1 Introdução

A denominação ***Open*BASE** se refere a uma família completa de produtos para desenvolvimento de sistemas aplicativos. Os produtos da família ***Open*BASE** são agrupados em duas grandes categorias:

1. Sistema Gerenciador de Banco de Dados (SGBD)
2. Ambiente integrado de desenvolvimento de sistemas

Os Sistemas Gerenciadores de Bancos de Dados (SGBD) são caracterizados, principalmente, pelos seus modelos de implementação. O ***OpenBASE*** aderiu ao modelo relacional por causa das suas vantagens. Atualmente, grande parte do mercado de SGBDs corporativos são produtos relacionais.
O modelo lógico de Banco de Dados adotado constitui apenas uma das características de um SGBD. O **Modelo Relacional** visa prover mais **flexibilidade** na organização de grandes bancos de dados e eliminar os problemas encontrados nos modelos tradicionais **hierárquicos** e **em rede**, assim como nos modelos baseados em **listas invertidas**. O preço a pagar por causa dessa maior flexibilidade, oferecida pelo modelo relacional, recai sobre o desempenho, nem sempre comparável ao conseguido nos outros modelos (menos lógicos e mais físicos) de implementação de Bancos de Dados.
Através de sua longa história, os sistemas de banco de dados tiveram ampla utilização. Porém, atualmente, novas aplicações demandam requisitos de modelagem de dados não disponíveis nos modelos existentes. Surgem, deste modo, os modelos relacionais estendidos ou relacionais objeto, cuja evolução converge para os chamados **bancos inteligentes**, **bancos universais**, **bancos orientados a objetos** e **bancos orientados a recuperação de informações**.

1.1 O gerenciador De Bancos *Open*BASE

O SGBD ***OpenBASE*** inclui dois servidores de banco de dados que se integram e se completam: O **OpenServer** e o **TsqlServer**. Estes servidores de Bancos de Dados ***OpenBASE*** possuem as seguintes características básicas:

- Banco de Dados multiplataforma
- Banco de Dados com arquitetura cliente-servidor
- Banco de Dados distribuído
- Serviços de administração e controle de Bancos Dados
- Linguagens de definição e manipulação de Bancos de dados
- Ambiente integrado de desenvolvimento de aplicativos
  - Linguagens e ferramentas de programação de aplicativos
  - Linguagem de consulta relacional SQL
  - Interfaces para ferramentas RAD (Rappid Application Development)

1.2 Características do gerenciador *Open*BASE

O ***OpenBASE*** se caracteriza pela implementação de um conjunto básico de facilidades e serviços voltados para os seguintes objetivos:

- Controle centralizado dos Dados através do Dicionário
- Integridade e consistência das informações e relacionamentos
- Tratamento eficiente das situações de contenção e concorrência entre transações
- Implementação de mecanismos de segurança no acesso às Bases de Dados
- Disponibilidade, desempenho e tempo de resposta satisfatórios
- Integração de serviços utilitários para carga, descarga, cópia e recuperação dos Dados on-line (on-the-fly)
- Implementação de rotinas e procedimentos de Backup, Restauração e replicação dos Dados on-line (on-the-fly)
- Facilidades de acesso aos Dados através da utilização de diversos mecanismos e interfaces

1.3 Documentação dos produtos *Open*BASE

Consulte nossa página Internet para acessar a documentação específica correspondente a cada um dos produtos básicos da família ***Open*BASE**. Você poderá navegar através das páginas desses documentos assim como consultar por assunto, obter cópias e imprimir os tópicos de interesse.

2 Ambientes *Open*BASE em rede

O ***OpenBASE*** implementa o modelo de processamento distribuído, que inclui a arquitetura cliente-servidor, utilizando os protocolos de rede e os mecanismos padrão de comunicação entre processos. O ***OpenBASE* Servidor** e o ***OpenBASE* Cliente**, por serem sistemas abertos, ambos executam em sistemas operacionais UNIX, Windows 9X, Windows NT e Windows 2000.

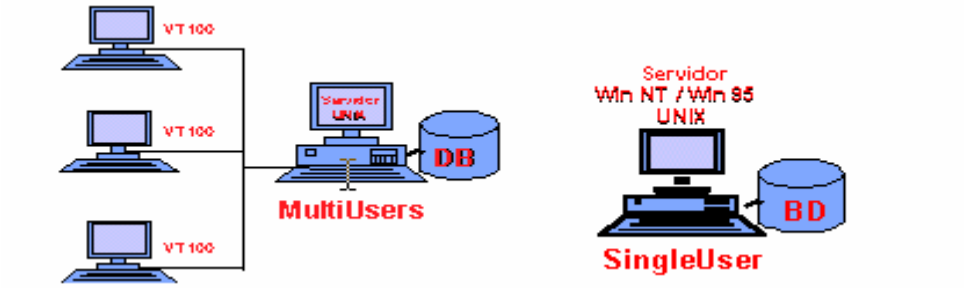
2.1 Arquiteturas *Open*BASE

O ***OpenBASE*** está disponível em três arquiteturas:

- Central
- Cliente/Servidor
- Cliente/MultiServidor (Bancos de Dados distribuídos).

2.1.1 OpenBASE com arquitetura central

Na arquitetura Central, os Bancos de Dados, os Servidores e os aplicativos ***OpenBASE*** residem no mesmo equipamento, interagindo com os usuários através de conexões Telnet, seja utilizando interfaces seriais ou redes locais.

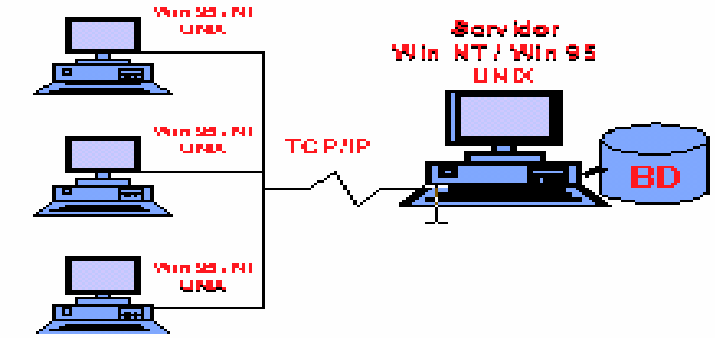


2.1.2 OpenBASE com arquitetura cliente-servidor

O modelo de computação cliente-servidor é um caso particular (e o mais comum) do modelo de computação distribuída. Um sistema pode ser cliente ou servidor, dependendo do relacionamento em que estiver envolvido, ora provendo ou solicitando serviços. Na prática, com o objetivo de otimizar a relação custo/benefício, cada sistema se especializa nas funções de cliente ou de servidor.

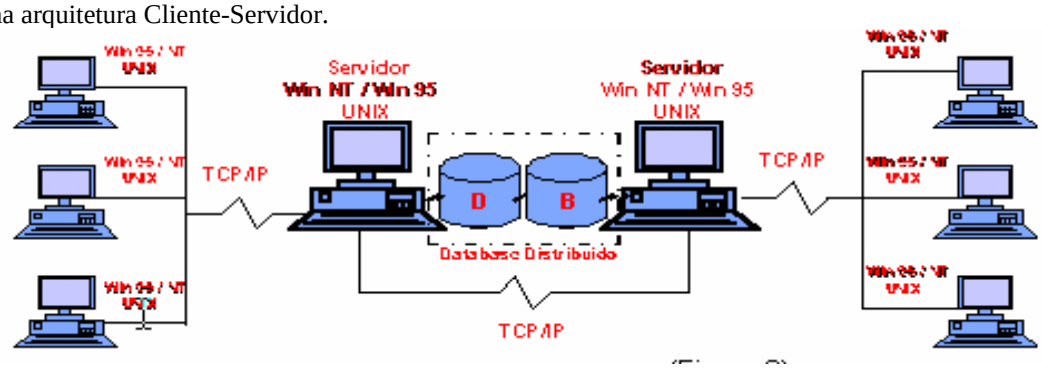
Na arquitetura Cliente-Servidor, os bancos de dados ***OpenBASE*** e os servidores de bancos de dados residem em um único equipamento (estação servidor), sendo que os programas aplicativos que utilizam esses bancos de dados são executados em outros equipamentos (estações cliente).

A grande vantagem desta arquitetura, além da descentralização do processamento, é a diminuição do tráfego na rede, uma vez que todas as operações de banco de dados são realizadas nos servidores.



2.1.3 OpenBASE com arquitetura Cliente/Multiservidor

Na arquitetura Cliente/MultiServidor os bancos de dados são distribuídos em mais de um equipamento. Neste caso, são instalados, em cada equipamento, os servidores de banco de dados, que desempenham os papeis de servidores e clientes. Os programas que utilizam os banco de dados distribuídos são executados em equipamentos Cliente, assim como na arquitetura Cliente-Servidor.

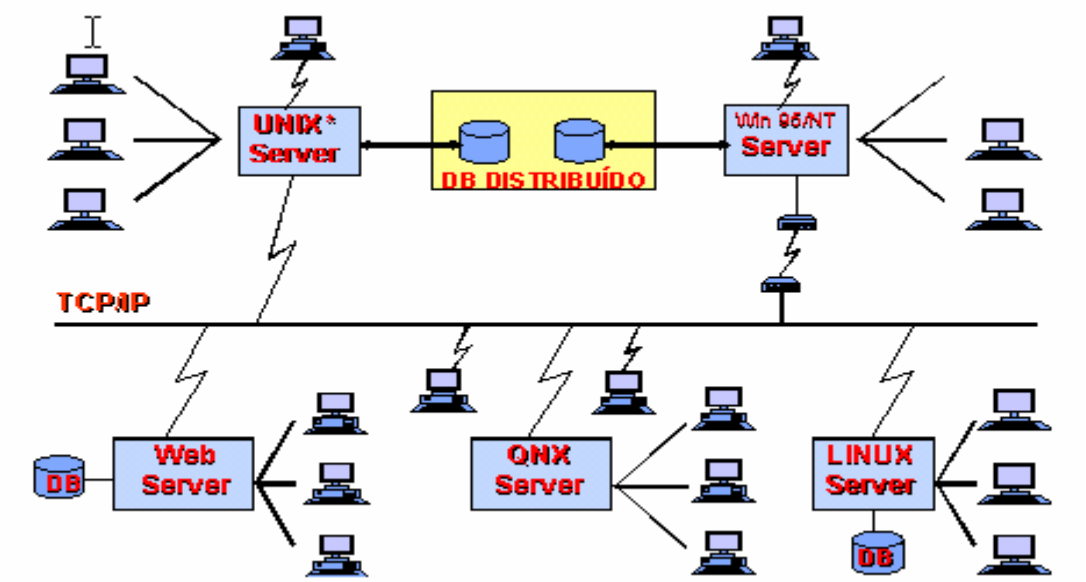


2.2 Plataformas *Open*BASE

O ***OpenBASE*** é um Sistema de Banco de Dados aberto, obedecendo aos padrões internacionais da tecnologia da informação e possuindo, em conseqüência, as características fundamentais de **interoperabilidade e portabilidade** que permitem:

- a sua utilização em plataformas de diferentes fabricantes
- a transferência entre plataformas sem necessidade de modificar o aplicativo nem o esquema lógico dos Bancos de Dados
- a escolha de equipamentos de diferentes portes, sem perda do investimento representado pelo código desenvolvido e pelos dados acumulados

O ***OpenBASE*** executa em plataformas UNIX e Windows nas arquiteturas acima referenciadas. Atualmente, são distribuídas versões para Windows 9X, Windows ME, Windows NT, Windows 2000 assim como versões para os seguintes "sabores" UNIX: LINUX , SOLARIS, AIX, HP-UX, SCO, FreeBSD e QNX.



3 Recursos dos ambientes *Open*BASE

O Sistema Gerenciador de Bancos de Dados ***OpenBASE*** implementa todos os recursos e facilidades inerentes ao modelo relacional adotado. Entretanto, cabe ressaltar:

- 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 ***OpenBASE*** pode ser dividido em vários Bancos de dados através de Subesquemas. 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 ***OpenBASE*** permite 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.
- O ***OpenBASE*** 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 e processamento das informações

A seguir detalhamos alguns dos recursos básicos dos ambientes ***OpenBASE*.**

3.1 Segurança e controle nos ambientes *Open*BASE

...
...
...

3.2 Mecanismos de bloqueio nos ambientes *Open*BASE

Os mecanismos de controle de acesso para atualização concorrente em Bancos de Dados OpenBASE são estabelecidos conforme as estratégias de bloqueio especificadas no Esquema do Banco de Dados. Para melhor compreensão dos mecanismos de bloqueio adotados, é importante compreender o que seja uma transação ***OpenBASE*.**
Uma transação ***OpenBASE*** é constituyída por uma determinada seqüência de atualizações nas Bases de Dados, podendo compreender um ou mais registros de um ou mais arquivos do Banco. O início de uma transação é definida por um comando de bloqueio (LOCK) e o seu final por um comando 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 de bloqueio escolhido para o Banco de Dados.
Em uma aplicação do usuário escrita na linguagem OPUS, as transações que envolvem uma única atualização não precisam explicitar os controles de LOCK e UNLOCK, pois estes são supridos internamente pela OPUS, de forma automática.
O ***OpenBASE*** implementa os mecanismos de bloqueio apresentados a seguir.

3.2.1 Bloqueio do Banco de Dados

Caso o usuário não especifique no Esquema do Banco nenhum tipo de bloqueio, o ***OpenBASE*** assume automaticamente (por *default*) um bloqueio por Banco, implementando em cada transação o bloqueio de todo o Banco de Dados.
Transações complexas, utilizando grandes Banco de Dados com intensa concorrência podem acarretar problemas de performance na utilização do bloqueio de Banco de dados. Em casos normais, o tempo que o Banco de Dados ficará bloqueado é praticamente imperceptível.
Para o ***OpenBASE***, este tipo de bloqueio é bastante eficiente, econômico e seguro, sem possibilidade de ocorrer "deadlock".

3.2.2 Bloqueio de Arquivos

Este tipo de bloqueio é útil quando os aplicativos se restringem a acessar registros de um único arquivo, ou seja, cada aplicação acessa um arquivo do Banco de Dados.

3.2.3 Bloqueio de Chave

Este tipo de bloqueio define que todos os registros que contiverem a mesma chave estarão bloqueados. São bloqueadas as chaves primárias que se relacionam com os registros de uma transação e, em conseqüência, todos os registros que possuírem chaves estrangeiras, que referenciam as chaves primárias bloqueadas, estarão também bloqueados.

3.2.4 Bloqueio de Página

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 equivale a 1024 bytes e compreende um registro, parte de um registro, ou vários registros.

3.2.5 Bloqueio de Registro

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

Observações gerais sobre bloqueio:

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

3.3 Implementação dos Relacionamentos

As Entidades no ***OpenBASE*** podem se relacionar com outras entidades direta ou indiretamente (relacionamento entre registros de arquivos distintos). Declarar um relacionamento no ***OpenBASE*** não é simplesmente declarar os itens associativos, ou seja, a chave estrangeira das entidades envolvidas nos relacionamentos. Mais do que isso, é estabelecer as **ligações** e os **caminhos** a serem percorridos pelo Banco de Dados.

3.3.1 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 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.

Exemplo:

nome: PESSOA e
NOMEP (*) **u30 << (*) será substituído por (1) >>**

IDADE **n2**
nome: FILHOS r
NOMEP1 (PESSOA) **U30**
IDADEF **n2**

3.3.2 Estabelecendo o 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.
Após definir as ligações e caminhos dos relacionamentos, o ***OpenBASE*** 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.
A definição de todas as ligações e caminhos dos relacionamentos 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.

3.4 Integridade Referencial

A integridade referencial implementa a consistência das referências nos relacionamentos em um Banco de dados garantindo que:

- jamais será referenciada uma entidade que não exista
- jamais será removida uma entidade que estiver sendo referenciada

Para efetivar a integridade referencial é necessário apenas definir os relacionamentos entre as entidades. O OpenBASE implementa automaticamente este recurso.

3.5 Subesquemas e visões

Bancos de Dados podem ser manipulados de maneira extremamente flexível com relação aos aplicativos. Dado um Esquema de um Banco de Dados qualquer, podem ser definidos subesquemas que irão manipular visões (ou porções) deste Banco de Dados.
O esquema base original, a partir do qual foram definidos os subesquemas, passa a ser chamado **esquema global**.
Assim sendo, um subesquema apresenta visões ou partes do esquema base global conforme as especificações dos projetistas dos Bancos de Dados e suas respectivas aplicações.
Os arquivos declarados no subesquema devem existir no esquema global, sendo que os atributos dos arquivos definidos no subesquema devem ser compatíveis com os atributos do esquema global, ou seja, os itens declarados nos arquivos do subesquema devem encontrar itens correspondentes no esquema global, conservando seus tipos e tamanhos.

3.6 Interseção de Bancos

O ***OpenBASE*** permite interseções de Bancos de Dados, através de referências externas aos Arquivos Entidades e Relacionamentos. Estas referências são tratadas pelo ***OpenBASE*** como Arquivos TABELA e CONSULTA.

3.7 Arquivos Tabela e Arquivos Consulta

São arquivos externos ao Banco de Dados onde estão sendo declarados e 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.

4 Estrutura dos ambientes *OpenBASE*

Os elementos estruturais do ambiente integrado constituído pelos produtos da família ***OpenBASE*** são agrupados conforme as seguintes categorias:

- Gerenciadores (Servidores) do ambiente ***Open*BASE**
- Sistema de Definição de Banco de Dados
- Utilitários para administração de Bancos de Dados
- Linguagens de Consulta e Atualização dos Bancos de Dados
- Ferramentas SQL no ambiente ***Open*BASE**
- Linguagens para desenvolvimento de aplicativos – **OPUS** e **OpusWin**
- Sistema de controle de impressão para ambienters WIN32 - **OpusRel**

- Interface visual para desenvolvimento de aplicativos - **VisualOpus**
- Ambiente integrado para desenvolvimento de aplicativos - **OpenIDE**
- Ambiente integrado para administração de Bancos de Dados – **OpenDBA**
- Desenvolvimento rápido de aplicativos (**RAD**) no ambiente ***Open*BASE**
- Desenvolvimento de aplicativos Internet e Intranet - **OpusWEB**

A seguir apresentamos com maiores detalhes os componentes acima referenciados.

4.1 Elementos dos Bancos de Dados *Open*BASE

Um Banco de Dados ***OpenBASE*** consiste de um Dicionário de Dados e de um ou mais arquivos de dados com seus respectivos índices.

4.1.1 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.

4.1.2 Arquivos de Dados

Os arquivos de dados armazenam os registros de dados, segundo as especificações contidas no Dicionário de Dados. Arquivos de dados no ***OpenBASE*** são categorizados em função dos conjuntos de objetos básicos do MER original, ou seja, representam os conjuntos de Entidades e Relacionamentos. Assim sendo, o ***OpenBASE*** 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.

4.1.3 Arquivos de Índice

Além dos arquivos de dados, o ***OpenBASE*** 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 ou Chave Primária)
- Item Associativo (Atributo Associativo ou Chave Estrangeira)
- Item de Busca (Atributo Detalhe ou Chave Alternativa)

No ***OpenBASE*** uma chave secundária (estrangeira ou alternativa) pode ser chave única. Não devemos confundir esta restrição de unicidade com as características da Chave Primária que é chave única em função de sua própria natureza (Atributo Determinante). Itens redefinidos ou virtuais podem constituir-se também em chaves. O ***OpenBASE*** implementa vários tipos de índices em função da unicidade ou multiplicidade das chaves:

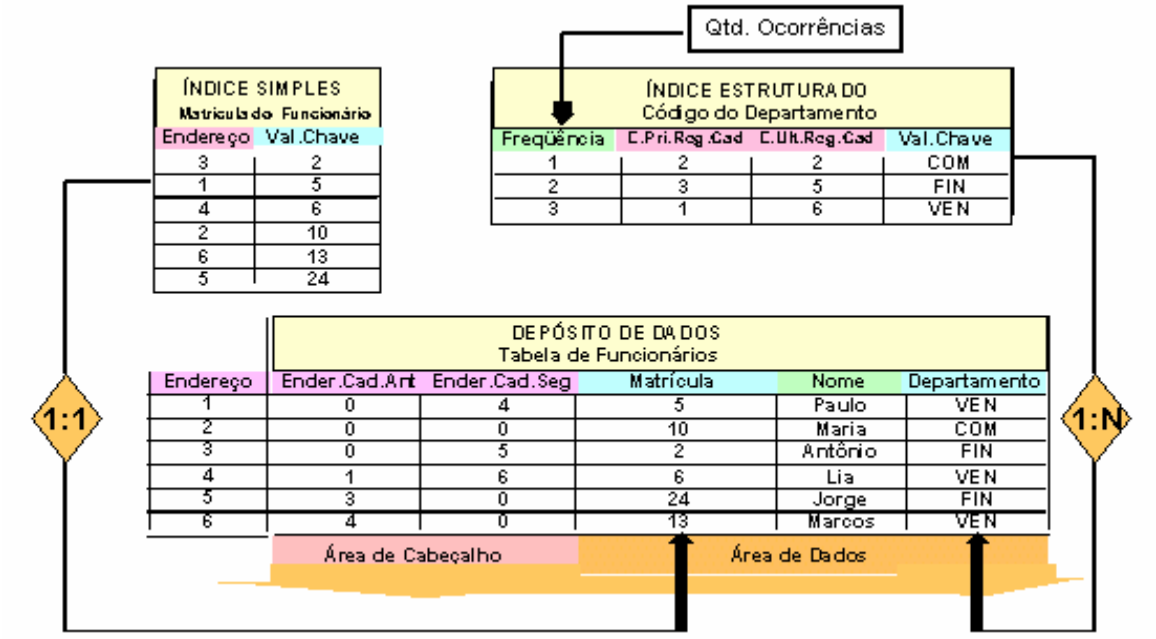
- Índices simples
- Índices estruturados
- Índices múltiplos

4.1.3.1 Índices simples

Os índices simples são construídos quando ocorrem chaves primárias (únicas por definição) ou chaves secundárias únicas, onde as chaves mantém uma relação 1:1 com os registros no arquivo de dados.

