipo. Veja tabela abaixo.

Tipo dos itens	Máscaras
U	Brancos
N,S,P,C,I,B.F	Cadeia de dígitos preenchida com o número nove (999999999)
D,D2,D4	Data em branco (" / / ")
L	Falso (zero)

POS <nome_item_rd> [+ <deslocamento>]
Define que o item é uma redefinição de um outro item de dados, permitindo assim que um item seja decomposto em subitens ou que um item seja composto por itens contíguos.
A implementação da redefinição de um item baseia-se na superposição de itens na mesma área de registro. Assim sendo, um item que redefine um outro, não ocupa espaço adicional no registro, em função de utilizar a mesma área de registro ocupada por um ou mais itens já declarados.
<nome_item_rd> É o nome do item a ser redefinido que consiste do nome de um item já existente (não pode ser o nome de um item repetitivo).
<deslocamento> É o deslocamento em bytes, em relação a posição inicial do item a ser definido que consiste de um número inteiro.
Se <nome_item> iniciar na mesma posição de registro, ocupada por <nome_item_rd>, não é necessário informar o <deslocamento>.

VIRTUAL (<nome_item_part>, ..., <nome_item_part>)
Define que o item é um item virtual.
A implementação de itens virtuais, baseia-se na montagem em memória da composição de itens. Assim sendo, itens virtuais não ocupam espaço adicional no registro.
<nome_item_part>, ..., <nome_item_part>) É a lista dos nomes dos itens que compõem o item virtual.

Observações sobre itens virtuais:

- Itens virtuais são os últimos itens a serem declarados no arquivo do qual fazem parte.
- Os itens que participam da lista, podem ser declarados na lista em qualquer ordem.
- tamanho do item virtual, corresponde a soma dos tamanhos dos itens que o compõem.
- Se os itens da lista são do tipo N, o tipo do item virtual pode ser N ou U. Se os itens da lista são do tipo N e U, obrigatoriamente o tipo do item virtual tem que ser U. Se na lista de itens houver item do tipo interno binário, obrigatoriamente o item virtual tem que ser do tipo U e o controle REDEFU tem que ser declarado.
- Itens virtuais não podem ser atualizados.

ÚNICA

Estabelece restrição de unicidade para chave secundária (chave única).
Não pode ser declarado em chaves primárias devido ao fato deste tipo de chave ser única por definição.

2.7.6 Exemplo

NOME: arq_dpto E << Entidade Departamento >>
<< Declaração dos itens da Entidade Departamento >>
sigl_dpto (1) u2
nome_dpto u30
NOME: arq_func E << Entidade Funcionário >>
<< Declaração dos itens da Entidade Funcionários >>
matr_func (1) u5
nome_func u30
data_nasc u6
dian u2 POS data_nasc
mesn u2 POS data_nasc + 2 <<ou POS dian + 2 >>
anon u2 POS data_nasc + 4 <<ou POS mesn +2 >>
sexo_func(0) u1
salario [12] B4,2 (5,10)
cpf_func (0) u11 UNICA
chave_aux(0) u4 VIRTUAL (anon, mesn)
NOME: arq_lota R << Relacionamento Lotação >>
<< Declaração dos itens do Relacionamento Lotação >>
dpto_lota (arq_dpto) u2
func_lota (arq_func) u5
data_lota d2

2.7.7 Itens com Valor Nulo

A definição de itens com valor nulo permite que um item de dados armazene um valor desconhecido ou não informado. Os tipos de itens de dados que aceitam valor nulo, são:

- U,A(alfabéticos)
- I,B,N,S,C,P<F(numéricos)
- D(Data)

Sintaxe

<*nome_item*> [{[<*rep*>] / (<*ligações*>) / (<*caminho*>)/(0)}]...
... <*tipo*>:<*tamanho*>[,<*num_decimais*>] ... | |
... [(<*num.nivel_ler*>,<*num_nivel_grav*>)] ...
... [{POS <*nome_item_rd*>[+ <*deslocamento*>] /...
... VIRTUAL (<*nome_item_part*>, ..., <*nome_item_part*>)}] ...
... [NULO] **[ATU = {C/S/R/}] [DEL = {C/R/S}][UNICA]**

Utilização

Um item nulo, possui um byte a mais do que seu tamanho original, pois o primeiro byte indica se o ítem é nulo (byte binário é zero) ou não (byte binário é um). Para se atribuir um valor nulo a um ítem do tipo U, A ou D (sem $date), utilize a constante cadeia NULLCHAIN que equivale a uma cadeia vazia (“”). Para se atribuir um valor nulo a um ítem nulo do tipo numérico, utilize a constante numérica NULLNUMBER, que equivale ao número –999999999999999999. Para se saber se o ítem lido é nulo ou não, basta compará-lo a NULLCHAIN se este item for do tipo U,A,D(sem $date) ,ou a NULLNUMBER caso este ítem seja numérico. Um item virtual que possua algum componente nulo, também será nulo.
Ao se utilizar o geral, os valores nulos são exibidos como ????...?
Se for utilizada a máscara “????....?” um ítem nulo será exibido como ????..?