4.1.3.2 Índices estruturados

Os índices estruturados são determinados pela ocorrência de chaves secundárias não únicas, onde as chaves mantém uma relação 1:N com os registros no arquivo de dados.
Para cada chave secundária não unívoca é criada, nos registros do arquivo de dados, uma área adicional denominada **cabeçalho** correspondente a entradas numa lista duplamente encadeada que gerencia os registros que contém as mesmas chaves.



4.1.3.3 Índices múltiplos

…
….
…

4.1.4 Arquivo de Recuperação

Este componente de Bancos de Dados ***OpenBASE*** é opcional. No Esquema de um Banco de Dados pode ser definido o recurso de recuperação através da opção **ARQRECUP** cuja finalidade é armazenar no **arquivo de recuperação** as imagens originais dos dados das uma transações **antes** de serem modificados.
Este recurso permite que uma ou mais transações possam ser **desfeitas** através dos procedimentos rollback ou undo). A quantidade de Arquivos de Recuperação deste tipo varia em função da estratégia de bloqueio adotada.

4.1.5 Arquivo Diário

Este componente de Bancos de Dados ***OpenBASE*** é opcional. No Esquema de um Banco de Dados pode ser definida a utilização de um recurso de recuperação através das opção **DIARIO** ou **DIAREC**.
Especificando a opção **DIARIO,** o OpenBASE armazena no **arquivo diário** (journal) as informações lógicas dos dados **depois** de serem modificados permitindo, assim, que uma ou mais transações sejam **refeitas** a qualquer momento, a partir de uma determinada hora ou a partir de uma determinada transação OpenBASE. Especificando a opção **DIAREC,** o OpenBASE armazena no **arquivo diário** (journal) as informações lógicas dos dados **antes** de serem modificados **e depois** de serem modificados permitindo, assim, que uma ou mais transações sejam **desfeitas ou refeitas** a qualquer momento, a partir de uma determinada hora ou a partir de uma determinada transação OpenBASE.
As opções DIARIO e DIAREC implicam na utilização automática da opção ARQRECUP. O arquivo diário pode ser único para todos os Bancos de Dados que utilizem estas opções ou não, dependendo das especificações no esquema. O arquivo diário é criado pelo programa BDSGBD, responsável por alguns controles do ambiente operacional dos Bancos de Dados ***OpenBASE***.

4.1.6 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.

4.2 Implementação de Bancos de Dados *Open*BASE

O processo de implementação de Bancos de Dados ***OpenBASE*** inclui duas etapas:

- Elaboração do Projeto Lógico – Modelo Entidade-Relacionamento (MER)
- Elaboração do Projeto Físico – Esquema do Banco de Dados

A definição de Bancos de Dados ***OpenBASE*** inclui a elaboração de um **esquema**, que é a tradução do projeto lógico (MER) para a linguagem de definição de Dados do ***OpenBASE***. O esquema descreve detalhadamente todas as informações necessárias que constituem o Dicionário de Dados do Banco de Dados, tais como:

- Nome do Banco de Dados OpenBASE
- Senha e estratégia de bloqueio e recuperação
- Nome e tipo dos arquivos de dados (Entidades e Relacionamentos)
- Nome, tipo e tamanho dos atributos ou itens
- Especificação da estruturas de Chaves (primária, secundária, estrangeira …)
- Regras de integridade referencial

O arquivo texto que contém o esquema é submetido ao programa "DEFINE" para processá-lo, sendo criado, efetivamente, um Banco de Dados ***OpenBASE*** com os seguintes componentes básicos:

- Dicionário de Dados
- Arquivos de Dados (ou Tabelas)
- Arquivos de Índices
- Arquivos complementares

O ***OpenBASE*** possui sua própria Linguagem de Definição de Dados (LDD) que permite construir o **esquema conceitual** de cada Banco de Dados, descrevendo sua estrutura completa, concentrando-se, especificamente, na arquitetura e descrição de entidades, atributos, relacionamentos, procedimentos de integridade e recuperação, estratégias de bloqueio, níveis de acesso e regras de utilização.
Existem diferentes terminologias associadas aos conceitos básicos do modelo relacional, dependendo do contexto. A Linguagem de Definição de Dados do ***OpenBASE***, baseada no modelo conceitual Entidades - Relacionamentos (E-R), utiliza a terminologia comum de processamento de dados, ou seja, **arquivo**, **registro** e **item** (ou **campo**). O modelo relacional formal apresenta a terminologia *relação, tupla e atributo* ou, ainda, *tabela, linha e coluna* (no contexto do SQL e modelo relacional informal).
O esquema conceitual de um Banco de Dados ***OpenBASE*** reside num arquivo texto, que pode ser construído ou modificado através de um editor de texto ou utilizando os recursos gráficos e visuais do ***OpenBASE*** tais como OpenIDE, OpenDBA e Visual OPUS.
Um dos componentes da Linguagem de Definição de Dados, denominado DEFINE, compila o esquema, processa seus componentes de acordo com a sintaxe LDD do ***OpenBASE*** e constrói ou modifica os objetos do Banco de Dados, tais como dicionário de dados, tabelas de controle, arquivos de dados e arquivos de índice. Depois de concluída a compilação do esquema e a definição dos Bancos de Dados, os usuários podem efetuar as tarefas básicas de incluir, selecionar, alterar e suprimir objetos. Vejamos algumas características importantes em relação ao esquema:

- O nome do arquivo que contém o fonte do Esquema pode ser qualquer um, desde que não transgrida as restrições do Sistema Operacional instalado no equipamento do usuário. É conveniente utilizar sufixos ou qualificadores tais como **.e** ou **.esq**.
- Para continuar na linha seguinte uma série de declarações do esquema, deve-ser colocado um **;** (ponto e vírgula) no final da linha a ser continuada
- O Esquema pode ser escrito em letras maiúsculas e minúsculas
- Para especificar na listagem do Esquema, um salto de linha, deve ser utilizado o controle $L na primeira coluna de uma linha em branco
- Para comandar um salto de página, deve ser utilizado o controle $P na primeira coluna de uma linha em branco

Quando o processamento do esquema é completado com êxito, o DEFINE imprime um sumário contendo as seguintes informações: nome e tipos dos arquivos, quantidade de itens por arquivo, tamanho dos registros, tamanho dos cabeçalhos, numero de ligações entre as entidades, quantidade de chaves por arquivo, ... etc ...

*4.2.1 Esquema dos Bancos de Dados Open***BASE**

Neste tópico apresentamos uma referência da Linguagem de Definição de Banco de Dados do ***OpenBASE*** comentando, passo a passo, sua sintaxe e mostrando alguns exemplos.

[<< <*comentário*> >>]
[$CONTROLE <*opção*>, ... , <*opção*>]
BANCO [<*percurso*>] <*nome_bd*> <*código_de_seguranca*>...
[{ARQRECUP / DIARIO / DIAREC}]...
[{BLOQARQ / BLOQCHA / 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]
...
...

4.2.1.1 Comentários no esquema

Dentro do esquema, podem ser incluídos comentários, podendo estar em qualquer lugar e ocupar várias linhas confirme a seguinte Sintaxe

[<< <comentário> >>]

Onde: "**<<**" marca o inicio de um comentário e ">>" marca o fim de um comentário

Exemplos:

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

4.2.1.2 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).

Exemplo:

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

Consulte o manual de referência "DEFINE" e as atualizações apresentadas nos BITs (Boletins Informativos Técnicos) para obter as informações detalhadas sobre a sintaxe e utilização do comando $CONTROLE e suas opções.
As opções especificadas poderão ter validade para todos os componentes de um Banco de Dados, desde que declaradas imediatamente após o comando **$CONTROLE**. Caso você deseje que as opções especificadas se apliquem apenas a determinados arquivos, basta que as mesmas sejam declaradas nos arquivos onde se deseja que as opções atuem. Vale lembrar que só faz sentido usar as opções de controle nos arquivos apenas para desligar opções ligadas no comando **$CONTROLE**.

Exemplos:

```
<<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        e
        bc (0)          u05
        ba (ae)         u03
```

Observação:

A opção de controle $**SEMINTEGR** do exemplo acima somente deve ser utilizada quando o usuário tiver certeza do que está fazendo. Utilizada de forma indevida, esta opção poderá acarretar sérios problemas de INTEGRIDADE no BANCO DE DADOS. Por exemplo, nas declarações do exemplo acima, um registro do arquivo "ae" poderá ser excluído no banco "a" mesmo havendo registros no arquivo "be" do banco "b" ligados a "ae", o que viola as regras básicas de integridade referencial no banco "b".

4.2.1.3 Declaração do Banco

A declaração do Banco de Dados e feita através de um conjunto de comandos especificados, de forma ordenada, dentro do esquema conforma a seguinte Sintaxe

```
BANCO [<percurso>] <nome_bd> <cod_seg> ...
[{ARQRECUP / DIARIO / DIAREC / AUTOREC}] ...
[{BLOQARQ / BLOQCHA / BLOQPAG / BLOQREG}] ...
[ESQUEMA = [<percurso_orig>]<nome_orig> ...
<seg_origem> ...
[<nivel_origem>]]
[DISTRIB = [<servidor>]
```

A seguir descrevemos os comandos utilizados na declaração de um Banco de Dados ***Open*BASE** assim como as suas respectivas opções. Veja o manual específico do sistema DEFINE para maiores detalhes.

Comandos e opções	Explicação
BANCO	Marca o início da declaração de um Dicionário de Dados ***OpenBASE***
<percurso>	Indica o diretório completo onde será criado o Dicionário de Dados, sendo o valor default **/usr/tsgbd/tsdic**). No ambiente **Windows** o nome completo do percurso inclui 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 abertura e utilização de um Banco de Dados é aceita a letra do Drive conforme especificado nas informações de Configuração do Registry • Pode ser utilizado o utilitário **Wincnfg.exe** (versão Windows do utilitário **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 migrado de Drive deve ser executado o comando define para inserir a informação do novo Drive no dicionário
<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 branco • deve ser único no diretório
<cod_Seg>	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
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)
DIÁRIO	Determina que seja utilizado o arquivo DIÁRIO para armazenar todos os dados das transações **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

Comandos e opções	Explicação
	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.
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
BLOQCHA	Determina para todos os processos de atualização do Banco de Dados que a estratégia de bloqueio será baseada em bloqueio de chaves
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 subesquema, ou seja, a estrutura parcial de um Banco de Dados já existente. Esta cláusula determina que na compilação do subesquema, seja aberto o Banco de Dados que dará origem ao subesquema (BDO), com a finalidade de consistir as estruturas definidas em ambos. Assim sendo: ▪ Todos os arquivos declarados no subesquema devem existir no BDO ▪ Todos os itens declarados em um arquivo no subesquema 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 subesquema deve contemplar todos os itens chaves do arquivo que esta sendo referenciado no BDO, na ordem em que foram declarados no BDO No caso de subesquema não é necessário declarar as cláusulas de controle (ARQRECUP, DIÁRIO, BLOQARQ / BLOQCHA / BLOQPAG / BLOQREG), em função de todos os controle estarem subordinados ao Banco de Dados ao qual o Subesquema se referencia
<percurso_orig>	Percurso onde se encontra o Dicionário de Dados do Banco de Dados que será referenciado no Subesquema
<nome_orig>	Nome do Dicionário de Dados do Banco de Dados que será referenciado no Subesquema
<seg_origem>	Código de segurança do Banco de Dados que será referenciado no Subesquema
<nivel_origem>	Palavra de nível do Banco de Dados que será referenciado no Subesquema
DISTRIB=	A opção **DISTRIB** indica que o banco de dados é distribuído. O nome do Host onde residem o Banco de Dados e seu servidor mestre está especificado pelo operando **<servidor>**

Exemplos:

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

Em caso de Bancos Distribuídos:

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

4.2.1.4 Declaração dos Níveis de Acesso

Em um Banco de Dados OpenBASE são estabelecidos níveis de acesso às informações para os diversos perfis de usuários. Os níveis de acesso especificados podem ser proporcionais aos perfis funcionais dos usuários que manipulam as informações dentro de uma organização qualquer.

Sintaxe

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

O comando **NIVEIS** define os níveis de segurança em termos de privacidade. O ***OpenBASE*** possibilita a atribuição de níveis de leitura e gravação, para cada item de dados. Cada nível de privacidade é definido associando-se um **número de nível** com sua respectiva **palavra de nível**. Os números de níveis são representados por inteiros de 1 a 15 e devem ser especificados em ordem crescente na codificação do Esquema do Banco de Dados. As palavras de níveis são constituídas por ate 6 caracteres, sendo o primeiro caractere 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.

Exemplo:

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

4.2.1.5 Declaração dos Arquivos de Dados

Após declarar o Banco e seus atributos, dentro do esquema, pode ser feita a declaração dos arquivos, tabelas ou relações de cada Banco de Dados conforme a seguinte Sintaxe

[RELACOES:]
NOME: [<percurso>] <nome_arquivo> <tipo_arquivo>...
[ESQUEMA = [<*percurso_orig*>] <*nome_orig*>...
<*seg_origem*>...
[<*nivel_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]
...
...

A seguir descrevemos os comandos utilizados na declaração dos arquivos de dados em Bancos de Dados ***Open*BASE** assim como as suas respectivas opções. Veja o manual específico do sistema DEFINE para maiores detalhes.

Comandos e opções	Explicação
RELAÇÕES:	Para documentar o início das declarações dos arquivos de dados de um Banco, utiliza-se, opcionalmente, a declaração **RELAÇÕES:**
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 até 10 caracteres com as seguintes restrições: ▪ pode conter letras, números, ponto (.) e sublinhado (_) ▪ tem que iniciar por letra ▪ não pode ter branco ▪ deve ser único no diretório No MS-DOS, <nome_arquivo> não pode ultrapassar 8 caracteres
<tipo_arquivo>	Especifica o tipo do arquivo de dados, designado por uma letra. **Tipos Básicos: 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 **Observações**

Comandos e opções	Explicação
	Arquivos do tipo 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, também podem participar de Subesquemas. 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 Subesquema, 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, ou seja, 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_orig>	Percurso onde se encontra o Dicionário de Dados do Banco de Dados ao qual o arquivo pertence efetivamente
<nome_orig>	Nome do Dicionário de Dados do Banco de Dados ao qual o arquivo pertence efetivamente
<seg_origem>	Código de segurança do Banco de Dados ao qual o arquivo pertence efetivamente
<nivel_origem>	Palavra de nível do Banco de Dados ao qual o arquivo pertence efetivamente

Exemplos:

<< 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 >>

4.2.1.6 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. Veja a sintaxe do comando:

[REGISTRO:]
<nome_item> [{[<rep>]/(<ligações>)/(<caminho>)/(0)}]
... <tipo>:<tamanho>[,<num_decimais>]
... [(<num_nivel_ler>,<num_nivel_grav>)]
... [{POS <nome_item_rd>[+<deslocamen>] |
... VIRTUAL (<nome_item_part>, ..., <nome_item_part>)}]
... [UNICA]

A seguir descrevemos os comandos utilizados na declaração de um Banco de Dados ***Open*BASE** assim como as suas respectivas opções. Veja o manual específico do sistema DEFINE para maiores detalhes.

Comandos e opções	Explicação
REGISTRO:	Opcionalmente, para documentar o inicio das declarações dos itens de dados que compõem o registro de um arquivo de dados, utiliza-se a palavra REGISTRO seguida de dois pontos (:)
<nome_item>	**<nome_item>** especifica o nome dos itens que compõem cada registro de um arquivo de dados o qual compõe-se de uma cadeia com até 63 caracteres com as seguintes restrições: • pode conter letras, números, ponto (.) e sublinhado (_) • tem que iniciar por letra • não pode ter branco • deve ser único no diretório
[<rep>]	É o fator de repetição de um item. A LDBD não permite que sejam implementados itens multivalorados. Para tanto, o usuário pode definir que um item seja repetido tantas vezes

Comandos e opções	Explicação
	quanto for determinado em **<rep>,** que consiste de um valor numérico inteiro contido entre colchetes. Este recurso quando utilizado faz com que o DEFINE associe aos nomes dos 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 é referenciada como chave estrangeira em outros arquivos do Banco de Dados. Assim sendo, **<ligações>** é 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** • 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 à 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 à 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 ***OpenBASE*** 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 se esta chave for a primeira 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>	Define o tipo e o tamanho do Item, assim como o número de casas decimais no caso do item

Comandos e opções	Explicação
[,<num_decimais>]	suportar valores numéricos. **Tipo do Item** O tipo do item a ser declarado em **<tipo>** poderá ser especificado como: • **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:
<tamanho>	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
<num_decimais>	Se o item suportar valores numéricos é possuir casas decimais, o número de casas decimais é 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 ***OpenBASE*** 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
POS <nome_item_rd> [+ <deslocamen>]	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 à posição inicial do item a ser

Comandos e opções	Explicação
	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 à 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 têm 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

4.2.1.6.1 Observações sobre o <tipo> de itens

A função OPUS "DTOC", para itens do tipo D2, retorna uma cadeia de caracteres no formato **dd/mm/aa** se foi especificado **set century off** e **1901 <= ano <= 1999**. Caso contrário, a função retorna uma cadeia no formato **dd/mm/aaaa**. Para itens do tipo D4, retorna uma cadeia no formato **dd/mm/aaaa**, mesmo para datas menores do que 1901 e datas maiores do que 1999. A função **"CTOD" c**onsidera uma cadeia no formato **dd/mm/aa** como **dd/mm/19aa**.

Quanto à conversão de variáveis para itens do banco, observe o seguinte:

Se tipo D2, 01/01/1901 <= data <= 04/06/2080, caso contrário é emitida a mensagem OPUS (varite) => Estouro na conversão numérica.

Quanto à conversão do tipo do item D2 para D4 observe o seguinte:

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.

4.2.1.6.2 Observações sobre o <tamanho> dos itens

Para melhor compreensão do **<tamanho>** do item, consulte a Tabela a seguir.

Tipo item	Tamanho a declarar	Tamanho em Bytes(tb)	Valores Suportados	Tipo Interno
U	1 a 2048	1 a 2048	Cadeia de caracteres	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 01/01/1901 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

O valor de retorno do item é mascarado em função do seu tipo, conforme tabela abaixo.

Tipo dos itens	Máscaras
U	Espaços em branco
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)

Exemplos:

```
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
```

4.2.1.7 Processamento do esquema

O comando DEFINE aciona o compilador de Esquemas e Subesquemas que darão origem efetivamente à criação do Banco de Dados conforme a seguinte Sintaxe

DEFINE [-<opcao>] <nome_Esquema>, onde:

A seguir descrevemos o comando DEFINE assim como as suas respectivas opções. Veja o manual específico do sistema DEFINE para maiores detalhes.

parâmetros	Explicação do parâmetro
-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>	O "DEFINE" cria um "pipe" para o programa **<nome do programa>**
-I	direciona a listagem do Esquema para o SPOOL
<nome_Esquema>	é o nome do arquivo fonte que contém o Esquema ou Subesquema 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

4.3 Administração dos ambientes *Open*BASE

Os ambientes ***Open*BASE** oferecem uma grande variedade de facilidades para auxiliar os usuários na execução das tarefas administrativas inerentes aos Bancos de Dados.
Esse conjunto integrado de recursos ***Open*BASE** recebe o nome de **Sistema de Utilitários de Bancos de Dados (SUBD)** apresentando programas voltados para a manutenção e administração de Banco de Dados, permitindo, por exemplo:

- configurar os ambientes ***OpenBASE***
- descarregar e recarregar um ou todos os arquivos de um Banco de Dados
- adicionar informações aos arquivos de um Banco de Dados
- desfazer transações não completadas

- fornecer informações sobre os recursos utilizados pelos Bancos de Dados
- verificar a consistência dos Banco de Dados
- converter Bancos de Dados externos para o formato ***OpenBASE***
- efetuar backup on-line com replicação de Bases de Dados

Este capítulo focaliza os utilitários, principalmente do ponto de vista das plataformas UNIX, cujos nomes são precedidos do prefixo BD (Banco de Dados). Na versão Windows, os nomes dos utilitários ***OpenBASE*** são prefixados por WIN em vez de BD.
Não é recomendável que os usuários acessem Bancos de Dados ***OpenBASE*** utilizando outros meios diferentes das interfaces e utilitários oferecidos pelo ***OpenBASE***, pois estes garantem a consistência e integridade das informações. As maneiras de invocar os utilitários ***OpenBASE*** são:

- através dos menus do programa **OPENB** (aplicação não gráfica**)**
- através dos menus do programa **Winutil** (aplicação gráfica**)**
- a partir da linha de comando (**prompt**) do sistema operacional

A seguir apresentamos uma lista dos utilitários ***OpenBASE*** e suas respectivas funções globais.
Para obter maiores detalhes consulte os manuais específicos na .

Utilitário	**Descrição global**
BDADIC	**Adiciona registros a arquivos de Bancos de Dados.**
BDCDBF	**Converte arquivos dbf para a estrutura dos Bancos *Open*BASE.**
BDCNFG	**Muda configuração de ambiente.**
BDDELE	**Elimina Bancos de Dados *Open*BASE.**
BDDESC	**Descarrega Banco de Dados *Open*BASE.**
BDESPA	**Lista relatórios estatísticos dos arquivos de Bancos de Dados.**
BDINDC	**Otimiza a criação de arquivos de índice.**
BDINDV	**Cria uma chave virtual e seu arquivo de índice.**
BDLICE	**Lista tabela de usuários e bloqueios.**
BDCODI	**Lista informações sobre a cópia OpenBASE instalada**
BDLIDI	**Lista transações do diário.**
BDMENS	**Lista mensagens de erro do ambiente *Open*BASE.**
BDOTIM	**Otimiza arquivos de índice.**
BDRECA	**Recarrega Bancos de Dados.**
BDRECU	**Recupera Bancos de Dados.**
BDREDI	**Refaz ou desfaz transações em Banco de Dados.**
BDRMCE	**Elimina pontos de controle (semáforos) em situações de contingência.**
BDSERV	**Servidor de Bancos de Dados *OpenBASE*.**
BDSGBD	**Gerenciador de bloqueios de Bancos de Dados *Open*BASE.**
BDTETE	**Lista tabela TERMINFO.**
BDVERI	**Verifica integridade dos Banco de Dados.**
BDVESQ	**Recupera o esquema de um Banco de Dados a partir do seu dicionário.**
BDVESCLI	**Recupera (recria) o esquema de um Banco de Dados Distribuído.**
BDSREP	**Implementa servidores de replicação de Bancos de Dados *Open*BASE.**
BDCREP	**Implementa clientes de replicação de Bancos de Dados *Open*BASE.**
BDTSQL	**Cria esquema SQL a partir do dicionário de Bancos de Dados OpenBase.**
OPENB	**Apresenta em menus os serviços do ambiente *Open*BASE.**

Para obter maiores detalhes destes utilitários assim como as informações referentes a todos os utilitários consulte os manuais específicos ("Utilitários do OpenBASE") na página internet do OpenBASE.

4.4 Sub-rotinas *OpenBASE*

Neste capítulo será apresentada a relação das sub-rotinas básicas dos produtos ***Open*BASE** que permitem acessar Bancos de dados ***OpenBASE a partir de*** quaisquer linguagens de alto nível. A seguir apresentamos uma lista contendo algumas dessas sub-rotinas e a descrição global:

Sub-routina	**Descrição global**
BDABRE	**abre Banco de Dados**
BDACHC	**acessa primeiro registro para leitura em cadeia**

BDACHP	**acessa primeiro registro para leitura por prefixo**
BDALTE	**altera itens não chave**
BDBLOQ	**bloqueia banco de dados**
BDDESB	**desbloqueia banco de dados**
BDDESF	**desfaz transação corrente**
BDESCH	**seleciona chave de ordem para leitura seqüencial**
BDESVA	**esvazia arquivo**
BDEXCL	**remove registros de arquivos**
BDFECH	**fecha arquivo do banco de dados**
BDINCL	**inclui registros**
BDINCA	**inclui registros no meio de uma cadeia**
BDJUNT	**faz junção no banco de dados**
BDPEGC	**faz leitura em cadeia**
BDPEGD	**faz leitura direta**
BDPEGI	**faz leitura seqüencial invertida**
BDPEGM	**faz leitura por chave**
BDPEGP	**faz leitura por prefixo**
BDPEGS	**faz leitura seqüencial**
BDPEGT	**faz leitura invertida em cadeia**
BDPEGV	**faz leitura invertida por prefixo**
BDPOSI	**posiciona em registro para leitura seqüencial**
BDREST	**restaura a tabela de execução de um arquivo**
BDSALV	**salva tabela de execução de um arquivo**
BDTROC	**altera valores de itens que não são chaves primárias**

Para obter maiores detalhes destas rotinas assim como as informações referentes a todos os utilitários consulte os manuais específicos ("Utilitários do OpenBASE") na página internet do OpenBASE.

4.5 Consulta e Atualização dos Bancos de Dados

O ambiente ***Open*BASE** disponibiliza um sistema Interativo de Consulta e Atualização de Bancos de Dados chamado **GERAL** (***Ger***ador de ***A***plicações on-***l***ine), o qual oferece uma interface interativa e amigável para consultas e atualizações em Bancos de Dados. Adicionalmente, o sistema **GERAL** oferece os seguintes recursos: apresentação de menus e opções, gerador de relatórios, execução de consultas e atualizações em modo interativo e batch.

Em resumo, o sistema **GERAL** oferece os seguintes recursos:

- Processador de telas
- Gerador de relatórios
- Acionador de procedimentos e rotinas
- Processador de cálculos
- Processador de registros
- Processador de programas executáveis
- Processador de comandos do sistema operacional

Para invocar o GERAL, via terminal, digita-se:

Geral [<arquivo>]

O GERAL responde com:

COMANDO ?

Neste momento, o usuário pode entrar com os comandos. As entradas para o GERAL podem conter qualquer numero de linhas, até 700 caracteres. Em caso de necessidade, o GERAL abre um prompt com o sinal de interrogação na primeira coluna da linha seguinte ao comando digitado. Todas as entradas de comandos ou dados para o GERAL, são terminados com a tecla {ENTER}.

O GERAL permite que o usuário trabalhe na modalidade Batch, ou seja, os comandos para o GERAL podem ser especificados dentro de um arquivo fonte, criado com um editor de textos.

Neste caso, o GERAL será invocado com o comando:

GERAL <nome-do-arquivo>

Onde: **<nome-do-arquivo>** é o nome do arquivo com os comandos do GERAL.

Por exemplo:

GERAL commands.p

O arquivo de comandos do GERAL deve conter apenas comandos válidos, um em cada linha. Caso contrário, será enviada mensagem de erro e a execução será cancelada.

A seguir apresentamos uma lista dos comandos do sistema GERAL com uma descrição geral:

comando	**descrição global**
ACEITE	**Invoca processador de telas ou retornar para a tela que invocou um processo**
ALTERE	**Substituir valores dos itens de um arquivo.**
BANCO	**Informar ao GERAL o Banco de Dados que será utilizado.**
CALCULE	**Calcular novos valores para itens numéricos no arquivo de seleção.**
DESLIGUE	**Desligar opções do GERAL.**
ENTRE	**Atribuir valor digitado pelo usuário a uma variável de memória.**
EXCLUA	**Remover registros de um arquivo do banco de dados.**
EXECUTE	**Executar um programa externo ao GERAL.**
EXIBA	**Exibir texto no dispositivo de saída.**
EXPLIQUE	**Exibir explicações sobre os comandos do GERAL.**
GERCLI	**Conectar um servidor de Banco de Dados.**
IMPRIMA	**Imprimir relatório formatado no dispositivo lógico associado.**
INCLUA	**Acrescentar registros em um arquivo do banco de dados.**
INFORME	**Exibir a estrutura do banco de dados utilizado.**
LIGUE	**Ligar opções do GERAL.**
LISTE	**Listar registros selecionados.**
PROCEDA	**Executar arquivos com comandos do GERAL.**
SAIA	**Terminar a execução do GERAL, retornando o controle ao sistema operacional.**
SELECAO	**Armazenar o endereço dos registros selecionados em arquivo para uso posterior.**
SELECIONE	**Selecionar registros de um arquivo do banco de dados.**
UNIDADE	**Trocar o dispositivo lógico para exibição das informações.**

Para obter maiores detalhes consulte os manuais específicos na página internet do OpenBASE.

4.6 Interfaces para ferramentas RAD

O OpenBASE disponibiliza as funções e rotinas de manipulação de Bancos de Dos para serem utilizadas pelas linguagens Visuais Basic, Delphi ou outras ferramentas RAD, podendo, assim, ser elaborados aplicativos para acessar Bases de Dados OpenBASE utilizando variadas ferramentas de desenvolvimento.
O acesso aos Bancos de Dados OpenBASE pode ser feito através de bibliotecas dinâmicas (DLLs do OpenBASE), através dos recursos ODBC (Driver ODBC do OpenBASE) e utilizando interfaces COM e DCOM para acessar objetos e métodos OpenBASE. Os componentes OpenBASE podem ser disponibilizados para aplicativos desenvolvidos em diversos ambientes de programação, por exemplo:

- Visual Basic (VB)
- Vsual Basic for Applications (VBA)
- Delphi
- PowerBuilder
- Windows Scripting Host (WSH)
- Active Server Pages (ASP)
- PHP: Hypertext Preprocessor (PHP)
- JAVA
- TCL/TK
- Perl
- Python

As ferramentas RAD (Rapid Application Development) possuem, geralmente, as seguintes características:

- São linguagens de quarta geração (4GLs)
- Utilizam modernas interfaces gráficas (GUIs)
- Estão integradas com arquiteturas OOP (Object-Oriented Programming)
- Implementam funções de acesso a Bancos de Dados
- Suportam, em plataformas Windows, acesso a Bancos de Dados utilizando ODBC, padrão Microsoft para Open DataBase Connectivity
- Invocam funções exportadas a partir de DLLs (Dynamic Link Libraries)
- Usam objetos padrão COM (Common Object Model), da Microsoft

As funções e objetos OpenBASE estão agrupados em duas bibliotecas de ligação dinâmica (DLLs), conforme a seguir:

ROTWIN32.DL Biblioteca principal com as rotinas básicas de acesso ao Banco de Dados OpenBASE.

CLIWIN32.DLL Similar a biblioteca ROTWIN32.DLL, porém as rotinas são voltadas para aplicações cliente dentro de uma arquitetura CLIENTE/ SERVIDOR em redes TCP-IP.

O módulo CLIWIN32.DLL inclui as funções para arquitetura cliente/servidor e permite desenvolver aplicativos cliente cujos servidores podem residir em qualquer outra plataforma, local ou remota, conforme veremos nos exemplos apresentados. As funções (ou rotinas) e métodos incluídos nessas bibliotecas dinâmicas permitem acessar e controlar Bancos de Dados OpenBase. Podemos agrupar todas as funções disponíveis em categorias, conforme a seguir:

Categoria das funções	Nome das funções
Leitura Seqüencial	ReiniciaCadeia LeProximo[Registro]Sequencial Le[Registro]AnteriorSequencial EscolheChave ObtemRegistrosNoArquivo
Leitura por Cadeia	IniciaCadeia LeProximo[Registro]Cadeia Le[Registro]AnteriorCadeia PosicionaNoRegistroPorChave ObtemRegistrosNaCadeia
Leitura por Prefixo	IniciaPorPrefixo LeProximo[Registro]PorPrefixo Le[Registro]AnteriorPorPrefixo
Leitura por Chave Primária	Le[Registro]PorChavePrimaria
Leitura por Endereço	Le[Registro]PorEndereco PosicionaNoRegistro ObtemEnderecoAtual
Inclusão, Exclusão e Alteração de Registro	ExcluiRegistro ExcluiRegistroCascata ExcluiRegistroPoeNulo Inclui[Todo]Registro Inclui[Todo]RegistroNaCadeia Altera[Todo]Registro Altera[Todo]RegistroCascata Altera[Todo]RegistroPoeNulo
Conexão e Controle de Bancos	AbreBancoDeDados FechaBancoDeDados JuntaBancoDeDados IniciaServidor FinalizaServidor
Manipulação de Memos	PegaGravaItemMemo LePoeItemMemo ExcluiItemMemo ObtemPercursoItemMemo ObtemTamanhoMemo
Itens especiais	PoeItem PegaItem
Rotinas Genéricas	Bloqueia Desbloqueia SalvaTabelaExecucao RestauraTabelaExecucao IniciaTransacao FinalizaTransacao DesfazTransacao LigaOpcao DesligaOpcao EsvaziaArquivo Crypt
Rotinas para Obter Informações	ObtemItensDoArquivo ObtemInfoSobreItem

Categoria das funções	Nome das funções
	ObtemQtdChaves ObtemChaves ObtemNumeroDoItem ObtemInfoSobreArquivo ObtemQtdLigacoes ObtemLigacoes ObtemNumeroDoArquivo ObtemQtdDeJuncoes ObtemJuncoes ObtemQtdVirtuais ObtemVirtuais ObtemTipoDaChave ObtemQtdItensBasicos ObtemItensBasicos ObtemQtdRedefinicoes ObtemRedefinicoes ObtemNiveis ObtemInfoSobreBanco ObtemInfoSobreCadeia ObtemDiretorio ObtemCliente ObtemMensagem

Neste manual **não apresentamos** em detalhe as funções e métodos incluídos nas DLLs nem os exemplos de aplicativos desenvolvidos em ambientes RAD tais como Visual Basic e Delphi.
Para obter maiores detalhes consulte os manuais específicos na página internet do OpenBASE.

4.7 Desenvolvendo aplicativos Internet - OpusWeb

O ***Open*BASE** incorporou às linguagens OPUS e OPUSWin algumas ferramentas de programação Internet, Intranet e Extranet, que facilitam o desenvolvimento de aplicativos em servidores WEB. Esses comandos e funções constituem uma extensão das linguagens OPUS e OpusWin e recebem o nome de OpusWEB e estão disponíveis para as seguintes plataformas e ambientes operacionais:

- **UNIX** (interface **CGI**)
- **Windows 9X/NT/2000** (interface **ISAPI**)

As linguagens **OPUS e OPUSWin** oferecem recursos para a **construção dinâmica** de aplicativos Cliente/Servidor em ambientes Internet e Intranet, utilizando as interfaces **CGI** (Common Gateway Interface) e **ISAPI** (Internet Server Application Program Interface), padrões amplamente utilizado nos servidores WEB.

4.7.1 Conceitos básicos

Ao desenvolver aplicativos **ISAPI** ou **CGI**, é necessário compreender claramente os mecanismos de funcionamento dessas interfaces de programação. Todos os problemas encontrados na utilização da OPUSWEB se relacionam, direta ou indiretamente, com o desconhecimento dos padrões e mecanismos CGI e/ou ISAPI.
Por isso, a compreensão destes assuntos constitui requisito básico para desenvolver **scripts**, **programas**, **extensões** ou **filtros** para servidores Web.
Abordamos, inicialmente, alguma idéias básicas a respeito dos seguintes assuntos:

a) O **modelo Internet** (cliente/servidor).
b) O **cliente** (“Browser”, “User Agent”)
c) O **servidor** (Web Server / HTTP Server)
d) O **protocolo** (HTTP - Hypertext transfer protocol)

4.7.1.1 O modelo Web cliente/servidor

Os serviços da plataforma Web estão baseados no modelo cliente/servidor, que permite distribuir e compartilhar os seus componentes básicos, ou seja, a **interface** com o usuário, a **lógica** dos programas e os **dados**. No modelo cliente/servidor, o **cliente**:

- Pode requerer dados do servidor
- Pode enviar dados para o servidor

- Pode solicitar do servidor a execução de processos
- Pode executar processos

No modelo cliente/servidor, o **servidor**:

- Pode enviar dados ao cliente
- Providencia o acesso a bases de dados
- Executa processos

Nos modelos tradicionais, os clientes e os servidores são classificados de “magros” ou “gordos”. Estes termos indicam mais uma relação funcional do que características físicas. Trata-se de uma divisão do trabalho ditada pelo perfil do próprio aplicativo. Por exemplo, é bastante freqüente que o servidor seja otimizado para fornecer dados a múltiplos clientes e, usualmente, a aplicação cliente é otimizada para, apenas, interagir com o usuário final.
O servidor é “gordo” quando toda a lógica funcional reside nele. Este é, ainda, o modelo mais comum na Web. O cliente “magro” é, usualmente, um Browser, que fornece apenas a interface com o usuário. Ou seja, as aplicações **CGI** ou **ISAPI** fornecem a lógica funcional dentro do servidor HTTP e o cliente **apenas exibe** os dados. Existem outros modelos **cliente / servidor**, com serviços distribuídos. Neste caso, as atividades são **compartilhadas** entre cliente e servidor.

4.7.1.2 O cliente (“User Agent”)

Os clientes Web são também chamados “User agents” ou, simplesmente, Browsers. No passado, por serem os Browsers meras interfaces com o usuário, o modelo era “servidor gordo, cliente magro”. Contudo, atualmente, existem tecnologias (por exemplo, Applets, Client-side Scripts, Style Sheets, plug-ins ...) que permitem maior grau de programação e processamento do lado do cliente.
A expressão cliente, ou “User Agent” se refere, usualmente, ao parceiro de uma sessão HTTP que inicia o pedido (“request”) a ser atendido (“response”) pelo servidor Web.
Na medida que a grande rede mundial (World Wide Web) cresce, em recursos e complexidade, novos tipos de “User Agents” são inventados para prover novas funcionalidades junto aos usuários.

4.7.1.3 O servidor (“Web Server”)

Servidores Web são processos que aceitam conexões (ou seja, sessões HTTP) solicitadas por clientes Web (Browsers) e, em resposta, fornecem informações na forma de mensagens e documentos de variados tipos, por exemplo, textos, imagens, som, vídeo ... etc ...
O desenvolvimento de servidores Web começou em 1990 e, atualmente, existem centenas de milhares de servidores Web na Internet, além de um número desconhecido de servidores utilizados nas Intranets corporativas.
Inicialmente, e durante algum tempo, apenas havia opções de servidores Web para plataformas **UNIX**. Atualmente, existem bons Servidores Web para plataformas Win32 e Mac.

4.7.1.4 O protocolo HTTP

Os servidores Web utilizam o protocolo HTTP (Hypertext Transfer Protocol) que foi implementado para permitir uma transferência rápida e eficiente de documentos na Internet”.
Consulte o manual específico da para obter maiores detalhes e aprender a programar OpusWEB.

4.7.1.5 Estrutura das transações HTTP

HTTP é um protocolo fácil de entender, pois é baseado no paradigma **pedido** e **resposta** (*request e response*). Uma transação HTTP, independente da sua complexidade, possui a seguinte estrutura elementar:

- O cliente abre uma conexão com o servidor, via TCP/IP, na porta 80.
- Estabelecida a conexão, o cliente envia um pedido ao servidor, na forma de mensagem.
- O servidor processa o pedido do cliente e responde, enviando de volta ao cliente a informação solicitada, em forma de documento padrão ou código de erro.
- O servidor fecha a conexão, terminando o processo ou “thread”.

O protocolo HTTP é “**stateless**” por natureza. Isto quer dizer que cada transação (pedido e resposta) é completada de forma independente, não sendo preservadas as informações de status referentes às transações anteriormente completadas. Isto exige um nível maior de controle e complexidade na programação de aplicativos Web.
Existem vários métodos e recursos que permitem administrar essa complexidade das transações HTTP.
Consulte o para maiores detalhes.

4.8 Ambiente e ferramentas SQL

A 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 ***OpenBASE***.
A linguagem do TSQL é totalmente compatível com a SQL padrão ANSI X3.135 - 1986, com as seguintes extensões apresentadas pelo ISO ANSI working draft (jan/1988): criação e eliminação dinâmica de Tabelas, Visões, Índices e

Privilégios, Comandos dinâmicos SQL, Comando LOCK TABLE, Dados do tipo DATA, Integridade Referencial, SQL embutida na linguagem OPUS, SQL embutida na Linguagem C, dentre outras.
Outros aspectos importantes da TSQL:

- Recompilação automática de programas SQL em função de eliminação de índices.
- Comandos SQL com Gerador de Relatórios e Gerador de Telas.

Para obter maiores detalhes consulte os manuais específicos na página internet do OpenBASE.