Exemplo

```
banco bdteste 1

nome:arq1 e
  cod1(0)  n5
  nome     u21    nulo
  salario  n10,2  nulo
  admiss   d3     nulo
```

****

Carga do banco definido

****

```
prog
database  bdteste 1 a 2
select a
use arq1
locate
for i = 1 to 10
 if (i%2) = 0
    replace    cod1     with i,;
               nome     with "Teste" ,;
               salario  with  2000,;
               admiss   with  date()
               insert
               if dberr() # 0
                 ? dbmens()
               endif
 else
    replace     cod1    with i,;
                nome    with NULLCHAIN,;
                salario with NULLNUMBER,;
                admiss  with NULLCHAIN
    insert
    if dberr() # 0
     ? dbmens()
    endif
 endif
next
return
```

****

Lê os dados carregados pelo programa acima

****

```
prog

database bdteste 1 a 2

use arq1
locate
Do While .not. eof()
 @01,02 say  cod1
 if nome = NULLCHAIN
   @01,10 say  " Nome NULO"
 else
   @01,10 say nome
 endif
 if salario = NULLNUMBER
   @03,10 say " Salario NULO"
 else
   @03,10 say salario
 endif
```

```
if admiss = NULLCHAIN

  @05,10 say " Data NULA"
else
   @05,10 say admiss

endif
wait
continue
Enddo
```

2.8 Compilação de esquema

O comando DEFINE aciona o compilador de Esquemas e sub-esquemas que darão origem efetivamente a criação do Banco de Dados.

2.8.1 Sintaxe

DEFINE [-<opcao>] <nome_Esquema>

Onde **<opcao>** É uma das opções relacionadas a seguir.

2.8.2 Opções de execução:

-a <nome arquivo> transfere a listagem do Esquema para o arquivo de <<nome Arquivo>>.
-c <nome arquivo> informa o nome de um arquivo que contém os nomes dos arquivos do Banco de Dados que deverão ser recriados, mesmo se existirem.
-e permite a execução do DEFINE em background.
-f suprime a listagem do Esquema.
-p <nome do programa> cria um "pipe" para o programa.
-i direciona a listagem do Esquema para o SPOOL.
<nome_Esquema> é o nome do arquivo fonte que contém o Esquema ou sub-esquema a ser compilado.
-s Indica que não será perguntado ao usuário, se deseja recriar os arquivos do banco de dados. Substitui a opção $controle semarquivo, utilizada no esquema do banco de dados.

3 Organização dos Bancos *Open*BASE

Um Banco de Dados *Open*BASE consiste de um dicionário de dados e de um ou mais arquivos de dados. Um arquivo de dados consiste de um ou mais registros. Um registro consiste de um ou mais itens de dados.

3.1 Dicionário de dados

O dicionário de dados é um arquivo em disco que contém a estrutura de um Banco de Dados e as informações necessárias para o gerenciamento do Banco de Dados.
É através do dicionário que as rotinas de manipulação de dados acessam o Banco de Dados e reconhecem as estruturas de dados de cada arquivo.

3.2 Arquivo de Dados

Os arquivos de dados, armazenam os registros de dados, segundo as estruturas e restrições definidas no dicionário de dados.
Um arquivo de dados no *Open*BASE é categorizado em função dos conjuntos dos objetos básicos do MER original, ou seja, representam os conjuntos de Entidades e Relacionamentos.
Assim sendo, o *Open*BASE oferece dois tipos básicos de arquivos de dados:

- **Arquivo Entidade**
- **Arquivo Relacionamento**

Em condições especiais um Arquivo Entidade pode assumir a qualidade de Arquivo Tabela e um Arquivo Relacionamento a qualidade de Arquivo Consulta.
O uso dos arquivos Tabela e Consulta, ocorre quando existe a necessidade de se definir em um Banco de Dados, arquivos que já foram definidos em outros Banco de Dados, ou seja, implementar interseções de Banco de Dados sem que haja redundância.
Na Linguagem de Definição de Banco de Dados OpenBASE, Arquivos Entidades e Relacionamentos podem ainda ser categorizados em função do MER estendido (MER/E) porém estas categorizações não impõem novas regras ou controles ao *Open*BASE.
Para o *Open*BASE, o uso das extensões propostas pelo MER/E é puramente documentacional, ou seja, todas as regras e controles baseiam-se na implementação do MER original (padrão).
Havendo uso das extensões, o *Open*BASE converte as categorias propostas pelo MER/E para as categorias básicas do MER original. Veja Tabela de Conversões.