4.9 Linguagens de programação OPUS e OpusWin

O ***OpenBASE*** possui linguagens de programação de alto nível. A linguagem OPUS permite desenvolver aplicativos para plataformas UNIX e WIN32 utilizando a interface caractere ou Console Application. No que diz respeito à manipulação de Bancos de Dados, a OPUS oferece ferramentas, comandos e funções poderosos e de grande performance pois está plenamente integrada ao ambiente e estrutura do ***OpenBASE***. Além disso, a linguagem OPUS permite acessar qualquer tipo de Banco de Dados utilizando o ODBC e conjugando, desta forma, os recursos da SQL.
Um programa escrito em **OPUS**, quando submetido ao processo de pre-compilação, é traduzido para linguagem C/C++, para então ser efetivamente processado pelo compilador C/C++. Isto quer dizer que todo programa escrito na linguagem OPUS precisa de um compilador C/C++ para tornar-se uma função, uma procedure, um programa executável. Consulte o para maiores detalhes.
Um programa escrito em **OpusWin**, quando submetido ao processo de pre-compilação, é traduzido para linguagem C/C++, para então ser efetivamente processado pelo compilador C/C++. Isto quer dizer que todo programa escrito na linguagem OPUS precisa de um compilador C/C++ para tornar-se uma função, uma procedure, um programa executável (.EXE) ou uma DLL (.DLL). Veja como construir DLLs utilizando a OpusWin consultando o .
O processo de gerar programas C/C++ oferece total portabilidade e alta performance às aplicações, além de permitir que o usuário possa agregar ao programa (no mesmo fonte e sem restrições), comandos da linguagem C/C++.

4.9.1 A linguagem OPUS

A OPUS é uma linguagem de alto nível que permite desenvolver aplicativos em múltiplas plataformas e arquiteturas.
Para obter maiores detalhes consulte os manuais específicos na página internet do OpenBASE.

4.9.2 A linguagem OpusWin

Os programas escritos na linguagem OpusWin são compatíveis com os programas desenvolvidos na tradicional OPUS, do OpenBASE e aderem plenamente aos padrões WIN32, possuindo, em conseqüência, todas as suas características e vantagens.
Para obter maiores detalhes consulte os manuais específicos na página internet do OpenBASE.

4.10 Impressão em ambientes Windows – OPUSRel

4.11 Interface visual de programação - VisualOpus

A Visual Opus é uma ferramenta projetada para desenvolvimento de sistemas, implementando os recursos e facilidades das plataformas Windows 9X/NT/2000/XP e visando maior produtividade para usuários dos produtos OpenBASE.
A Visual Opus não é uma linguagem de programação e sim uma interface gráfica que permite a elaboração e/ou modificação de programas OpusWin, utilizando recursos visuais padrão, de forma interativa. A linguagem de programação utilizada pela Visual Opus é a OpusWin, assim como, por exemplo, as linguagens utilizadas pelos ambientes Delphi e Visual Basic são, respectivamente, o Pascal e o Basic.
Os módulos (programas e procedures) produzidos ou manipulados através da utilização da ferramenta Visual Opus podem ser incorporados ao ambiente de produtos da família OpenBASE. Desta forma, a Visual Opus constitui, sem dúvida, um poderoso recurso a ser utilizado no processo de desenvolvimento de aplicações. Os objetivos básicos da Visual Opus são os seguintes:

- produzir código fonte OpusWin (programas ou procedures) utilizando interface visual e ferramentas interativas, não havendo necessidade de escrever uma só linha de código no que se refere ao tratamento das janelas de diálogo e dos controles nelas incluídos
- incorporar programas ou procedures OpusWin já existentes, produzidos ou não através da interface Visual Opus, ao ambiente interativo desta, permitindo a modificação desses módulos fonte

Para obter maiores detalhes consulte os manuais específicos na página internet do OpenBASE.

4.12 Ambiente integrado de desenvolvimento – OpenIDE

O OpenIDE oferece um ambiente integrado de desenvolvimento de aplicativos. Neste tópico apresentamos a estrutura e os conceitos básicos deste recurso OpenBASE.

Uma aplicação desenvolvida nas linguagem OPUS ou OpusWin resulta sempre em um programa principal e, normalmente, em um conjunto de rotinas ou funções. Os programas encontram-se inc;uídos em arquivos segundo critérios definidos pelos autores do projeto de um determinado sistema aplicativo.
O arquivo que contem o programa principal é chamado de arquivo principal e, geralmente, contém também as rotinas e funções utilizadas pelo programa principal. Os demais arquivos, se existirem, contém as rotinas e funções.
O código fonte das rotinas e funções especifica, através da opção $LIBRARY, as bibliotecas onde elas serão armazenadas depois de compiladas e ligadas. O programa principal, por outro lado, deve especificar quais as bibliotecas (opção $LIBRARY) que contém as rotinas e funções utilizadas.
As linguagens OPUS e OpusWin possuem a facilidade de incluir, dentro de um programa, função ou rotina, trechos de código contidos em arquivos chamados de inclusão (INCLUDE).
A compilação dos modolos fonte que compõem uma aplicação deve ser feita na seguinte ordem:

1. os arquivos que contém as rotina e funções, em qualquer ordem
2. o arquivo que contém o programa principal

Como conseqüência:

- uma modificação feita em um arquivo de rotina obriga a recompilação e ligação do arquivo rotina e do arquivo principal
- se um arquivo de inclusão é modificado todos os arquivos que o utilizam tem que ser compilados e ligados

4.12.1 Janelas do ambiente OpenIDE

O ambiente integrado OpenIDE é constituído por três janelas:

1. No alto, à esquerda, está a **janela do projeto**. Esta janela mostra os arquivos do projeto organizados em arvores hierárquicas.
2. No alto, à direita, se encontra a **janela de texto**, que exibe os programas do projeto
3. Embaixo está a **janela de mensagens**. Esta janela informa os procediemntos sendo executados assim como os avisos de erros encontrados na construção do projeto bem como os erros de compilação das rotinas.

O tamanho destas janelas pode ser alterado a qualquer momento utilizando o mouse, clicando e arrastando as barras de separação ou utilizando os botões apresentados na barra de ferramentas.

4.12.2 Projetos no ambiente OpenIDE

Os elementos de um determinado sistema aplicativo são agrupados em projetos, contituídos por:

- a lista dos arquivos da aplicação,
- a lista dos diretórios envolvidos com a aplicação e
- o conjunto de programas operando sobre estas listas com o objetivo de:

  - Exibir de forma amigável os arquivos da aplicação
  - Incluir e remover arquivos e diretórios nas listas do projeto
  - Localizar (pelo nome) e exibir as rotinas independente de qual o arquivo que as contém
  - Verificar e garantir a existência das bibliotecas e suas regras de utilização
  - Verificar e garantir a existência dos arquivos de inclusão referenciados no projeto
  - Executar corretamente as compilações e ligações que se fizerem necessárias em conseqüência de modificações nos elementos do projeto
  - Vefificar e garantir a existência do ambienbte de Bancos de Dados do apliacativo

4.12.2.1 Abertura de projetos no ambiente OpenIDE

Para criar novos projetos basta acionar o menu **Arquivos/Criar Projeto**. O OpenIDE mostrará uma caixa de dialogo onde deverá ser informado o nome completo (incluído o percurso) do projeto a ser criado. Isto feito, uma nova caixa de dialogo será exibida permitindo a especificação dos arquivos que farão parte do projeto. Se nada for informado o projeto será criado vazio, isto é, sem arquivos.
Para abrir um projeto existente basta acionar o menu **Projeto/Abrir Projeto**. O OpenIDE mostrará uma caixa de dialogo onde será informado o projeto que está sendo aberto.
Assim que o projeto for criado ou aberto ele será mostrado na árvore da **janela projeto**.

4.12.2.2 Estrutura dos projetos OpenIDE

Os elementos que compõem um determinado projeto OpenIDE são agrupados, organizados e apresentados utilizando árvores com a seguinte estrutura hierárquica:

- No **nível 0** da árvore está o nome do projeto
- No **nível 1** da árvore está a classificação dos arquivos: fontes, inclusões, bibliotecas e bancos de dados. Os diretórios e a documentação do projeto são exibidos no **nível 2**.
    - Os **níveis 2** de inclusões e documentação são os arquivos de inclusão e de documentação pertencentes ao projeto.
    - Os **níveis 2** de fontes separam o arquivo principal dos arquivos de rotinas e funções sendo exibidos no nível 3
    - Os **níveis 2** de bibliotecas separam as bibliotecas criadas das bibliotecas externas sendo exibidas no nível 3
    - O **nível 2** de banco exibe as informações sobre a localização dos esquemas, dicionários e arquivos dos bancos de dados utilizados pelo aplicativo
- O **nível 3** de Diretórios informa os diversos diretórios envolvidos no projeto, a saber:
    - diretório de localização do projeto
    - diretório de execução ou seja o diretório que conterá os resultados das compilações e ligações efetuadas pelo projeto.
    - diretórios de Inclusões
- No **nível 4** de Bibliotecas Externas são mostradas as rotinas que as compõem

Ao lado de cada item da árvore pode existir um pequeno quadrado contendo o símbolo mais (+) ou o símbolo menos(-). A sua inexistência informa que o item não possui subitens. Se estiver preenchido com (+), significa que existem subitens, porém não estão sendo mostrados. Para exibi-los basta clicar no pequeno quadrado. Se existir preenchido com (-),significa que existem subitens e estão mostrados. Para que os subitens deixem de ser vistos, basta clicar no pequeno quadrado.

4.12.2.3 Construção dos Projetos OpenIDE

A construção de um projeto é iniciada pelo menu **Projeto/Construir**. Construir ium projeto significa executar, na ordem correta, todas as compilações e ligações que se fizerem necessárias como conseqüência das modificações feitas nos arquivos após a ultima construção realizada. O arquivo principal é compilado se todos os arquivos de rotina foram corretamente compilados. O resultado da compilação do arquivo principal é um arquivo executável ou DLL.
O processo de construção pode ser acompanhado pelas mensagens mostradas na janela de mensagens. No caso de erros de compilação de um arquivo, um duplo clique na linha que o informa fará com que o arquivo seja colocado na janela de texto com a linha do erro selecionada.
A construção de um projeto pode ser interrompida pelo comando de menu **Projeto/Interromper Construção**. A construção será interrompida ao final da compilação que estiver sendo executada.

4.12.2.4 Manutenção dos projetos OpenIDE

Os projetos OpenIDE podem ser modificados a qualquer momento, podendo-se incluir ou remover os elementos que o compõem.

4.12.2.4.1 Reconstrução de projetos OpenIDE

Através do menu **Projeto/Reconstruir** todo um poderá ser reconstruído sendo recompilados todos os seus programas, rotinas e funções.

4.12.2.4.2 Inclusão e remoção de arquivos

Para inserir arquivos no projeto basta comandar o menu **Projeto/Inserir** ou utilizar a tecla Insert com a janela do projeto selecionada. A resposta a este comando é a exibição de uma caixa de dialogo aonde os arquivos a serem inseridos poderão ser selecionados. Para remover arquivos de um projeto basta seleciona-los na árvore e teclar a tecla Delete ou acionar o menu **Projeto/Remover**.

4.12.2.4.3 Exibição de arquivos e rotinas

Para exibir um arquivo do projeto basta dar um duplo clique em seu nome na arvore do projeto. O conteúdo desse arquivo será mostrado na janela texto.
Na janela ComboBox acionada através da barra de ferramentas estão todas as rotinas e funções pertencentes ao projeto. Selecione a região de edição da ComboBox, digite a inicial do nome da rotina a ser vista. Uma janela ListBox será aberta mostrando todas as rotinas ou funções cujos nomes iniciam com a letra inicial escolhida. Clique na rotina que deseja visualizar e o arquivo que a contém será colocado na janela texto.

4.12.2.4.4 Execução dos projetos OpenIDE

O comando de menu **Projeto/Executar** dispara, se necessário, a construção do projeto e em seguida se a construção foi bem sucedida comanda a execução do .exe gerado.

4.12.2.4.5 Compilação de módulos fonte

Para compilar um determinado módulo fonte de um projeto:

- Coloque o módulo fonte na janela texto.
- Acione o menu **Projeto/Compilar janela ativa**.

Os erros de compilação serão mostrados na janela de mensagens e um duplo clique na linha que o informa fará com que a linha do erro apareça selecionada na janela texto, aonde poderá ser feita a correção.

4.12.2.4.6 Formatação de arquivos fonte

Embelezar um arquivo é reformatá-lo utilizando uma indentação que realça os loops, os case etc... permitindo que a leitura do programa seja facilitada. Para embelezar um determinado arquivo, faça o seguinte:

- Coloque o arquivo na janela texto.
- Acione o menu **Projeto/Embelezar janela ativa**.

4.12.2.5 Verificação dos Projetos OpenIDE

A função de verificação do projeto, acionada através do menu **Projeto/Verificação**, analisa os componentes do projeta para detectar:

- Rotinas, Includes e Libraries chamados porém não incluídos no projeto
- Arquivos contendo rotinas sem especificação ou com especificação incorreta da Library
- Inexistência ou duplicidade de arquivos de programas

Todos os problemas detectados são informados na janela de mensagens. Um duplo clique na linha da de uma mensagem fará com que o arquivo nela citado seja visualizado na janela texto. Se a mensagem referencia uma rotina ou função, a linha que contém a rotina será selecionada.

4.13 Ambiente integrado DBA – OpenDBA

O Opendba é uma ferramenta de Administração de Dados cujo objetivo é auxiliar o DBA na execução de trabalhos de criação e manutenção de banco de dados. Você pode:

- Criar um banco de Dados Openbase.
- Criar um banco de Dados Openbase com acesso em SQL.
- Preparar um Banco de Dados Openbase para acesso SQL.
- Executar operações de manutenção de banco de dados, tais como:
- Carregar e descarregar Backup.
- Recuperar e desbloquear um banco de dados.
- Otimizar índices
- Verificar estado do banco de dados.

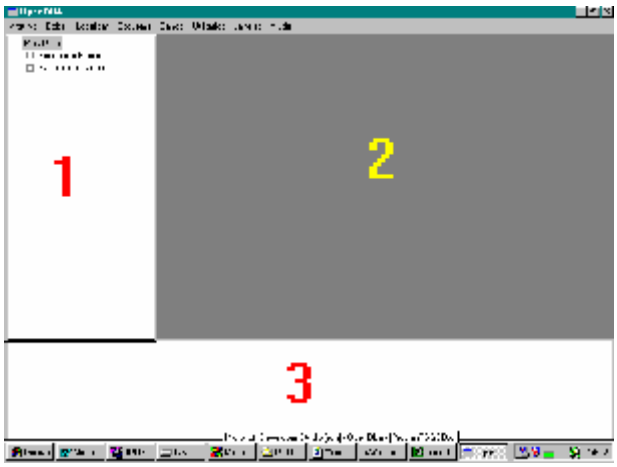
4.13.1 Modo de operação do Opendba

Existem três maneiras de operação no Opendba:

1. Mostrar só SQL: Trabalha com a interface totalmente voltada para o trabalho com banco de dados SQL.
2. Mostrar só Esquemas: Trabalha com a interface totalmente voltada para o trabalho com banco de dados OPENBASE (define).
3. Mostrar SQL e Esquemas: Trabalha com a interface voltada para o trabalho com banco de dados OPENBASE e/ou banco de dados OPENBASE convertidos para aceitar acesso SQL.

Para ativar o modo de operação selecione no menu: **Arquivo/Preferências**.

A tela do **Opendba** é dividida da seguinte maneira:



Onde:

1. Janela da arvore de banco de dados: É a janela mais a esquerda. Na árvore estão os bancos existentes, seus arquivos e itens componentes.
2. Janela de trabalho: Utilizada para trabalho com textos, definição de esquemas, etc.
3. Janela de Mensagens: Recebe as mensagens enviadas pelo programa na depuração de erros.

Todas as funções do Opendba aplicam-se no item que está selecionado na árvore.

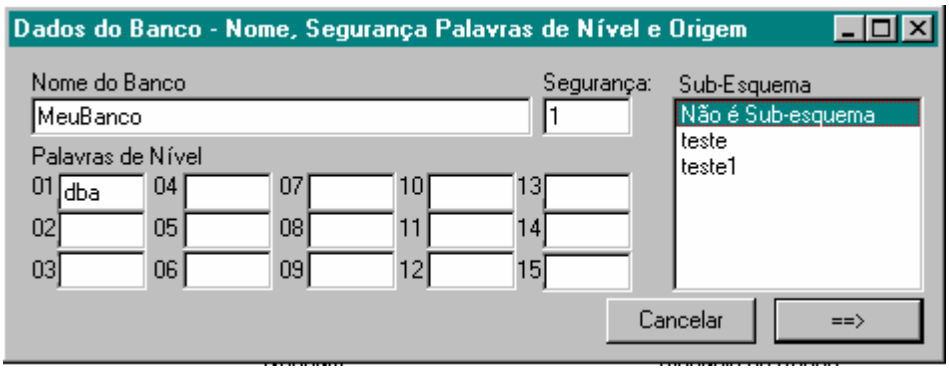
4.13.2 Como criar um banco de Dados Openbase

Você pode criar um banco Openbase se uma das opções do menu mostradas abaixo estiverem selecionadas:

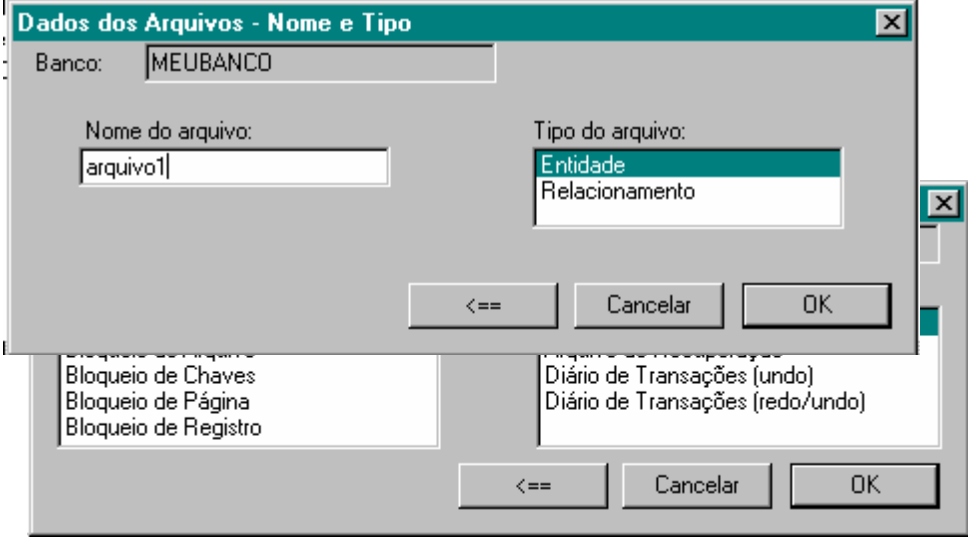
a) Arquivo/Preferências/Mostrar só Esquemas ou
b) Arquivo/Preferências/Mostrar SQL e Esquemas

Primeiramente você precisa de um esquema para compilar. Neste esquema estão definidos os atributos do banco, as tabelas e os itens que pertencem às mesmas.

- Posicione-se na árvore em **Esquemas Básicos**.
- Se você selecionou a opção **'a)'** acima, selecione no menu **Esquema/Criar objeto**. Senão, selecione **Esquema/Criar Esquema em Metadata**. Abre-se o diálogo para criação de esquemas. Você também pode apertar a tecla **INS** como atalho.
- Preencha os dados da primeira caixa de diálogo.

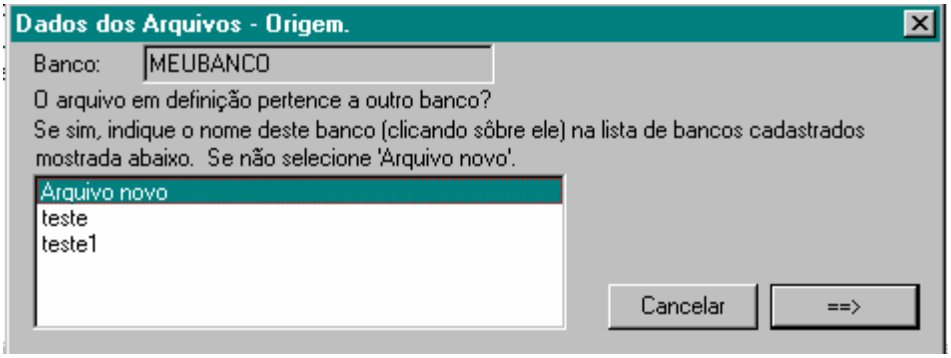


- Escolha o tipo de Bloqueio e tipo de Processo de Recuperação. Aparece então na janela de textos, o esquema básico, sem nenhum arquivo criado. Na árvore, o esquema recém-criado aparece e está selecionado. Agora, é preciso definir os arquivos que farão parte do esquema.