Tabela de conversões

MER estendido	MER original / OpenBASE	Obs
Entidade autônoma	Entidade	(**)
Entidade associativa	Entidade	(*)
Entidade de ligação	Entidade	(*)
Entidade composta	Entidade	(**)
Sub-entidade	Entidade	(**)
Entidade complementar	Relacionamento	(**)
Entidade fraca	Relacionamento	(*)
Relacionamento	Relacionamento	(*)

(*) podem ser definidas na LDBD porem são convertidas pelo DEFINE para as categorias do MER original.
(**) não podem ser referenciadas na LDBD; devem ser definidas segundo o MER original.

3.3 Item de dado

Um item de dado consiste de um valor associado a um atributo e corresponde à menor unidade de informação acessível em um Banco de Dados.
A estrutura de um item de dado é parte da estrutura de dados de um registro, assim sendo, compreende parte da estrutura de dados de um arquivo de dados. Estas estruturas estão definidas no dicionário de dados.
Para as aplicações, a seqüência de armazenamento do item de dados no registro e totalmente transparente, já que toda operação em um registro e realizada pelo *Open*BASE, item a item.
Um item no *Open*BASE, pode ser classificado quanto ao Tipo, a Forma e quanto a Ocupação.

3.3.1 Classificação de um item quanto ao Tipo

- Universal (Cadeia de caracteres de qualquer tipo)
- Numérico sem sinal
- Numérico com sinal
- Numérico compactado sem sinal
- Numérico compactado com sinal
- Numérico Inteiro
- Numérico Binário
- Numérico Ponto Flutuante
- Data
- Lógico (booleano)
- Multimídia (Objeto Binário Longo)
- Memo (tipo M e O)

Num esquema, no lugar de se definir um item memo M4 pode ser definido um item tipo O4. Neste caso, no lugar de ser criado um diretório para cada item M4 é criado um arquivo para cada item O4, contendo um texto ou imagem.

3.3.2 Exemplo

nome: arq e
c1 (0) N3
mem1 O4
mem2 O4

São criados automaticamente os arquivos:

nome: oo000002r e
chave(0) I4
valor U500

e

nome:oo000003r
chave(0) I4
valor U500

onde os valores do item chave dos arquivos oo000002 e oo000003 são os valores dos itens mem1 e mem2 quando diferentes de zero. Os nomes dos arquivos são formados de oo + número de seis dígitos sendo este número equivalente a ordem seqüencial que os arquivos (dados e de índices) são gerados após a compilação do esquema.
Na linguagem OPUS pode ser utilizadas as funções MEMOPUT, MEMOGET, MEMODEL, MEMOCOUNT para processar itens tipo O4.
O utilitário bdveri verifica a integridade referencial entre os itens memo (M4 ou O4) e os arquivos no diretório mm<arq>/<item> ou cadeias nos arquivos oo <número>. Para não ser efetuada a verificação especificar a opção –m.

O utilitário bddesc, descarrega itens do tipo M4, de tal forma, que os dados descarregados, possam se carregados para itens tipo O4, com o utilitário bdadic, bastando que após a alteração dos itens M4, para O4, no esquema do banco, o mesmo seja recompilado, e o arquivo alterado seja recriado.

Observações:

- Os Itens Numéricos podem ter ou não casas decimais.
- Itens tipo Multimídia são implementados como arquivos em disco.

3.3.3 Classificação de um item quanto a Forma

- Simples (Indivisíveis)
- Composto (Divisíveis)
- Repetitivos (pseudo multivalorado)

3.3.4 Classificação de um item quanto a Ocupação

- Reais (possui área reservada no registro)
- Virtuais (não possui área reservada no registro)
- Redefinidos (não possui área reservada no registro)

3.4 Arquivos de Índice

Além dos arquivos de dados, o *Open*BASE implementa internamente arquivos de índices. Os arquivos de índices são gerados quando são atribuídas determinadas características especiais aos itens de dados, como:

- Item Determinante (Atributo Determinante / Chave Primária)
- Item Associativo (Atributo Associativo / Chave Estrangeira)
- Item de Busca (Atributo Detalhe / Chave Alternativa)

No *Open*BASE uma chave secundária (estrangeira / alternativa) pode ser chave única. Não confundir esta restrição de unicidade com Chave Primária que por definição é uma chave única, em função de sua própria natureza (Atributo Determinante).

Itens redefinidos ou virtuais podem constituir-se Chaves.

3.5 Arquivo De Recuperação

Um dos aspectos mais importantes na administração de Bancos de Dados é a sua possibilidade de recuperação. O OpenBASE implementa, por padrão, recursos de recuperação de Bancos de Dados, permitindo que transações possam ser desfeitas em condições adversas (rollback / undo).

3.6 Arquivo Diário (opcional)

No Esquema de um Banco de Dados pode ser definida a utilização de um dispositivo de recuperação denominado DIÁRIO que tem por finalidade, armazenar em um arquivo especial, as imagens originais dos dados das transações, antes deles serem modificados e deles após serem modificados.

Este dispositivo permite que uma ou mais transações possam ser desfeitas (rollback) ou refeitas (rollforward / re-do) em condições adversas.

O arquivo Diário e único para todos os Bancos de Dados que utilizem este dispositivo. Este arquivo é criado a partir de um programa especial chamado BDSGBD, o qual é responsável por alguns controles do *Open*BASE.

3.7 Arquivo Transposto

3.7.1 Definição dos arquivos

Um arquivo transposto é definido especificando-se ao lado do nome e tipo a opção transposto ou especificando-se $CONTROLE TRANSPOSTO quando todos os arquivos serão transpostos.

3.7.2 Definição dos itens

Na definição do item pode ser especificado um “domínio” para definir a compressão dos valores. O domínio pode ser uma lista de valores ou intervalo para itens numéricos.

Se especificada um alista de valores (para tipos U, I, N, P e D) serão atribuídos bits para representar estes valores: 1-2 valores 1 bit, 3-4 valores 2 bits, etc.

Se especificado um intervalo (para tipos N, I, P e D) será compactado o valor segundo o intervalo, 1-2 1 bit, 3-4 2 bits etc.

3.7.3 Criação dos arquivos

O arquivo de dados será criado com o cabeçalho e 1 byte para indicar registros excluídos.

Para cada item será criado um arquivo com nome nnn<arq>. O formato do arquivo será relativo acessando-se um valor na posição (n-1)*t onde n é o número do registro e t o tamanho dos valores em bits.

3.7.4 Exemplo

nome: arqtran e transposto
item1 (0) n3

sexo **u1 valores (m,f)**
idade **n2 intervalo (0-99)**

4 Estrutura dos Bancos *Open*BASE

O objetivo deste tópico é prover de informações o usuário, para que ele possa calcular o custo de acesso e o espaço em disco, em função das reais necessidades das Aplicações.

4.1 Dicionário De Dados

A estrutura interna de um dicionário de dados é bastante complexa e sua representação externa mais ainda, o que levaria inúmeras páginas para comentá-la. Desta forma serão apresentadas partes destas estruturas, ou seja, apenas as necessárias para a compreensão do usuário.
O dicionário de dados é um arquivo seqüencial de formato binário.
Seu tamanho e variável em função da quantidade de elementos que compõem o Banco de Dados (arquivos de dados, itens de dados, dentre outros).
Recomenda-se que o usuário verifique o espaço em disco ocupado pelo dicionário, logo após a definição do Banco de Dados. Deve ser utilizado o utilitário BDESPA para esta finalidade.
Uma aplicação que utiliza Banco de Dados, quando executada, leva consigo para a memória, uma imagem do dicionário de dados.
Geralmente o tamanho do dicionário não compromete o compartilhamento da memória do computador. Caso o dicionário seja muito grande, recomenda-se a utilização de SUB-ESQUEMAS, ou seja, um Esquema parcial do Banco de Dados.

4.2 Arquivos Complementares ao Dicionário de Dados

Após a compilação do esquema do Banco de Dados, são gerados os seguintes arquivos com o mesmo nome do Dicionário de Dados, mas com diferente extensão:

<dicion>	Nome do dicionário de Dados.
<dicion>.B	Arquivo criado, quando se utiliza bloqueio de banco.
<dicion>.C	Indica que estão sendo usados domínios para arquivos transposto.
<dicion>.G	Quando se utilizam bancos de dados distribuídos, indica o arquivo gerado na máquina cliente que informa o percurso local do dicionário de dados no servidor.
<dicion>.H	Quando se utilizam bancos de dados distribuídos, este arquivo é gerado em cada máquina que compõe o banco de dados distribuído:
<dicion>.L	Quando se utilizam bancos de dados distribuídos, este arquivo armazena a imagem anterior das transações não completadas nos arquivos que compõem o banco de dados distribuído.
<dicion>.P	Arquivo criado, quando o percurso e nome do dicionário, arquivos e itens, possuírem tamanho maior que 56 posições.
<dicion>.R	Arquivo de recuperação, gerado quando foi especificado o parâmetro <ARQRECUP> na declaração do esquema. O OpenBASE gera sempre este arquivo, anão ser que seja especificado <NAOARQRECUP>.
<dicion>.S	Arquivo que contém as informações de data-hora relativas aos arquivos replicados.
<dicion>.T	Indica que está se utilizando espelhamento de arquivos de dados.
<dicion>.Z	Arquivo com as informações necessárias para refazer as transações. <ARQREFAZ> especificado na declaração de um Banco de Dados OpenBASE.