4.13.3 Como criar arquivos num banco de dados Openbase

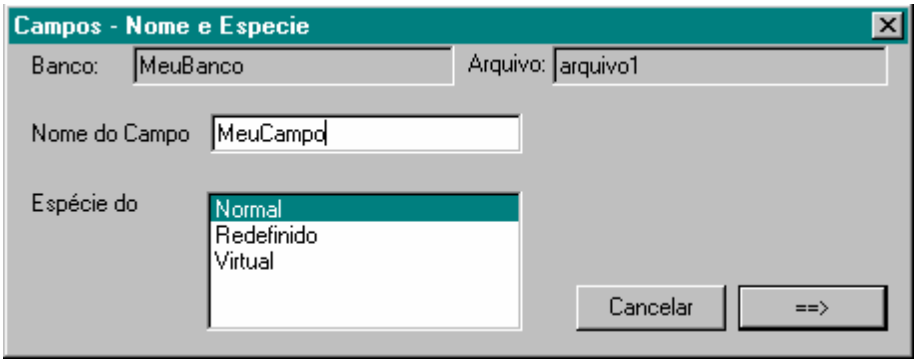
Mantenha o esquema selecionado na árvore, e ative no menu **Esquema/Inserir Arquivo em ...**, ou aperte a tecla **INS** . Aparece o diálogo para criação de arquivos.



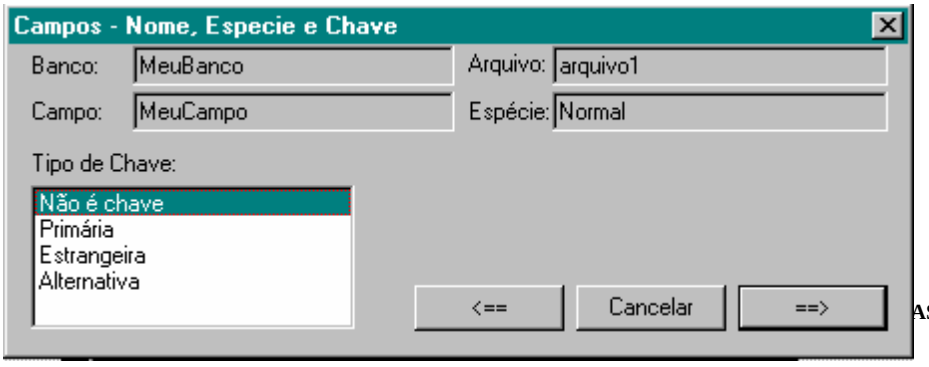
Leia as instruções com atenção. Depois de clicar na seta, digite o nome do arquivo e selecione o tipo. Para encerrar clique no botão '**OK**'. Repita este passo para cada arquivo que quiser inserir no banco de dados.

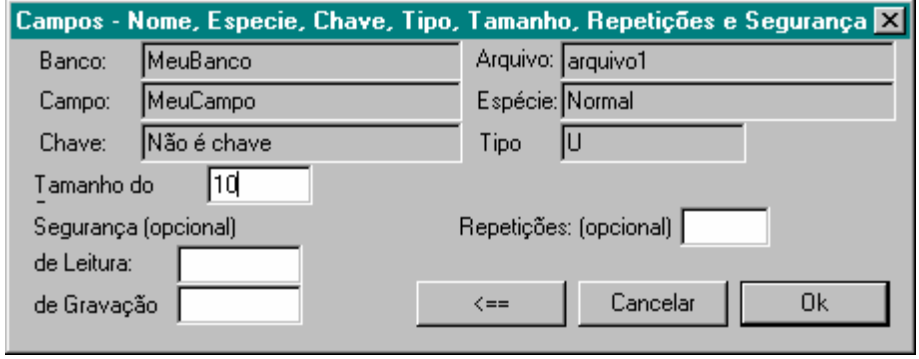
4.13.4 Como criar campos num banco de dados Openbase

Depois que o arquivo foi criado é hora de inserir campos. Na árvore, posicione-se sobre um arquivo. No menu, selecione **Esquema/Inserir Campo em...** ou aperte a tecla INS como atalho. Repare que surge na árvore a mensagem **''Item sendo definido''**. No diálogo para criação de campos, digite o nome do item e a espécie e clique no botão com a seta para direita.

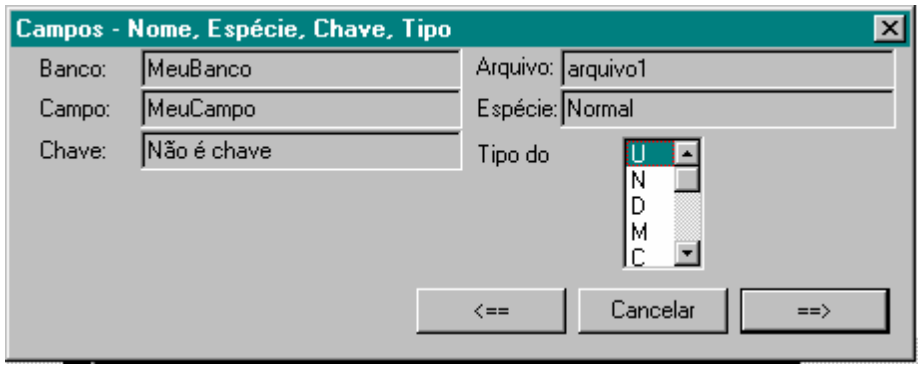


Depois, selecione o tipo da chave,





selecione o tipo do item



e vá para a próxima caixa de diálogo.

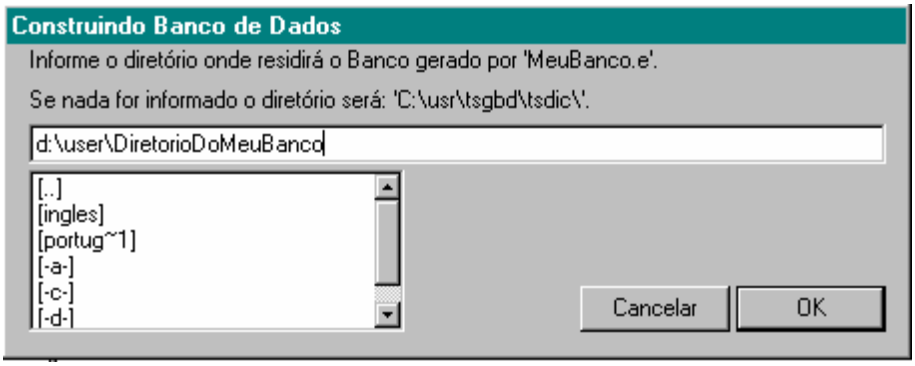
Digite o tamanho do campo, o tipo de segurança e o número de repetições. Clique '**OK**' para adicionar o item ao esquema. Repita para cada item que for inserido no arquivo.

4.13.5 Compilando um esquema Openbase

Para compilar o esquema criado, selecione:

- **Banco/Construir** se **Arquivo/Preferências/Mostrar só Esquemas** tiver sido selecionado ou
- **Banco/Construir Banco TSGBD** se **Arquivo/ Preferências/Mostrar SQL e Esquemas** tiver sido selecionado.

Na caixa de diálogo selecione o diretório onde ficará o banco e clique 'OK'.



4.13.6 Removendo um Banco de Dados Openbase

Selecione na árvore, na seção **Metadata/Bancos de Dados** o banco que será removido. Depois selecione no menu:

- **Banco/Remover** se **Arquivo/Preferências/Mostrar só Esquemas** tiver sido selecionado.
- **Banco/Remover Banco TSGBD** se **Arquivo/ Preferências/Mostrar SQL e Esquemas** tiver sido selecionado.

Você também pode usar a tecla **DEL** como atalho.

4.13.7 Removendo Objetos de um esquema Openbase

Primeiramente, uma das opções abaixo deve estar selecionada:

- **Arquivo/Preferências/Mostrar SQL e Esquemas** ou
- **Arquivo/Mostrar só Esquemas**

Se **Arquivo/Preferências/Mostrar SQL e Esquemas** estiver selecionado:

4.13.7.1 Removendo um esquema:

Para remover um esquema selecione na árvore o esquema que você quer remover, e no menu, selecione **Esquema/Remover Esquema...** ou aperte a tecla **DEL**.

4.13.7.2 Removendo um arquivo:

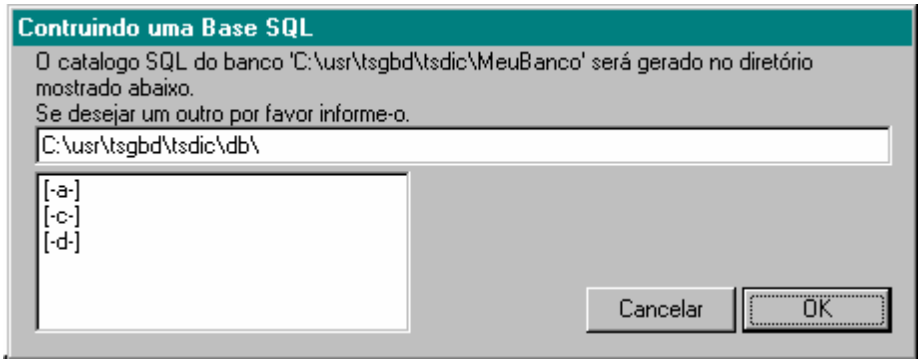
Para remover um arquivo, selecione na árvore o arquivo que voçê quer remover, e no menu, clique em **Esquema/Remover Arquivo...** ou aperte a tecla **DEL**.

4.13.7.3 Removendo um campo:

Para remover um campo, selecione na árvore o campo que voçê quer remover, e no menu, clique em **Esquema/Remover Campo...**.
Se **Arquivo/Mostrar só Esquemas** estiver selecionado: Selecione na árvore o objeto Openbase que quer remover e clique em **Esquema/Remover Objeto** para cada um deles.

4.13.8 Preparando um banco de dados Openbase para acesso SQL

Primeiramente, a opção **Arquivo/ Preferências/Mostrar SQL e Esquemas** deve estar selecionada. Depois selecione na árvore em **Metadata/Bancos de Dados**, o banco que vai ser preparado. Selecionado o banco, ative no menu a opção **Banco/Integrar Catálogo SQL**. Leia o texto que está na dialog box e caso queira, modifique o caminho onde será



gerado o catálogo SQL. O diretório default para criação da base SQL é **c:\usr\tsgbd\tsdic\db**. O diretório db é o diretório de localização dos arquivos que contém informações sobre a base SQL que está sendo criada. Quando você selecionar o diretório, por exemplo, **c:\teste**, você deve acrescentar mais um, por exemplo, **c:\teste\base1**ou **c:\test\db** ou qualquer nome que queira. Este procedimento é obrigatório, caso contrário aparecerá a mensagem 'Diretório já existe'.

4.13.9 Removendo Acesso SQL em um banco de dados Openbase

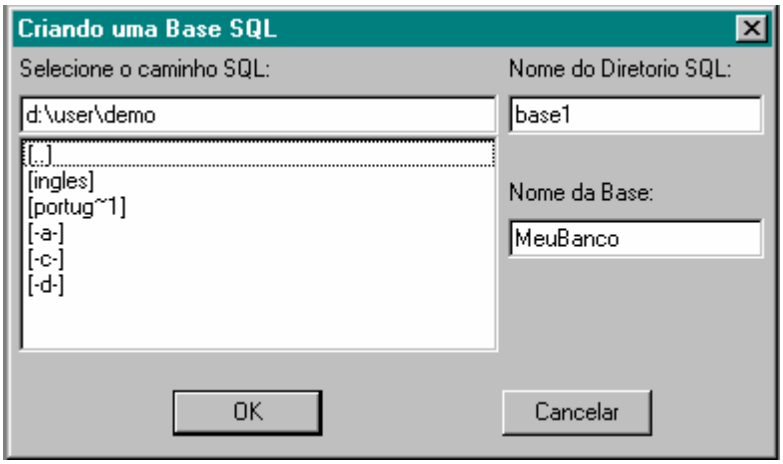
Primeiramente, a opção **Arquivo/Preferências/Mostrar SQL e Esquemas** deve estar selecionada.Selecione na árvore, no caminho **Metadata/Bancos de Dados** um banco que tenha associado a ele o ícone (H). Este ícone indica que o banco esta com o acesso SQL disponível. Depois, selecione no menu **Banco/Remover Catálogo SQL**, << Falta terminar >>.

4.13.10 Criando um banco de dados SQL

Para criar um banco de dados SQL umas das duas opções do menu abaixo precisam estar selecionadas:

- **Arquivo/Preferências/Mostrar SQL e Esquemas** ou
- **Arquivo/Preferências/Mostrar só SQL**

Selecione no menu **Banco/Cria Base SQL**, que ativará a caixa de diálogo abaixo:



- Escolha o caminho SQL. É o caminho básico do diretório que conterá o dicionário de dados SQL.
- Digite o nome do diretório SQL. O diretório **base1** conterá o banco SQL **MeuBanco**.
- O nome da base é o nome pelo qual o banco de dados será identificado. No exemplo acima você criou o banco de dados SQL chamado **MeBanco** no diretório **d:\user\demo\base1**. Note que depois de clicar **'OK'**, o nome do banco (**MeuBanco)** é colocado na árvore, no caminho **Metadata\Banco de Dados SQL**. Neste caminho estão todos os bancos SQL criados. Clicando no nome do banco, abrem-se mais dois ramos da árvore: **Tabelas** e **Visões**. Em **Tabelas** estão todas as Tabelas que fazem parte do banco de dados, e em **Visões** estão todas as visões criadas.

4.13.11 Removendo um banco de dados SQL

Primeiramente é necessário que no menu uma das duas opções abaixo estejam selecionadas:

- **Arquivo/Preferências/Mostrar SQL e Esquemas** ou
- **Arquivo/Preferências/Mostrar só SQL**

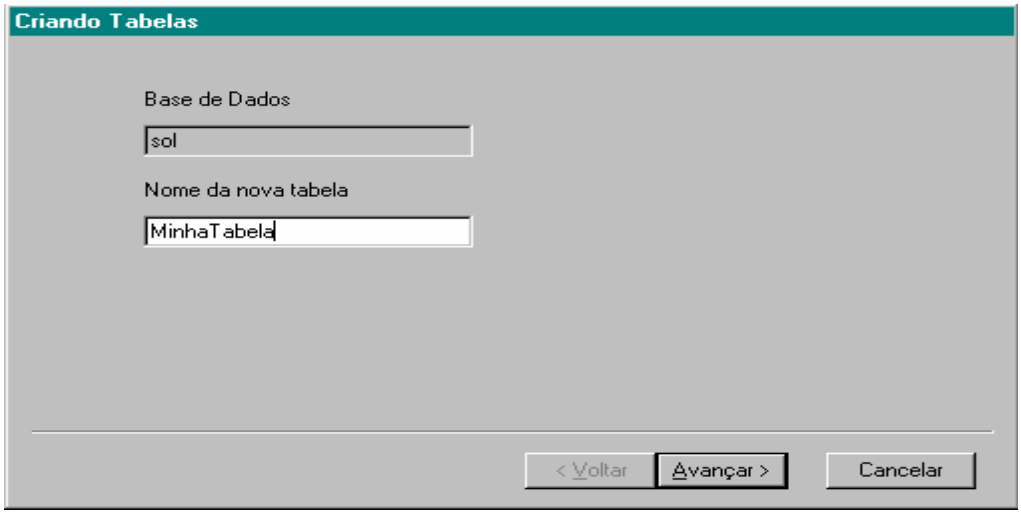
Selecione na árvore, no caminho **Metadata/Bancos de Dados SQL** o banco SQL que quer remover. Depois, selecione no menu **Banco/Remove Base SQL** ou simplesmente aperte a tecla **DEL**. Responda **SIM** a confirmação.

4.13.12 Criando e removendo uma tabela em bancos de dados SQL

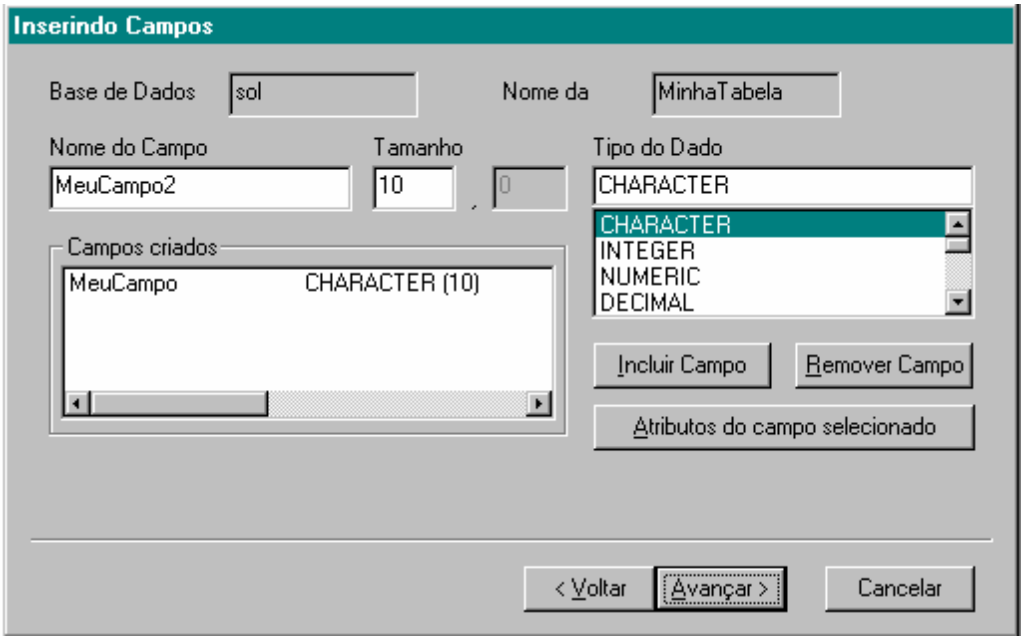
Primeiramente é necessário que no menu uma das duas opções abaixo estejam selecionadas:

- **Arquivo/Preferências/Mostrar SQL e Esquemas** ou
- **Arquivo/Preferências/Mostrar só SQL**

Depois, selecione na árvore o banco de dados SQL que quer usar, no caminho **Metadata/Bancos de Dados SQL**.
No menu, selecione **Banco/Inserir Objeto Sql/Tabelas** e a primeira caixa de dialogo aparece:



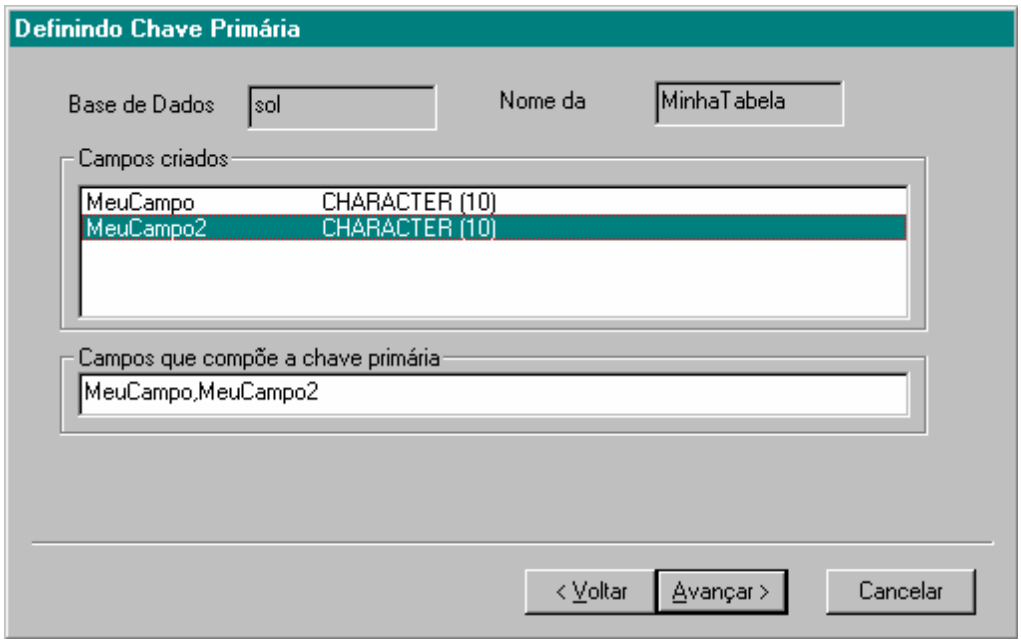
- Digite o nome da tabela e clique em **‘Avançar’**. Repare que o botão só se torna ativo depois que o nome da tabela é digitado.
- No próximo diálogo, digite o nome do campo, selecione o tipo do dado e clique no botão **‘Incluir Campo’**. Repita este passo para cada campo que for incluir na tabela.



Para verificar os atributos de um campo, selecione na lista **‘Campos Criados’** , e clique no botão **‘Atributos do campo selecionado’** . Alguns atributos também podem ser definidos nesta caixa.