4.3 Tabela de Execução

Para cada arquivo de dados e reservada uma área no dicionário, denominada Tabela de Execução (TE). Estas tabelas contém os controles de acesso aos arquivos do dados, manipulados pelas aplicações.
As principais informações de controle dentro da TE são:

- Endereço do registro acessado
- Endereço do próximo registro a ser acessado
- Quantidade de registros a serem acessados

4.4 Arquivos de Dados e de Índices

Um **arquivo de dados** é um arquivo relativo. Seus registros são de tamanho fixo, podendo ser dividido em duas áreas distintas (cabeçalho e dados). Um **arquivo de índice** é um arquivo indexado tipo "Btree".
O *Open*BASE implementa dois tipos de arquivos de índices (simples e estruturado), em função da restrição de unicidade das chaves.
Chaves primárias e chave secundária únicas determinam a utilização de índices simples, onde a chave mantém uma relação das mesmas com os registros no arquivo de dados de (1:1).

Chaves secundárias não únicas determinam a utilização de índices estruturados, onde a chave mantém uma relação com o registro no arquivo de dados de (1:n).
No arquivo de dados, para cada chave secundária não única, é criada uma área adicional no registro, representada por duas colunas. Esta área é denominada de cabeçalho e cada coluna desta área corresponde a uma entrada na lista duplamente encadeada.

Tamanho do Registro de Arquivo de Dados

O tamanho em bytes do registro de um arquivo de dados, varia em função do tamanho da área de dados (soma dos tamanhos dos itens de dados do registro) e do tamanho da área de cabeçalhos (soma dos cabeçalhos resultantes das chaves secundárias não unívocas).
Para calcular o tamanho em bytes do registro de um arquivo de dados aplica-se a seguinte fórmula:
TRAD = (QCNU * 6) + TSID

- **TRAD** tamanho do registro
- **QCNU** quantidade de Chaves secundárias não unívocas
- **TSID** tamanho da soma dos itens de dados

Tamanho de Registro de Arquivo de Índice

O tamanho em bytes do registro de um arquivo de índice varia em relação ao seu tipo (simples ou estruturado).
Antes de descrever as fórmulas de cálculo do tamanho dos registros desses arquivos de índices, é importante lembrar que a implementação dos índices está baseada na estrutura de árvore binária balanceada: uma "Btree".
A "Btree" do *Open*BASE, possui controles internos que necessitam de uma área adicional nos registros de índices. Estes controles são denominados de nos da "Btree" e são totalmente transparentes para o usuário. Para não fugir do objetivo deste manual, será descrita somente a sua ocupação, sem entrar em maiores detalhes sobre a arquitetura de uma "Btree".

Tamanho dos arquivos de Índice Simples

Para calcular o tamanho em bytes do registro de um arquivo de índice do tipo simples, aplica-se a seguinte fórmula:

TRIS = TACB + TAE + TC
TRIS tamanho do registro
TACB tamanho da área de controle da "Btree" = 1 byte
TAE tamanho da área de endereço = 4 bytes
TC tamanho da chave

Tamanho dos arquivos de Índice Estruturado

O cálculo do tamanho em bytes do registro de um arquivo de índice do tipo estruturado, é efetuado aplicando-se a fórmula:

TRIE = TACB + TAFR + TAEP+ TAEU + TC
TRIE tamanho do registro
TACB tamanho da área de controle da "Btree" = 1 byte
TAFR tam. área de armazenamento da freqüência = 4 bytes
TAEP tam. área de endereço do primeiro da lista = 4 bytes
TAEU tam. área de endereço do ultimo da lista = 4 bytes
TC tamanho da chave

4.5 Limites de um Banco *Open*BASE

O *Open*BASE foi desenvolvido para atender a grandes aplicações e Sistemas de Informação complexos. Assim sendo, seus limites foram baseados em estruturas sofisticadas de BD‘s e em grandes volumes de dados.
Alguns limites:

Elemento	Limite mínimo	Limite máximo
Arquivos de índices	0	32767
Arquivos de dados	1	32767
Campos por arquivo de dados	1	3000
Chaves por arquivo de dados	0	127
Tamanho de campo (Bytes)	1	(*)4294836,225
Tamanho de chave (Bytes)	1	(*)118

Elemento	Limite mínimo	Limite máximo
Registros por arquivo (dados e índice)	0	4294836,225

(*) O tamanho máximo de um campo e de uma chave varia em função do seu tipo. Os limites máximos acima mencionados, baseiam-se nos limites máximos de um campo tipo multimídia e de uma chave tipo caracter.

5 Recursos do SGBD *Open*BASE

O *Open*BASE permite acesso concorrente a seus Bancos de Dados, ou seja, um Banco de Dados pode ser acessado por varias aplicações concorrentemente.
Um Banco de Dados *Open*BASE pode ser dividido em vários Bancos de dados através do dispositivo de sub-esquemas. Com isto podemos concluir que um Banco de Dados Corporativo pode ser dividido em Bancos de Dados de Assunto e estes por sua vez podem ser divididos em vários Bancos de dados de Aplicação.
O *Open*BASE permite também interseções de Banco de Dados através de referências externas (arquivos TABELA e CONSULTA), evitando desta forma, redundância de dados e num aspecto mais amplo, redundâncias de arquivos inteiros.
Permite ainda que os Relacionamentos entre registros de arquivos distintos sejam efetivamente declarados, possibilitando o controle automático da Integridade Referencial e a "navegação" automática para obtenção de dados.

5.1 Estratégia de Bloqueio

O acesso de atualização concorrente em um Banco de dados é definido pela estratégia de bloqueio, a qual será definida no Esquema do Banco de Dados.
Antes de descrever os tipos de bloqueios é importante explicar o que vem a ser uma transação para o *Open*BASE: é uma seqüência de atualizações, podendo compreender um ou mais registros de um ou mais arquivos.
O início de uma transação é definida no *Open*BASE, por um comando de bloqueio (LOCK) e o seu final por um comado de desbloqueio (UNLOCK). Isto não significa que somente os registros envolvidos na transação sejam bloqueados. Na verdade, o que vai determinar o universo de registros a serem bloqueados, é o tipo do bloqueio escolhido para o Banco de Dados.
Em uma aplicação do usuário escrita em OPUS, as transações que envolvem uma única atualização, não requerem que sejam explicitados os controles de LOCK e UNLOCK, devido ao fato destes controles, neste caso, serem tratados internamente pela OPUS. O OpenBASE implementa os seguintes tipos de bloqueio:

5.1.1 Bloqueio do Banco de Dados

Caso o usuário não defina no Esquema nenhum tipo de bloqueio para o banco de Dados, o *Open*BASE assume o bloqueio automático do Banco de Dados, bloqueando todo o Banco de Dados a cada transação. Este tipo de bloqueio é "default", por ser o mais utilizado, em função das aplicações serem processadas em mono processadores. Isto quer dizer que o computador processa uma instrução de código de máquina por vez. Deste modo o acesso a disco também é feito um de cada vez.
As transações muito grandes em Banco de Dados gigantescos e de intensa concorrência podem inviabilizar a utilização do bloqueio de Banco de dados, caso contrário, o tempo que o Banco de Dados ficará bloqueado é praticamente imperceptível.
Para o *Open*BASE, este tipo de bloqueio é bastante eficiente, econômico e seguro:

- Não ocupa memória para os controles de concorrência, uma vez que bloqueia a cada transação o dicionário de dados.
- Inibe a proliferação dos arquivos de segurança para rollback" (ARQRECUP).
- Reduz sensivelmente o processamento, em função de não ter que administrar varias transações concomitantes.
- Garante com segurança que jamais ocorrerá "deadlock", ou seja, duas transações ficarem presas indefinidamente, em função de uma depender do desbloqueio da outra e vice-versa.

5.1.2 Bloqueio de Arquivos (BLOQARQ)

Este tipo de bloqueio é útil quando a natureza das aplicações se restringe a acessar registros de um único arquivo, ou seja, cada aplicação acessa um arquivo.

5.1.3 Bloqueio de Página (BLOQPAG)

Este tipo de Bloqueio define que todas as páginas onde estão contidos os registros a serem atualizados pelas transações, sejam bloqueados.
O tamanho de uma página de um arquivo, eqüivale 1024 bytes (1 Kb) e compreende um (ou parte de um) ou mais registros.