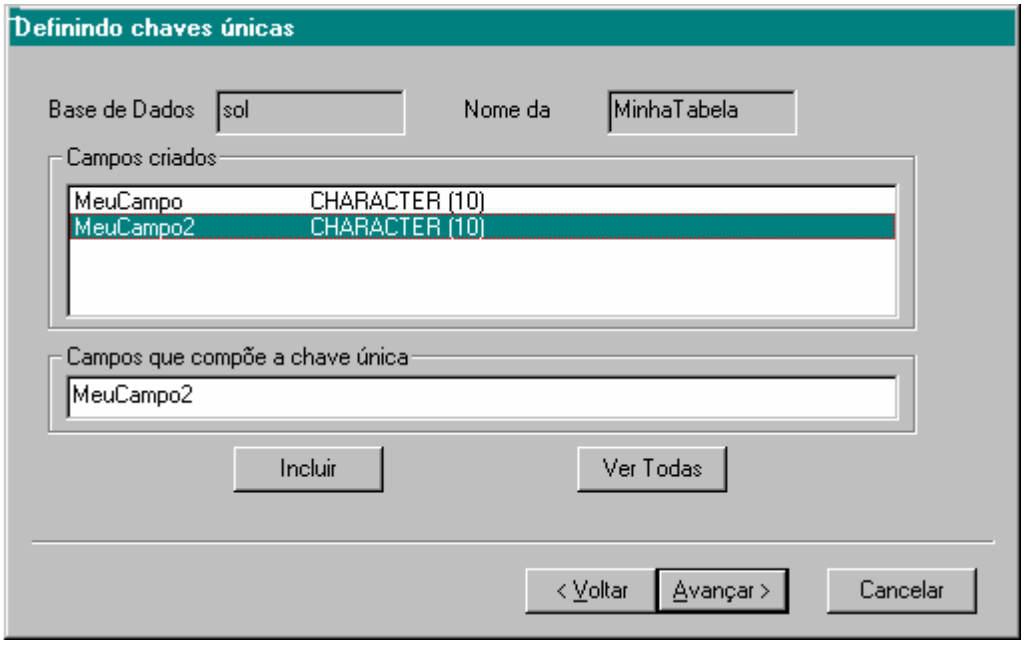
Para remover um campo , selecione na lista **‘Campos Criados’** e clique no botão ‘Remover Campo’. Clique em **‘Avançar’** para continuar.

Para criar uma chave primária , dê um duplo clique na lista ‘Campos Criados’ , em cada campo que pertença a chave. Para remover algum campo da chave, basta posicionar o ponto de inserção na caixa ‘*Campos que compõem a chave primária*’ e remova com a tecla DEL.. Clique em **‘Avançar’** para continuar.

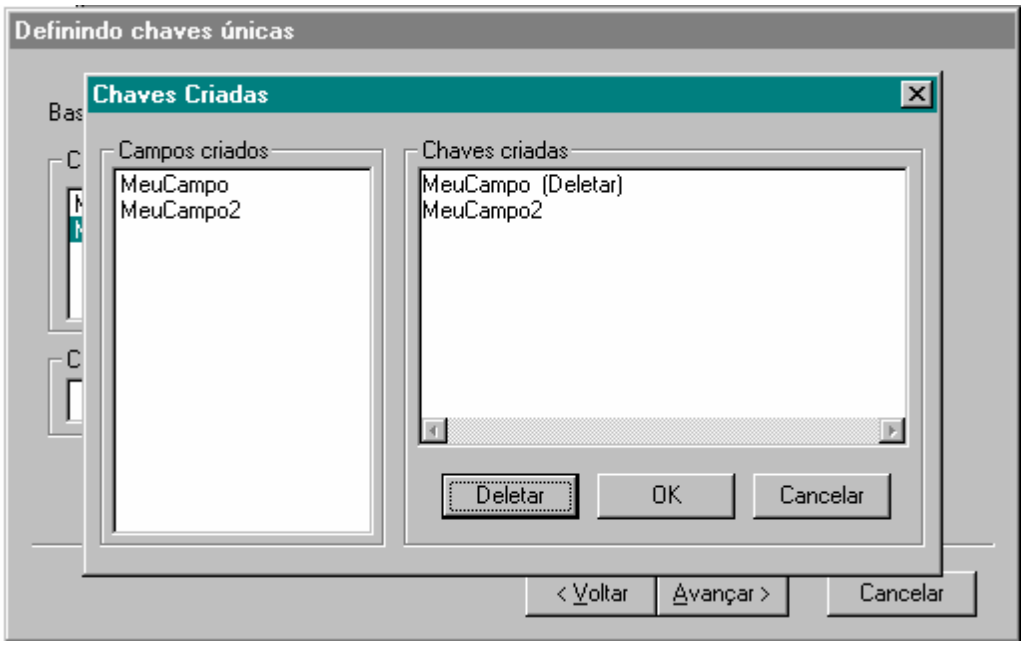


- Para definir chaves únicas, proceda da mesma maneira que o passo 3. Após selecionar os campos que compõe a chave, clique no botão **'Incluir'**. Para cada chave repita o procedimento.

Note que ao criar a primeira chave, o botão **'Ver Todas'** é ativado. Este botão permite que você veja todas as chaves que foram criadas.

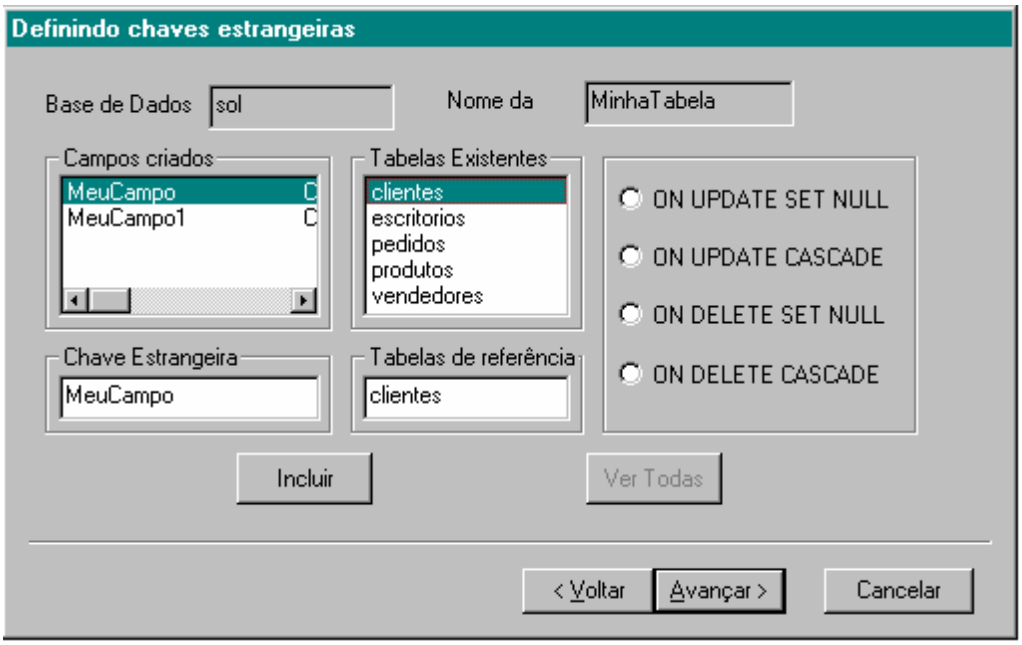


Com um duplo clique sobre um campo na lista **'Campos Criados'** os atributos do campo selecionado são mostradas numa caixa de propriedades.
Para remover uma chave, selecione uma chave na lista **'Chaves Criadas'** e clique no botão **'Deletar'**. A chave é marcada para ser deletada. Para confirmar clique no botão **'OK'**. Para sair do diálogo sem modificações, clique em **'Cancelar'**.



Para continuar, clique em **'Avançar'**.

5) Para criar chaves estrangeiras selecione o campo na lista **'Campos Criados'** e selecione qual a tabela que será a referencia para este campo na lista **'Tabelas Existentes'**. A lista de tabelas existentes só será preenchida com tabelas do banco que **tenham chaves definidas**, senão a lista estará vazia, porque só é possível fazer referência a chaves existentes.
Escolha o tipo de **UPDATE** e **DELETE** e clique em **'Avançar'**.

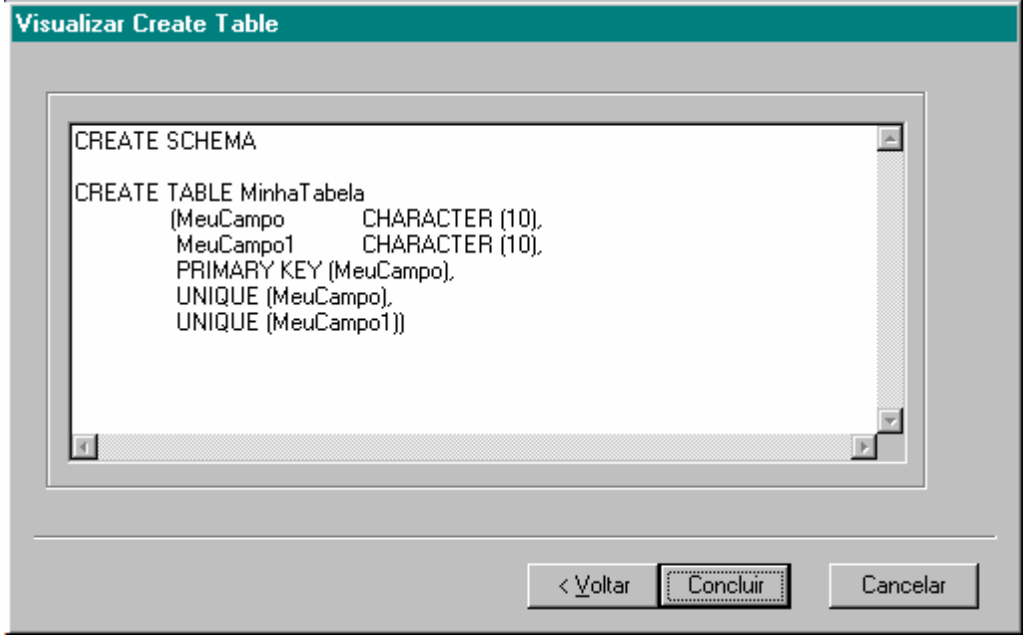


Por fim, entra o último diálogo, que mostra o comando **CREATE TABLE** que dará origem a nova tabela. O comando **CREATE SCHEMA** é usado para identificar que as linhas seguintes serão comandos SQL.

Primeiramente é necessário que no menu uma das duas opções abaixo estejam selecionadas:

- **Arquivo/Preferências/Mostrar SQL e Esquemas** ou
- **Arquivo/Preferências/Mostrar só SQL**

Depois, selecione na árvore o banco de dados SQL que quer usar, no caminho **Metadata/Bancos de Dados SQL**.
Depois, selecione a tabela que quer remover.
No menu, selecione **Banco/Remover Objeto SQL/Tabelas**. Confirme a operação.

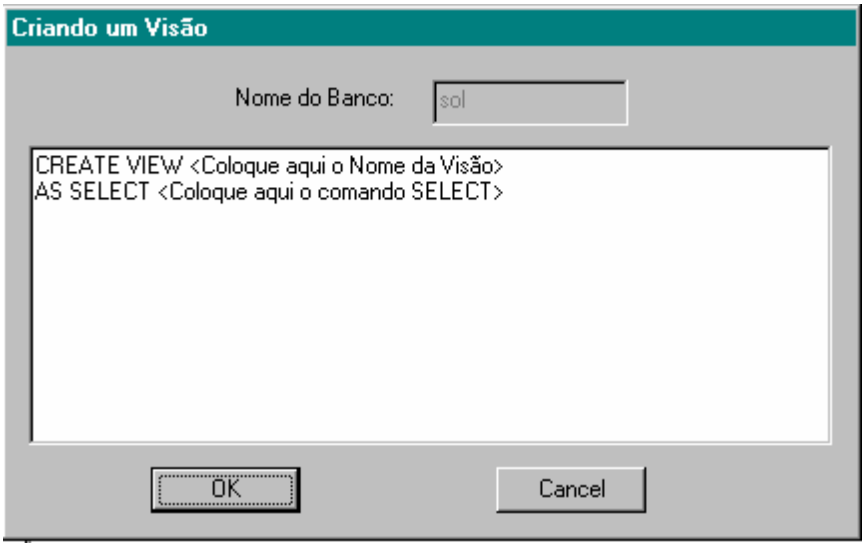


4.13.13 Criando uma visão num banco de dados SQL

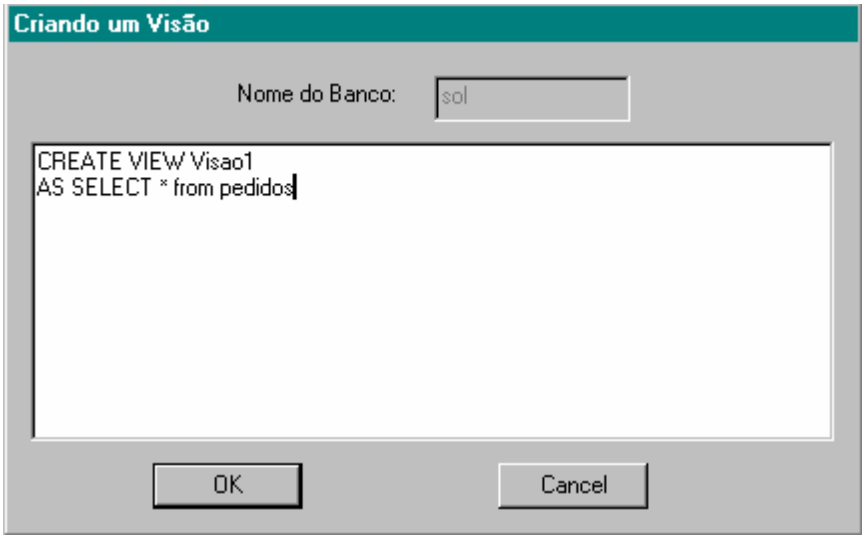
Primeiramente é necessário que no menu uma das duas opções abaixo estejam selecionadas:

- **Arquivo/Preferências/Mostrar SQL e Esquemas** ou
- **Arquivo/Preferências/Mostrar só SQL**

Depois, selecione na árvore o banco de dados SQL que quer usar, no caminho **Metadata/Bancos de Dados SQL**. No menu, selecione **Banco/Inserir Objeto Sql/Visões** e uma caixa de dialogo aparece:



Na caixa de texto já vem escrito um template para o comando.
Substitua **'<Coloque aqui o nome da visão>** pelo nome da visão que vc quer criar.
Substitua **'<Coloque aqui o comando SELECT>** pelo comando de seleção de registros.
Exemplo: Criando uma visão chamada **'Visao1'** que conterá todos os registros do arquivo **'pedidos'**, do banco **'sol'**.



Note que depois que a visão foi criada, ela automaticamente é incluida na árvore, no caminho do banco do qual ela se originou.

4.13.14 Removendo uma visão de um banco de dados SQL

Primeiramente é necessário que no menu uma das duas opções abaixo estejam selecionadas:

- **Arquivo/Preferências/Mostrar SQL e Esquemas** ou
- **Arquivo/Preferências/Mostrar só SQL**

Depois, selecione na árvore o banco de dados SQL que quer usar, no caminho **Metadata/Bancos de Dados SQL**. Depois, selecione a visão que quer remover.
No menu, selecione **Banco/Remover Objeto SQL/Visões**. Confirme a operação

4.13.15 Compilando um esquema SQL

Um esquema SQL é um arquivo texto contendo comandos SQL. O texto deve ser obrigatóriamente iniciado pelo comando **CREATE SCHEMA**, indicando que as linhas a seguir conterão comandos SQL. Para que o esquema seja executado, um arquivo de esquema deve estar aberto, ou o esquema deve estar digitado na janela de texto.
Primeiramente é necessário que no menu uma das duas opções abaixo estejam selecionadas:

- **Arquivo/Preferências/Mostrar SQL e Esquemas** ou
- **Arquivo/Preferências/Mostrar só SQL**

Depois, selecione na árvore o banco de dados SQL que quer usar, no caminho **Metadata/Bancos de Dados SQL**. No menu, selecione **Banco/Executar Script**.
Abre-se um diálogo de abertura de arquivo. Por default, os arquivos que contém esquemas SQL tem extensão **'.esq'**, mas pode se usar qualquer extensão. Escolha o arquivo selecionando-o e clicando **'OK'**. Se o arquivo já estiver aberto, isto é, está na janela de edição, clique em **'Cancelar'**. Depois da operação acima, clique **'OK'** ou **'Cancelar'**, para confirmar ou abortar a execução.Abaixo, vemos um pequeno exemplo de arquivo de esquema:

```
CREATE SCHEMA

CREATE TABLE MinhaTabela
        (MeuCampo       CHARACTER (10),
        MeuCampo1       CHARACTER (10),
        PRIMARY KEY (MeuCampo),
        UNIQUE (MeuCampo),
        UNIQUE (MeuCampo1))
```

4.13.16 Recuperando um banco SQL danificado

Às vezes um banco pode ficar bloqueado por várias razões, como falta de luz, processos bloqueados por um programa mal-comportado, etc. Para resolver este problema, selecione na árvore o banco que você quer recuperar e:
Se você tiver selecionado **Arquivo/Preferências/Mostrar SQL e Esquemas**, clique no menu na opção **Utilitários/Específicos para SQL/Recuperar Banco SQL.**

Se você selecionou **Arquivo/Preferências/Mostrar só SQL**, clique no menu na opção **Utilitários/Recuperar Banco SQL**.

4.13.17 Refazer ou Desfazer Transações num banco de dados SQL

Para que seja possível refazer ou desfazer transações num banco de dados TSQL a opção para criação do jornal deve ter sido ativada quando da criação do banco SQL. Esta opção não deve ser usada quando da criação de acesso SQL para bancos de dados Openbase.
Existem duas maneiras de se ativar a função, dependendo do modo de trabalho selecionado:
Arquivo/Preferências/Mostrar SQL e Esquemas**:**
Selecione na árvore, abaixo de **Bancos de Dados SQL** o banco de dados que deve ter as transações modificadas. Então, clique no menu em **Utilitários/Específicos para SQL/Transações**.

Arquivo/Preferências/Mostrar só SQL

Selecione na árvore, abaixo de **Bancos de Dados SQL** o banco de dados que deve ter as transações modificadas. Então, clique no menu em **Utilitários/Transações**.
No diálogo que é mostrado abaixo, é necessário que algumas informações sejam preenchidas:

O nome do arquivo diário tem como default **\usr\tsgbd\tsdic\diario**, troque o nome do arquivo se necessário. Se for preciso que se comece de uma data específica, digite a data inicial. Se ficar preenchida com o default **dd/mm/aaaa** todas as operações gravadas no arquivo diário serão refeitas ou desfeitas a partir data em o arquivo diário começou ser gravado.

5 Manutenção e recuperação das Bases de Dados

Uma vez concluídas as tarefas de implementação, poderá ser iniciado o processo de manutenção dos Bancos de Dados, processando as informações neles contidas, selecionando, incluindo, alterando ou removendo elementos.
A manutenção das Bases de Dados ***OpenBASE*** é feita, principalmente, através da utilização dos utilitários de administração que apresentaremos mais adiante. Porém, os objetos que compõem os Bancos de Dados ***OpenBASE*** podem ser manipulados através das seguintes maneiras:

- Executando programas desenvolvidos nas linguagens OPUS e OpusWin
- Executando os utilitários do ***OpenBASE*** para processar, remover, incluir, adicionar, recuperar, reorganizar, obter cópias e replicar as informações contidas em Bases de Dados ***OpenBASE***
- Executando programas desenvolvidos em linguagens de alto nível que invoquem as rotinas e métodos de acesso do ***OpenBASE***
- Utilizando as ferramentas RAD tais como Visual Basic ou Delphi, acessando os objetos dos Bancos de Dados através de DLLs (distribuídas com o ***OpenBASE)***
- Usando o sistemas GERAL, ferramenta de consulta e atualização dos Bancos de Dados
- Usando o SQL de forma gráfica e interativa em ambiente Windows (programa TSQLWI)
- Usando o SQL embutido nas linguagens OPUS e OpusWin
- Usando os mecanismos ODBC incluídos nas linguagens OPUS e OpusWin
- Usando os módulos COM/DCOM do OpenBASE
- Utilizando aplicativos Internet desenvolvidos em OpusWeb
- Utilizando aplicativos Internet que utilizam linguagens Script, ASP, PHP, Perl e outros

5.1 Recuperando Bases de Dados OpenBase

Este capítulo tem por finalidade orientar o usuário sobre como proceder em caso de ocorrência de pane no sistema devido, por exemplo, a:

- simples falhas no sistema causadas por queda de energia
- bugs de software
- falhas nos processos do sistema operacional
- falhas no disco rígido
- etc ...

A primeira conseqüência de uma pane no sistema é que todo o trabalho e dados armazenados em memória são perdidos, a não ser que tais informações tenham sido armazenadas em lugar seguro. Entretanto, o usuário OpenBase não precisa se alarmar em caso de pane no sistema, já que é possível recuperar os dados perdidos e re-estabelecer a integridade referencial das Base de Dados.
O OpenBase permite a gravação de um log das transações denominado **diário**, possibilitando, com isso, recuperar todo o trabalho realizado a partir do último backup. O diário contém as transações realizadas nas bases de dados, podendo refazer tais transações.

5.1.1 Como recuperar Bancos de Dados

A seguir explicaremos como funcionam os métodos de recuperação de uma base de dados OpenBase, descrevendo como um esquema deve ser projetado, para uma possível recuperação da base de dados gerada pelo OpenBase, em caso de pane.