5.1.4 Bloqueio de Registro (BLOQREG)

Este tipo de Bloqueio define que todos os registros a serem atualizados pelas transações sejam bloqueados.
Observações :

- Quanto maior for o universo de registros bloqueados, aumenta a probabilidade de "deadlock".
- Em bloqueios de arquivo, página ou registro, os mesmos podem ser exclusivos ou compartilhados, desde que estas qualidades sejam associadas as formas disponíveis de leitura.
- Para todos os tipos de bloqueios, salvo o Bloqueio de Banco de Dados, o *Open*BASE reserva uma área na memória para armazenar os controles dos bloqueios das transações concomitantes.

5.2 Definição do Relacionamento

As Entidades no *Open*BASE podem se relacionar a outras Entidades diretamente ou indiretamente (Relacionamento entre registros de arquivos distintos).
Declarar um Relacionamento no *Open*BASE não é simplesmente declarar em um arquivo de dados, os itens associativos (chave estrangeira das Entidades envolvidas no Relacionamento): é estabelecer as ligações e caminhos a serem percorridos pelo Banco de Dados.

5.2.1 Estabelecendo a Ligação

Toda Entidade é identificada através de seu atributo determinante (chave primária). Aliado a este atributo é informado o número de ligações, ou seja, em quantos arquivos de dados a chave primária da Entidade, é referenciada como chave estrangeira.
Numa relação entidade (tipo E ou T) o número de ligações pode ser especificado como *, o que será automaticamente substituído pelo número de referências nos arquivos posteriormente definidos, por exemplo

nome: PESSOA e
NOMEP (*) u30 <<(*) será substituído por (1) >>
IDADE n2
nome: FILHOS r
NOMEP1 (PESSOA) U30
IDADEF n2

5.2.2 Estabelecendo a Caminho

Todo Relacionamento é identificado através dos atributos associativos (chave estrangeira). Aliado a estes atributos, são informados os nomes das Entidades aos quais eles se referem, ou seja, as procedências das chaves estrangeiras.

5.2.2.1 Vantagens:

Após definir as ligações e caminhos dos Relacionamentos, o *Open*BASE administra automaticamente a Integridade Referencial, além de permitir, para obtenção de dados (se não existir itens com nomes repetidos no Banco de Dados), junção de dados dos registros relacionados em vários níveis de Relacionamento.

5.2.2.2 Observação:

Na construção do esquema de dados, não é preciso que o usuário se preocupe com a ordem em que os arquivos foram declarados, isto é, os arquivos ou tabelas mestres podem ser definidos após os arquivos ou tabelas ligados a eles por chave estrangeira

5.2.2.3 Exemplo

banco bd_empresa 1
nome: empregado E
emp_mat (0) u06
emp__nome u40
emp_depart (depart) u03
nome:depart E
dep_cod(1) u03
dep_nome u30

A definição de todas as ligações e caminhos dos eelacionamentos é efetuada no Esquema do Banco de Dados, ou seja, estarão definidas também no dicionário de dados. O fato de ter que definir estas informações no esquema, não determina na implementação física do BD, ponteiros (elos explícitos) entre registros. Todas as ligações são efetuadas de forma lógica, através do dicionário de dados.

5.3 Integridade Referencial

É a garantia de que não haverá perda de referências em um Banco de dados, ou seja, jamais será referenciada em um Relacionamento uma entidade que não exista e jamais será removida uma Entidade se ela estiver sendo referenciada.
Para se usufruir deste controle automático do *Open*BASE, é necessário definir os Relacionamentos entre as Entidades.

5.4 Redução de Arquivos

Quando entre as Entidades observa-se um Relacionamento de classes 1:N) ou (1:1), uma das Entidades poderá absorver o Relacionamento.

A escolha de qual das Entidades irá absorver o Relacionamento, será determinada a partir de uma análise de otimização.
Por exemplo, no Relacionamento entre DEPARTAMENTOS e FUNCIONÁRIOS (1:N), a Entidade FUNCIONÁRIO pode absorver o Relacionamento LOTAÇÃO, através da declaração da chave estrangeira (referência ao atributo determinante da Entidade DEPARTAMENTO), uma vez que todos os funcionários estão lotados em um departamento.
Outro exemplo, é o do Relacionamento CHEFIA, entre as mesmas Entidades, onde a Entidade encarregada de absorver o Relacionamento deve ser a Entidade DEPARTAMENTOS, pois não justifica declarar o Relacionamento dentro da Entidade FUNCIONÁRIOS, já que nem todos os funcionários são gerentes de departamento.
Relacionamentos de Classe de (N:N) são sempre implementados através de Arquivos Relacionamentos.

5.5 Sub-esquemas

No *Open*BASE, um Banco de Dados pode ser manipulado de maneira Extremamente flexível com relação as aplicações e suas respectivas necessidades de manipulação de dados. Isto significa dizer que dado um Esquema de um Banco de Dados qualquer, podem ser definidos também sub-esquemas que irão manipular porções deste Banco de Dados.
O Esquema do qual foram tirados os sub-esquemas, passa a ser Chamado de Esquema Global. Assim sendo, em um sub-esquema pode ser "vistas" partes do Esquema Global. A definição destas partes, é feita pelo projetista do Banco de Dados e são especificas para cada aplicação.
Para que este recurso seja utilizado no *Open*BASE, todos os arquivos declarados no sub-esquema devem existir no Esquema Global.
É obrigatório que o tipo do arquivo no sub-esquema deva ser compatível com o do Esquema Global. Da mesma forma, os itens declarados nos arquivos do sub-esquema devem encontrar correspondentes no Esquema Global, conservando também o tipo e tamanho do item.
Um dos principais motivos para a implementação deste recurso no *Open*BASE, é a considerável economia de memória principal que pode ser obtida, pois o dicionário gerado a partir do sub-esquema será menor do que aquele gerado no Esquema Global.

5.6 Interseção de Bancos de Dados

O *Open*BASE permite interseções de Bancos de Dados, através de referências externas aos Arquivos Entidades e Relacionamentos. Estas referências são tratadas pelo *Open*BASE como Arquivos TABELA e CONSULTA.
Arquivos tabela:
São arquivos externos ao Banco de Dados onde estão sendo declarados. Se referem a arquivos entidades pertencentes a outros Banco de Dados.
Arquivos consulta:
São arquivos externos ao Banco de Dados onde estão sendo declarados. Se referem a arquivos relacionamentos pertencentes a outros Banco de Dados.
Ao ser declarado um arquivo Tabela ou Consulta, deve-se especificar a sua procedência, ou seja, a qual Banco de Dados ele pertence efetivamente. Por serem arquivos externos pertencentes a outros BDs, estes só poderão ser lidos e nunca atualizados. Na declaração de um arquivo Tabela ou Consulta não é obrigatório declarar todos os itens de dados que estão definidos nos arquivos aos quais se referenciam (Entidades e Relacionamentos), porém todos os itens chaves devem ser declarados, na mesma ordem em que o foram, nos respectivos arquivos que estão sendo referenciados.

5.7 Carga de Arquivos em Memória

A carga de arquivos em memória serve para agilizar o tempo de acesso a um banco de dados Openbase. Esta opção permite que arquivos de dados Openbase sejam carregados em memória compartilhada, melhorando consideravelmente o tempo de acesso ao banco de dados. Deve ser utilizada apenas quando o tempo de acesso ao banco de dados for um fator determinante.
Além dos arquivos de dados são carregados os arquivos de índice também. A atualização destes arquivos em memória é permitida , entretanto o tempo de acesso aos mesmos é prejudicado.

5.7.1 Sintaxe

Criar um arquvo texto , com o mesmo nome do dicionário de dados e com extensão .ini
Este arquivo deverá ser preenchido da seguinte forma:

```
UNIX:       shmkey = <numero>
            shared  = <arquivo>
             .......
              ......
            shared  = <arquivo>
WINNT
            shared  = <arquivo>
             .......
              ......
```

shared = <arquivo>

5.7.2 Argumentos

número - número da memória compartilhada a ser criada.
nome - arquivo do banco a ser carregado em memória

5.8 Dicionário de Dados e Arquivos Locais

O OpenBASE permite replicar parte de um banco de dados de uma máquina servidor para uma máquina cliente, melhorando, assim, o tempo de acesso.

5.8.1 Sintaxe:

No esquema do banco de dados deve existir:

banco <nomebanco> <seg> ..... [LOCAL=<diretório na máquina cliente onde será copiado o dicionário de dados>]
nome: <nomearquivo> <tipo> ..... [LOCAL=<diretorio na máquina cliente onde será copiado o arquivo de dados>]

5.8.2 Utilização

Esta opção deve ser utilizada quando o tempo de acesso ao banco de dados é fundamental.
O arquivo de dados copiado para máquina cliente, não é atualizável. Esta atualização é feita diretamente na máquina servidor. Apenas quando o banco for aberto novamente , o arquivo de dados atualizado será copiado para máquina cliente. O tempo de acesso ao banco de dados é diminuído, quando existem muitas consultas ao mesmo. Para se acessar esta opção, utilize,

OPUS:	**$local= "Nome/IP máquina servidor)"**
set LOCAL on	
dupwin32.dll	**LigaOpcao("local")**
GERAL	**ligue LOCAL**

No caso da opus, deve-se utilizar a biblioteca **libbddup.a** (unix) ou **libbddup.lib** (windows).