5.1.1.1 Elaboração do esquema do Banco

A definição do Banco de Dados deverá incluir algumas cláusulas importantes que permitem implementar métodos de recuperação das informações, conforme sintaxe a seguir:

BANCO [<percurso>] <nome_bd><codigo_de_seguranca>...
...[{ARQRECUP/DIARIO[=<arq>]/DIAREC[=<arq>]/AUTOREC}]...
...[{BLOQARQ/BLOQCHA/BLOQPAG/BLOQREG}]...
...

As cláusulas referenciadas na sintaxe acima já foram devidamente abordadas neste e em outros manuais. Vamos apenas ressaltar alguns aspectos importantes relacionados com a recuperação de Bancos de Dados. Para maiores detalhes, consulte, neste manual, o capítulo “Sistemas de definição de Bancos de Dados” assim como o manual específico do sistema DEFINE.
Veja a seguir as cláusulas pertinentes ao objetivo de recuperação de Bancos de Dados.

ARQRECUP

Determina que sejam criados um ou mais arquivos destinados a armazenar os dados das transações antes deles serem modificados. (imagem anterior do registro), permitindo assim que uma ou mais transações não completadas possam ser desfeitas a qualquer momento (rollback/undo).

DIÁRIO

Determina que seja utilizado o arquivo DIÁRIO (criando pelo programa BDSGBD) para armazenar todos os dados das transações **depois** deles serem modificados (imagem posterior do registro), 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 número. É necessário que a base de dados na qual tais transações serão refeitas, seja a mesma base de dados existente antes da geração do arquivo diário. Por isso, antes de “baixar” o diário, restabeleça via backup, tal base de dados.
Caso o usuário deseje, o arquivo DIÁRIO, poderá ser criado por banco. Para isto, basta que no esquema do banco de dados conforme exemplificado acima, na cláusula DIARIO=<arq> seja informado o nome do arquivo diário em <arq>.
Novas opções foram adicionadas ao utilitário BDSGBD para maximizar a utilização do arquivo diário. O usuário pode, a partir de agora, especificar o tamanho para o arquivo diário e onde serão exibidas mensagens de alerta sobre a gravação deste arquivo. Além destas adições, é possível a partir de agora, a gravação de pontos de sincronismos, forçando a atualização das informações em disco.

DIAREC

Determina que seja utilizado o arquivo DIÁRIO (criando pelo programa BDSGBD) para armazenar todos os dados das transações **antes** deles serem modificados e depois deles serem modificados. (guarda a imagem anterior e posterior do registro), 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

AUTOREC, 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.

5.1.1.2 Iniciação do gerenciador OpenBASE

O gerenciador de ambiente OpenBASE (DBSGBD) deverá ser iniciado conforme a seguir:

bdsgbd [-d –l<numero1> [-a<arquivo>] –t<device> -y<numero2> [-e]]

As cláusulas referenciadas na sintaxe acima já foram devidamente abordadas neste e em outros manuais. Vamos apenas apresentar algumas cláusulas importantes relacionados com a recuperação de Bancos de Dados. Para maiores detalhes, consulte, neste manual, o capítulo "Utilitários do ambiente OpenBASE" assim como o manual específico de utilitários OpenBASE.
Veja a seguir as cláusulas pertinentes ao objetivo de recuperação de Bancos de Dados.

-d

Determina a criação e utilização de jornalização (arquivo diário).

-l<numero1>

Define o tamanho em bytes do arquivo diário. <numero1> deve ser um valor inteiro.

-a <arquivo>

Determina (quando utilizada a opção –d) o nome do arquivo diário. Caso se omita, o nome do arquivo default é /usr/tsgbd/tsdic/diário. Caso se especifique d**iario=<arq>** no esquema do banco de dados, não é preciso utilizar esta opção (apenas no unix).

-t<device>

Define o terminal onde serão exibidas as mensagens sobre a gravação do diário.

-y <numero2>

Define a freqüência para gravação de pontos de sincronismo. **<numero2>** deve ser um valor inteiro.

-e

Determina que na execução deste utilitário o arquivo diário deve ser esvaziando.

Quando o arquivo diário exceder o tamanho definido por <numero1>, o arquivo corrente é fechado e outro é aberto, possibilitando que sejam feitas cópias de segurança a cada troca de arquivo. Após a troca de arquivo a seguinte mensagem será enviada para o terminal indicado em <device>: "Gravando diário em <nome>", onde <nome> é o nome do diário atual. O nome do arquivo diário é acrescido de um dígito numérico, iniciando em 1 (um), e a cada troca de arquivo este dígito é incrementado de uma unidade. São possíveis até 9 (nove) arquivos diários.
A utilização da opção **–y** determina que a cada <numero2> transação serão executados dois comandos "sync" além da gravação de uma transação SYNTRA no arquivo diário, indicando um ponto de sincronismo. Os pontos de sincronismo gravados no arquivo diário servem para futuras recuperações do banco de dados pelos utilitários BDREDI e BDLIDI.
Para que o usuário desfrute plenamente do recurso da jornalização, deve haver uma rotina de backup do banco de dados e dos arquivos diários.
O comando "sync" grava efetivamente no disco as páginas alteradas no buffer pool (páginas armazenas em memória evitando o acesso sucessivo a disco). Deste modo estas informações não são perdidas em eventuais variações na tensão elétrica, isto acarreta em uma diminuição da performance, pois todos o buffers do sistema são atualizados.

5.1.1.3 Utilitários OpenBASE a serem acionados

Os utilitários aqui referenciados já foram devidamente abordadas neste e em outros manuais. Vamos apenas apresentar algumas cláusulas de utilitários relacionados com a recuperação de Bancos de Dados. Para maiores detalhes, consulte, neste manual, o capítulo "Utilitários do ambiente OpenBASE" assim como o manual específico de utilitários OpenBASE.
Veja a seguir algumas opções de utilitários relacionados com recuperação de Bancos de Dados.

BDLIDI

Para facilitar sua utilização foram adicionadas duas novas opções, uma para listar todas as transações de todos os bancos de dados e outra para listar apenas os pontos de sincronismo de um banco de dados. Veja a seguir a sintaxe e argumentos:

bdlidi [-S] –b <banco> -s<seg> -n<nivel> {-d<data-hora> l –t<numero>} [-a<diario>] [-y]

onde:

-S Lista todas as transações de todos os bancos de dados.

<banco> Nome do Banco de Dados

<seg> Representa o código de segurança do banco, especificado no esquema. Se omitido será assumido o valor 1.

<nivel> Representa a palavra de nível do usuário especificado no esquema. Se omitido,será considerado o valor "a".

-d<data_hora> Determina a data e a hora início da listagem. Data e hora devem estar no formato aa/mm/dd-hh:mm:ss.

-a<diario> Representa o nome do arquivo diário quando diferente do DEFAULT.

-t<numero> Representa o número da transação para início da listagem.

-y Lista todos os pontos de sincronismo do arquivo diário.

Permite verificar as transações do diário a partir de determinada data-hora ou a partir de uma determinada transação. Quando se utiliza a opção "-1" no utilitário BDSGBD, para dividir o arquivo diário, deve-se informar o nome do diário que se quer listar informações, caso contrário será listado o diário atual.

BDREDI

Permite refazer ou desfazer transações a partir do arquivo diário. Novas opções foram adicionadas ao utilitário BDREDI otimizando a recuperação de um banco de dados. Com essas adições pode-se recuperar somente um determinado arquivo, utilizar os pontos de sincronismo além de ser possível a geração de um scrip shell com as chamadas ao utilitário BDVERI. Veja a seguir a sintaxe e argumentos:

bdredi –b<banco> -s<seg> -n<nivel> [-r] [-a<diario>] [-y<numero> -f<arquivo>] {-d<data-hora>I t<numero1>}[-e][-O]

Para obter uma explicação a respeito de cada uma das cláusulas apresentadas na sintaxe acima consulte o capítulo "Utilitários do ambiente OpenBASE" deste manual assim como a documentação específica dos utilitários do ambiente OpenBASE.
Este utilitário serve para recuperar bancos de dados a partir de uma data ou transação. A opção "-r" determina que as transações serão desfeitas e só pode ser utilizada para bancos de dados que tenham a opção DIAREC. O utilitário BDREDI ao executar, lê seqüencialmente o arquivo diário pesquisando a data/transação inicial para o banco de dados, prossegue até o fim do arquivo diário para que todas as transações sejam refeitas. Caso a opção "-r" (desfazer), seja especificada a leitura do arquivo diário será de trás para frente, até que a data/transação seja encontrada. Neste caso, cada transação encontrada para o banco de dados especificado, será refeita.
Se a data/transação não for especificada, a recuperação será feita a partir da data 00/00/00- 00:00:00, logo a partir da primeira transação.
Ao utilizar a opção –y, BDREDI lê o arquivo diário a partir do último registro contando os pontos de sincronismos até <numero>, neste instante a leitura do arquivo diário se inverte e as transações são recuperadas até a ultima transação especificada no arquivo diário.
Caso seja especificada a opção –e, o arquivo "bdveri.l" é criado a partir dos registros lidos pelo utilitário BDREDI no arquivo diário. Ao executar tal arquivo (sh bdveri.l) os registros com problemas são corrigidos a partir de chamadas ao utilitário BDINDC, que irá reconstruir somente as chaves com problema.

5.1.1.4 Problemas na abertura de bancos de dados

A seguir, segue um roteiro de como o usuário deve proceder em caso de problemas na abertura do banco de dados. Devem ser utilizados os seguintes utilitários.

BDRECU

Este utilitário serve para recuperar e desbloquear um banco de dados. Para obter uma explicação a respeito de cada uma das cláusulas apresentadas na sintaxe acima consulte o capítulo "Utilitários do ambiente OpenBASE" deste manual assim como a documentação específica dos utilitários do ambiente OpenBASE.
Veja a seguir a sintaxe e argumentos:

bdrecu –b<banco> [-s<segurança>] [-n<nivel>] [-c] [-d<num>] [-O]

O utilitário BDRECU recupera e desbloqueia um Banco de Dados OpenBase e, para Bancos de Dados que possuam arquivo de recuperação, desfaz a última transação pendente no arquivo de recuperação.
Ao executar este utilitário, é mostrado o último modo de abertura e o número de vezes que o banco foi aberto sem ter sido corretamente fechado.
Deve-se executar este utilitário sempre que um processo, que utilize banco de dados OpenBase, seja interrompido de forma anormal, principalmente no uso de bloqueio de dados, onde para cada atualização é gravado um registro no

arquivo de recuperação, de forma que, após a sua execução, o banco de dados volte ao estado normal. No caso de bloqueio central, é criado um arquivo de recuperação para cada processo que efetuar um pedido de bloqueio. Neste caso, o BDRECU verifica se para cada arquivo de recuperação do banco de dados o processo correspondente esta inativo, recuperando somente as transações de processos inativos. O utilitário BDRECU pode ser utilizado a qualquer momento, exceto para bancos de dados que utilizem bloqueio de banco. Neste caso, nenhum usuário deve estar utilizando o banco de dados, para que a tabela de bloqueios seja inicializada e o arquivo de recuperação do banco de dados seja esvaziado.

BDINDC

Este utilitário serve para otimizar a criação de arquivo de índices. Para obter uma explicação a respeito de cada uma das cláusulas apresentadas na sintaxe acima consulte o capítulo “Utilitários do ambiente OpenBASE” deste manual assim como a documentação específica dos utilitários do ambiente OpenBASE.
Veja a seguir a sintaxe e argumentos:

bdindc –b <banco> [-s <segurança> [-n <nível>] [-a <arq_ban> [-S]
[-i {<nome>|<número1>}] [-c] [-v <valor> | -r [-m <numero>]] [-O | -o]

Novas opções foram implementadas ao utilitário BDINDC para otimizar a criação de arquivos de índices, permitindo a reconstrução das chaves de um índice que tenham um determinado valor. Deste modo BDINDC percorrerá toda a cadeia deste valor refazendo apenas estes registros, este utilitário pode ser usado após a verificação do banco de dados pelo utilitário BDVERI que retorna as chaves e valores com erro.
O exemplo a seguir mostra a saída deste utilitário.

BDINDC -b exemplo -s1 -n niv15 -a pessoa -i “NOME.P”

Quando se utiliza a opção "-v <valor>" o arquivo de índice é mantido e somente a cadeia de <valor> é atualizada, por este motivo não teria sentido utilizar esta opção com a opção "-r".

BDVERI

Este utilitário serve para verificar as ligações entre os registros dos arquivos de um banco de dados. Para obter uma explicação a respeito de cada uma das cláusulas apresentadas na sintaxe acima consulte o capítulo “Utilitários do ambiente OpenBASE” deste manual assim como a documentação específica dos utilitários do ambiente OpenBASE.
Veja a sintaxe e argumentosdeste utilitário no capítulo Admoinistração dos ambientes ***Open*BASE.**
O utilitário BDVERI verifica um Banco de Dados quanto a correção das ligações entre registros de seus arquivos. Ao executar, lê os registros de cada arquivo verificando os endereços no cabeçalho do registro e seus índices, emitindo mensagens, se encontrar algum erro.
A opção “-r” é útil para diminuir a quantidade de registros verificados, já que geralmente os problemas ocorrem nos últimos registros dos arquivos.
Visando melhorar a recuperação e verificação de um banco de dados OpenBase e otimizar o algoritmo de verificação foram adicionadas novas opções ao utilitário BDVERI, para que na ocorrência de problemas em um banco de dados a recuperação seja mais rápida e eficiente. Com estas modificações BDVERI não verifica o banco de dados informando apenas os arquivos com problema, e sim arquivos, registros, valores das chaves e grava um arquivo chamado “bdindc.1” que contém chamadas ao utilitário BDINDC para que este reconstrua somente as chaves com problema, diminuindo a necessidade de descarregar e recarregar o arquivo com problema.
Caso o usuário prefira, poderá utilizar os utilitários BDDESC para descarregar todo o banco (ou um determinado arquivo) e BDRECA para carregar todo o banco ou BDADIC para carregar um único arquivo. Tais utilitários para correção de integridade e consistência de dados são utilizados para banco de dados com arquivos com muitos registros, pode ser um processo demorado. A descrição e sintaxe de tais utilitários podem ser obtidas no manual de utilitários da Tecnocoop Sistemas.

BDOTIM

Este utilitário serve para otimizar arquivos de índice ou arquivos de dados. Para obter uma explicação a respeito de cada uma das cláusulas apresentadas na sintaxe acima consulte o capítulo “Utilitários do ambiente OpenBASE” deste manual assim como a documentação específica dos utilitários do ambiente OpenBASE.
Veja a seguir a sintaxe e argumentos:

bdotim -b<banco> [-s<seguranca] [-n<nivel>] [-O] [-a<arquivo>] [-d] [-e] [-p] [-t]

Para otimizar arquivos de índice quando estes estiverem ocupando um espaço muito grande, ou a performance do banco de dados não estiver em bom nível.
Este utilitário, ao otimizar os arquivos de índices, cria uma cópia do arquivo original com a extensão ".V" no diretório do banco de dados.

Mesmo que não se especifique o arquivo do qual se deseja otimizar os índices, o BDOTIM só otimizara aqueles que forem necessários.
Objetivando dinamizar o processo de eliminação de espaços deixados pelos registros deletados em arquivos de dado, foi desenvolvida a opção "-p" para o utilitário BDOTIM. Com esta opção, ao executar este utilitário além de otimizar os arquivos de índice de um banco de dados também reorganiza-se os arquivos de dados retirando os registros "deletados", desta forma o antigo e oneroso processo de descarregar, definir e recarregar o arquivo é desnecessário.

5.1.1.5 Mensagens de erro mais freqüentes

Segue abaixo a relação de erros que podem ocorrer, ao se acessar uma base de dados OpenBase. A descrição destes erros pode ser obtida pelo comando:

bdmens <num do erro>

onde **<num do erro>** é o código do erro em relação ao qual se deseja informação. Estes números podem ser:

Erro 1 "DICIONÁRIO <banco> NÃO ACESSÁVEL"

Ocorre na tentativa de abertura de um banco de dados, quando este não existir ou estiver protegido. Neste caso, consulte o DBA ou verifique:

- Se o nome do banco que se deseja abrir está correto.
- Se as permissões de todo o percurso estão disponíveis.

Erro 2 "DICIONÁRIO <banco> INVÁLIDO"

Ocorre na tentativa de abertura de um banco de dados quando este não for um arquivo no formato do dicionário de dados. Neste caso, verifique se nome do banco que se deseja abrir está correto.

Erro 3 "CÓDIGO DE SEGURANÇA INCORRETO"

Ocorre na tentativa de abertura de um banco de dados (DATABASE), quando o código de segurança passado no programa não é igual ao definido no banco de dados. Neste caso, consulte o DBA ou verifique:

- Se os valores passados no comando DATABASE estão corretos.
- Se o valor passado para variável SECURITY ESTÁ CORRETO.
- O esquema do banco de dados.

Erro 815 "Erro na abertura do arquivo"

Se puder perder os dados já contidos no arquivo, simplesmente passe um define no esquema, recriando apenas aquele arquivo. Se os dados já existentes não puderem ser perdidos, utilize BDDESC no arquivo, e redefina o esquema, recriando apenas este arquivo. Depois utilize BDADIC para recuperar os dados do arquivo. Caso a sugestão acima não resolva, o problema pode estar no tamanho do arquivo que está muito grande ou no número de arquivos abertos ter ultrapassado o limite do sistema operacional . Neste caso ocorrerá o erro 1035, erro de leitura no arquivo. Recomenda-se alterar os parâmetros em maxfiles, nfile (hpux) ou o tamanho máximo de arquivos no arquivo /etc/security/limits (aix)

Erro 1012 "Não pode abrir arquivo"

Este erro ocorre quando algum arquivo de dados ou de índice não foi encontrado. Tais arquivos podem estar protegidos, ou a posição dos números sequenciais dos nomes arquivos de índices, já existentes, está incorreta. Neste último caso, execute o utilitário bdcnfg e acerte a posição(ínicio ou fim) do número do índice nos arquivos de índice e compile(define) o esquema do banco de dados, para que o dicionário de dados seja recriado.

Erro 1035/1036 "LEITURA E GRAVAÇÃO DO ARQUIVO <arquivo>".

Este erro, identificado pelo número 1035 (leitura) ou 1036 (gravação) é um erro indireto do OPENBASE, causado por algum problema físico no disco ou alguma configuração do sistema operacional estar sub-dimensionada. O arquivo pode estar danificado ou este arquivo pode estar com o tamanho muito grande. No primeiro caso, recomenda-se utilizar o fsck do unix, ou scandisk do windows que corrigirá a possível área do disco danificada. No caso de o arquivo em questão estar com o tamanho muito grande, recomenda-se alterar os parâmetros em maxfiles, nfile (hpux) ou o tamanho máximo de arquivos no arquivo /etc/security/limits (aix). Uma terceira causa deste erro, é o numero máximo de arquivos abertos simultaneamente, do sistema operacional Ter estourado. Neste caso, deve-se Rever os parâmetros de configuração do sistema operacional ou utilizar a opção $file.

Erro 813 <arquivo <nome do arquivo> inválido tamanho no registro<num> no dicionário(num1>

Este erro ocorre, quando algum item de algum arquivo no esquema do banco de dados é alterado de tamanho ou tipo ou a este item é atribuído a qualidade de chave; o esquema do banco é compilado(define), o dicionário de dados é re-gerado, mas o arquivo de dados que teve o item alterado, não foi recriado. Com isso, o dicionário de dados contém um determinado tamanho de registro diferente do tamanho do registro no arquivo de dados, já que como o arquivo não foi recriado, ele não alterou o tamanho do registro. Para corrigir este erro, antes de compilar o esquema(define), descarregue o arquivo que foi modificado(bddesc), recrie tal arquivo após compilar o esquema do banco (define) e por fim, re-carregue o arquivo(bdadic).

6 Acesso a Bancos de Dados em ambientes *Open*BASE

A tabela a seguir apresenta os serviços de acesso a bases de Dados e como eles são efetivamente acionados dentro dos diversos componentes dos ambientes OpenBASE.

Serviço solicitado	Comandos (Opus/OpusWin)	Comandos (sistema GERAL)	Rotinas e métodos (bibliotecas dinâmicas)	obs
Leitura seqüencial			ReiniciaCadeia LeProximo[Registro]Sequencial Le[Registro]AnteriorSequencial EscolheChave ObtemRegistrosNoArquivo	(01)

7 Recursos adicionais do ambiente OpenBASE

7.1 Replicação assíncrona e backup incremental

Bancos de Dados ***Open*BASE** podem ser espelhados (copiados) para uma outra máquina, funcionando como um backup incremental.

7.1.1 Sintaxe

No esquema do Banco, deve-se definir o arquivo conforme a seguinte sintaxe:

nome: arquivo e replicacao

7.1.2 Utilização

Ao se definir um esquema com a cláusula REPLICACAO ao lado do nome do arquivo, serão criados automaticamente três novos itens nesse arquivo conforme a seguinte tabela:

Nome do item	Tipo do item
DAT	T8
DEL	L1
CHA (0)	D7 pos (DAT+1)

Você poderá observar que:

- é criado também um arquivo **<dicionário>.S** que guarda a data e hora da última replicação de cada arquivo replicado
- o item DAT armazena a data e hora em que o registro foi incluído, alterado ou excluído
- o item DEL, armazena o valor 1, quando o registro for excluído (exclusão lógica)
- quando um registro é incluído ou alterado, o item DEL recebe o valor 0
- os registros com DEL=1 não são lidos ou excluídos, a menos que a opção DELETED (SET DELETED ON) esteja ligada
- caso seja incluído um registro com o mesmo valor de chave do registro removido logicamente (DEL=1), o item DEL é alterado de 1 para 0

Daí a necessidade de os arquivos replicáveis serem do tipo entidades (chave primária) ou do tipo relacionamento com pelo menos um item chave única.

7.1.3 Funcionamento

No servidor (máquina para onde serão replicados os dados)

- deve ser iniciado o servidor de replicação através do comando:
  nohup bdsrep &.
- é recebido o nome do banco e o arquivo a replicar
- o servidor de replicação (**bdsrep**) lê a chave primária de cada registro recebido e:
  - se não existir e DEL = 0, inclui
  - se existir e DEL=1, exclui
  - se existir e DEL=0, altera

No cliente (máquina de onde vem os dados replicados)

- deve-se executar o comando:

bdcrep –h<host servidor> -b [percurso bd origem] <banco bd origem>
-r [percurso bd destino] <banco bd destino> [-s<seg>] [-n<nivel>]

- o programa **bdsrep** é iniciado no host servidor. (é carregado automaticamente)
- é enviado o nome do banco no servidor e o arquivo a replicar
- serão pesquisados no banco do cliente todos os registros que possuam o item CHA>=data_hora da última replicação (contida no arquivo **<dicionario.S>**) e CHA<=data_hora do sistema operacional. Tais registros são enviados para o servidor
- todos os registros enviados e que possuam DEL=1 serão excluídos fisicamente no cliente
- o programa **bdsrep** na máquina do host servidor é finalizado. (sai do ar)

7.1.4 Observações

- Para o bom funcionamento da replicação assíncrona, os dois bancos devem ser iguais, tanto no cliente quanto no servidor.
- Caso se deseje replicar um arquivo que já possua dados gravados, deve-se:
  - descarregar o arquivo com **bddesc**
  - acrescentar a cláusula REPLICACAO no esquema, ao lado do arquivo que se deseja replicar
  - compilar o banco de dados (**DEFINE**) recriando o arquivo
  - carregar de novo o arquivo com **bdadic**, utilizando-se a opção **–g**

7.2 Banco de dados distribuído com replicação

Para compreender os exemplos a seguir, imagine que os arquivos **arq1**, **arq2** e **arq3** estão presentes nas máquinas **h1**, **h2** e **h3** respectivamente e o arquivo **arq1** também esta presente no servidor **h2**.
O esquema **dist.esq** referente ao Banco de Dados distribuído (denominado **dist** neste exemplo) é descrito a seguir.

```
banco dist 1          arqrecup distrib=h1
      nome: arq1 e servidor=h1 replicação=h2
            C1 (1)            N3
      nome:arq2     R  servidor=h2
            C2 (arq1)         N3
      nome:arq3     E  servidor=h3
            C3(0)             U03
```

O esquema **dist.esq** será compilado executando com o comando:

```
deficli –h h1 dist.esq
```

onde: **h1** é o nome do servidor onde foi especificada a cláusula **distrib**.
Após a compilação do esquema, serão gerados os seguintes arquivos.

dist.B Arquivo de bloqueio de banco de dados gerado nos servidores h1, h2, h3.
dist.R Arquivo de recuperação gerados nos servidores h1,h2,h3.
dist.h Dicionário de dados distribuídos. Gerados nos servidores h1,h2,h3.
dist.L Arquivo de log que registra as transações ocorridas nos arquivos distribuídos pelos servidores que compõem o banco distribuído, estando presente apenas no servidor **h1**.

O Banco de Dados distribuído presente no servidor **h1**, utiliza arquivos do servidor h1 (**arq1**), h2 (**arq1**/**arq2**) e h3 (**arq3**), sendo que o arquivo arq1, presente no servidor h1 é replicado, ou seja, qualquer alteração neste arquivo, é refletido no arquivo arq1, presente no servidor h2. Com isso, o arquivo arq1 do servidor h2, funciona como um backup on-line do arquivo arq1 no servidor h1, sendo útil em caso de pane no servidor h1.
Para funcionar corretamente a implementação de bancos distribuídos com replicação é necessário que o arquivo **bdserv**, esteja carregado em todas as máquinas (hosts) que participam do esquema do banco de dados distribuído. O arquivo **bdsera**, também deve ser carregado no servidor onde será compilado o banco de dados distribuído caso haja necessidade de se garantir a integridade referencial dos arquivos (presença de chave estrangeira).

7.3 Arquivos transpostos

O OpenBASE …

O que é arquivo transposto …

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.

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.

7.3.1 Definição de itens em arquivos transpostos

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 uma lista 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.

7.3.2 Exemplo

nome: arqtran e transposto
item1 (0) n3
sexo u1 valores (m,f)
idaden2 intervalo (0-99)

8 Dúvidas mais freqüentes sobre *Open*BASE

dlls - " Erro de Leitura na rotina <bd...>Client Error...". Como Proceder?

A mensagem é dada no cliente, mas é provocada por algum problema de comunicação com o servidor. Para saber qual é o problema ou corrigi-lo , basta tirar o BDSERV do ar, finalizar toda aplicação que está conectada ao bdserv e colocá-lo de novo no ar (nohup bdserv &). Com isso teremos um arquivo **nohup.out** vazio, no diretório onde foi dado esse comando **nohup**. Caso o erro persista, veja o conteúdo do arquivo **nohup.out**.

"Sistema Openbase não pode executar". Como Proceder?

Verifique se o arquivo FACPRINT está presente em sua máquina.

- Unix -> /usr/lib
- Windows -> /usr/tsgbd

Só deve existir uma arquivo deste, para cada cópia instalada.

"Sistema Openbase não pode executar". Como Proceder?

Prazo de validade de sua cópia, terminou. Consulte o suporte.

"Sistema Openbase Já Instalado". Como Proceder?

Neste caso, sua cópia é shareware, ou seja funciona sem **facprint**.
Entretanto existe um facprint na máquina, se existir, remova-o.

Duas cópias OpenBASE podem estar na mesma máquina?

Sim, no ambiente unix. Basta que as cópias sejam instaladas em diretórios diferentes. Com isso, ao se logar na máquina, o usuário de acordo com os parâmetros definidos em seu .profile, acessarão uma determinada cópia. No ambiente windows95, será considerada a cópia que estiver instalada no diretório definido no path.

O que fazer se o utilitário BDCODI traz informações diferentes das da cópia instalada?

Procure pelo arquivo FACPRINT em sua máquina. Com certeza existe mais de um deles, e está sendo acessado o que não deveria.

Ao se levar dados de um banco para outro, em máquinas e sistemas operacionais diferentes, mesmo usando os utilitários BDDESC e BDADIC/BDRECA, ocorrem alguns erros. O que pode ser ?

Quando são duas máquinas com sistemas operacionais diferentes, recomenda-se o uso da opção **-t** do **BDDESC** para itens binários(D2,D4,I2,I4,O4,M4,Q4). Os arquivos devem ser transportados com ftp(ascii), com exceção do arquivo com extensão **m** (descarga de arquivos com ítens O4,Q4,M4), que é binário.

DLLS - Como converter uma aplicação local para cliente/servidor ?

São 3 passos:

a) Alterar a declaração das rotinas de RotWin32.dll para CliWin32.dll;
b) Executar a funçãoIniciaServidor antes da AbreBancoDeDados;
c) Executar a função FinalizaServidor depois da FechaBancoDeDados;

Se alterar no banco de dados ítens do tipo M4 para ítens do tipo O4 ou Q4, é preciso que se faça alguma alteração nos fontes ?

Não. Internamente o OpenBASE trata a leitura e a gravação de maneira diferente, conforme for o tipo no banco de dados, mesmo sendo o mesmo comando ou função.
Ao se alterar um ítem do tipo M4 para tipo O4 ou Q4 ou vice-versa, deve-se descarregar o arquivo, alterar o esquema. compilar o esquema(define), recriando-se o dicionário de dados e o arquivo que foi alterado, e recarregar tal arquivo(bdadic).

Ao se compilar um fonte com o OPUS ou OPSWIN, vem "Compilação sem Erros", e logo após "Arquivo não Encontrado" 2 vezes, e o executável não é criado. Como Proceder ?

Podem estar ocorrendo duas possibilidades:

- O Compilador C não está instalado na máquina;
- O executável do Compilador C não está sendo acessado, ou seja, o seu diretório não está no PATH.

Se for o primeiro caso, instale o Compilador C. Se for o segundo, deve-se adicionar o tal diretório ao PATH do sistema operacional.

Programa recebeu sinal 11(ambientes DOS/UNIX). Ou Atenção, este programa executou uma operação ilegal e será fechado(Ambiente Windows). Como Proceder?

Este erro é ocasionado, por violação de compartilhamento quando um comando no programa em C que gera a Opus e suas bibliotecas tem alguma área de alocação de memória estourada.
Tal erro pode ser ocasionado, ao se tentar acessar algum arquivo a partir de um percurso que não existe.

O QUE FAZER SE, Ao tentar acessar um base de dados em outra máquina, ocorrerem as mensagens "inicli: não connect 2 " e "inicli: no such file or directory" ?

Não foi possível acessar a máquina servidor a partir de um aplicativo que roda na máquina cliente. Verifique se o arquivo bdserv está rodando no servidor, e se no comando $client=<nome ou "endereço IP do servidor"> está correto, no aplicativo que roda na máquina cliente.(em caso de se utilizar o endereço IP, este deve vir entre aspas.)

Por quê acontece o erro "Arquivo (nome do arquivo) não acessado" ?

Este erro ocorre, ao se tentar alterar ou excluir um registro do arquivo de dados que não foi, previamente, selecionado.

"Opus(varite) - Erro na conversão numérica". Como Proceder ?

Acontece ao se atribuir um valor numérico negativo, a um itém do banco definido como numérico positivo(N) ou na atribuição de um valor numérico de tamanho maior que o tamanho definido no ítem de dado . Para corrigir este erro, aumente o tamanho do ítem (numérico ou data) no banco ou mude o tipo do ítem de N(numérico sem sinal) para S(numérico com sinal).

Ao executar um programa acontece o seguinte erro "Opus(alodecl) - Não alocou vetor <nome do vetor> qtd=0". O que fazer ?

Este erro ocorre quando um programa tenta executar um vetor vazio. Basta carregar o vetor com pelo menos um registro, e tal erro não se repetirá.

Ao compilar um programa acontece o seguinte erro "Excedeu a tabela de símbolos locais". O que pode ser?

Recomenda-se usar $symbols=<número> no código fonte onde ocorre tal mensagem. Com isso o tamanho da tabela de símbolos utilizada pelo compilador OPUS é alterada, possibilitando que os programas que excedam tal limite sejam compilados sem erro. O parâmetro <número>, representa o número de símbolos para tabela e varia de 1 a 5000, onde o default é 512 símbolos.

"Opus(ctod) - erro na conversão de data". Como proceder ?

Este erro ocorre, quando se utiliza a função ctod(vartext) , onde vartext é uma variável alfanumérica , que representam uma data , mas a variável vartext não está no formato "dd/mm/aaaa" (set century on) ou no formato "dd/mm/aa"(set century off)

"DICIONÁRIO <banco> NÃO ACESSÁVEL". O QUE FAZER ?

Ocorre na tentativa de abertura de um banco de dados que não existe ou está protegido. Verifique:

- Se o nome e percurso do banco que se deseja abrir está correto.
- Se as permissões do arquivo e de todo o percurso estão disponíveis.

"DICIONÁRIO <banco> INVÁLIDO". O QUE FAZER ?

Ocorre na tentativa de abertura ou (re)criação de um banco de dados quando já possuir um arquivo no mesmo diretório do dicionário de dados, com o nome deste, ou o arquivo do dicionário de dados possui algum problema físico.
Recomenda-se no primeriro caso, mudar o nome do dicionário de dados ou removê-lo e recriá-lo com **define**(segundo caso)

"CÓDIGO DE SEGURANÇA INCORRETO". O QUE FAZER ?

Ocorre na tentativa de abertura de um banco de dados, quando o código de segurança passado no programa no geral, não é igual ao definido no banco de dados. Verifique:

- Se o valor passado no comando DATABASE estão corretos.
- Se o valor passado na abertuta do banco pelo geral está correto.
- O esquema do banco de dados.

"BANCO DE DADOS NÃO FOI ABERTO". O QUE FAZER ?

Ocorre quando um banco de dados já está aberto, e o dicionário de dados é recriado ou acelerado (copiado). Verifique se ocorreu alguma modificação do dicionário de dados, após o banco ter sido aberto

"PALAVRA DE NÍVEL INCORRETA". O QUE FAZER ?

Ocorre na tentativa de abertura de um banco de dados quando a palavra de nível passada no programa ou pelo geral, é diferente da definida no banco de dados. Verifique:

- Se o valor passado no comando DATABASE estão corretos.
- Se o valor passado na abertura do banco no geral, está correto.
- O esquema do banco de dados.

"MODO DE ACESSO <modo> INCORRETO". O QUE FAZER ?

Ocorre na tentativa de abertura de um banco de dados quando o modo de acesso passado no programa ou pelo geral é diferente do definido no banco de dados. Verifique:

- Se os valores passados no comando DATABASE estão corretos.
- Se o valor passado na abertura do banco no geral, está correto.
- O esquema do banco de dados.

O QUE FAZER QUANDO RECEBER MENSAGEM "FALTA DE MEMÓRIA" ?

Ocorre na tentativa de acesso a um banco de dados quando não existe memória suficiente. Verifique:

- A quantidade de memória disponível para cada usuário.
- O tamanho da sua aplicação.

"BANCO DE DADOS <banco> NÃO ESTÁ DISPONÍVEL PARA USO EXCLUSIVO". COMO FAZER PARA UTILIZÁ-LO ?

Ocorre na tentativa de abertura de um banco de dados em modo exclusivo (modo de abertura igual a 3), quando este já se encontra aberto. Verifique:

- Se existe algum usuário utilizando o banco de dados.
- Se o banco está sendo bloqueado devido a saída anormal do programa(utilize bdrecu)
- Os utilitários bdadic,bddesc,bdindc, bdveri ao abrir o banco , utiliza por default o modo de abertura 3. Caso se utilizar tais utilitários estando o banco já aberto, utilize a opção -l (modo de abertura igual a 2)

O QUE FAZER QUANDO ACONTECER O ERRO "ARQUIVO DE RECUPERAÇÃO NÃO ACESSÁVEL" ?

Ocorre quando for feito um pedido de recuperação das transações anteriores (bdrecu), e o arquivo de recuperação não existir ou não permitir o acesso. Verifique:

- Se ocorreu algum erro durante a abertura do banco de dados.
- O esquema do banco de dados.
- O diretório do banco de dados.
- As permissões para o arquivo de recuperação.

AO EXECUTAR UM PROGRAMA ACONTECE O SEGUINTE ERRO "PROGRAMA BDSGBD NÃO ACESSÁVEL". COMO RESOLVÊ-LO ?

Ocorre na tentativa de abertura de um banco de dados, sem que o BDSGBD esteja inicializado. Verifique:

- Se o processo do programa BDSGDB está ativo.
- Utilize o utilitário bdrmce e coloque o bdsgbd no ar.

"EXCEDIDO MÁXIMO DE USUÁRIOS". COMO RESOLVER ?

Ocorre quando exceder o número de usuários permitidos para a cópia. Verifique:

- O limite de usuários de sua cópia.
- O número máximo de usuários logados, que seu sistema operacional suporta.
- Verifique se o arquivo BDSGBD está no ar.

QUANDO UM PROGRAMA EMITE A MENSAGEM "ARQUIVO <arquivo> DE OUTRO BANCO NÃO ACESSÁVEL". O QUE SIGNIFICA ?

Ocorre na tentativa de abertura de um arquivo TABELA ou CONSULTA, quando esta não é encontrada ou os bits de segurança não permitem o acesso ao arquivo. Verifique:

- Se ocorreu algum erro durante a abertura do banco de dados.
- O valor passado para a variável FILE.
- O diretório onde se encontram os arquivos do banco de dados.
- Os bits de segurança para o arquivo.
- Se o arquivo foi removido.
- O sistema de arquivos de seu equipamento.
- O esquema do banco de dados.

O QUE SIGNIFICA MENSAGEM "ARQUIVO <arquivo> NÃO É ENTIDADE OU TABELA" ?

Ocorre na tentativa de pesquisa por chave primária em um arquivo que não esteja definido como ENTIDADE no esquema

PORQUE ACONTECE A MENSAGEM "ARQUIVO <arquivo> NÃO EXISTE" ?

Ocorre na tentativa de acesso a um arquivo que não exista no banco de dados. Verifique:

- Se ocorreu algum erro durante a abertura do banco de dados.
- O valor passado para variável FILE.
- O nome do arquivo passado para as funções Fs.
- O esquema do banco de dados.
- As permissões do diretório do banco de dados e do aruivo acessado.
- Se o arquivo selecionado no programa executável existe no Banco de Dados.

QUANDO ACONTECER ERRO "REGISTRO DO ARQUIVO ENTIDADE <arquivo> COM LIGAÇÃO". O QUE FAZER ?

Ocorre na tentativa de excluir um registro de uma entidade que tenha ligação com registros de outra entidade.

COMO RESOLVER ERRO "REGISTRO NÃO EXISTE NO ARQUIVO ENTIDADE <arquivo>" ?

Ocorre na tentativa de inclusão ou alteração dos valores de chaves em arquivos detalhe, quando o valor da chave de ligação não existir no arquivo-mestre. Verifique:

- Se o valor atribuídos a chave estrangeira existe no arquivo mestre.
- O esquema do banco de dados.

PORQUE ACONTECE ERRO "REGISTRO JÁ EXISTE NO ARQUIVO <arquivo>" ?

Ocorre na tentativa de inclusão ou alteração do valor de uma chave única que já exista no arquivo.

O QUE FAZER PARA RESOLVER ERRO "PALAVRA DE NÍVEL NÃO PERMITE GRAVAÇÃO DO ITEM <item>" ?

Ocorre na tentativa de alteração dos valores dos itens (inclusão, exclusão e alteração) que tenham nível de gravação superior ao referente a palavra de nível passada na abertura do banco de dados.

QUANDO ACONTECER ERRO "ITEM <item> NÃO PERTENCE AO ARQUIVO <arquivo>", O QUE FAZER PARA CORRIGÍ-LO ?

Ocorre na tentativa de leitura ou gravação dos itens de arquivos que não tenham ligação com o arquivo selecionado ou não pertençam ao arquivo selecionado. Verifique:

- O arquivo selecionado.
- O esquema do banco de dados.

QUANDO ACONTECER ERRO "VALOR DO ITEM <item> NÃO CORRESPONDE COM O SEU TIPO", O QUE FAZER PARA CORRIGÍ-LO ?

Ocorre na tentativa de atribuição de valores, com tipo diferente do tipo definido para o ítem no esquema do banco de dados. Verifique:

- O ítem selecionado.
- O esquema do banco de dados.

"TENTATIVA DE ALTERAÇÃO DO ITEM CHAVE <chave>", COMO RESOLVER ?

Ocorre na tentativa de alteração do valor de uma chave primária de entidade. (veja tópico 42)

"ITEM <item> NÃO É ITEM CHAVE", COMO RESOLVER ?

Ocorre na tentativa de leitura direta de um registro(acesso pelo índice) por um item que não seja item chave.

"FALTOU ITEM CHAVE <chave>", COMO RESOLVER ?

Ocorre na tentativa de inclusão de um registro, onde na lista de itens incluídos falta algum item chave. Verifique:

- Os nomes dos itens passados para o comando REPLACE.
- Os nomes dos itens passados nos vetores para as funções FINSERT( ) e FBINSERT()
- O esquema do banco de dados.

"ITEM <item> NÃO TEM LIGAÇÃO COM O ARQUIVO <arquivo>", O QUE FAZER ?

Ocorre na tentativa de leitura de itens de outros arquivos através do arquivo selecionado (JOIN), quando estes itens pertençam a um arquivo que não tenha relação alguma com o arquivo selecionado. Verifique:

- Os nomes dos itens declarados no programa.
- O esquema do banco de dados.

"ITEM <item> É CHAVE PRIMÁRIA EM ENTIDADE", COMO FAZER ?

Ocorre na tentativa de alteração do valor da chave primária de um arquivo -mestre. Verifique:

- Os nomes dos itens passados para os comando de alteração.
- O esquema do banco de dados.

"REGISTRO FOI MODIFICADO APÓS LEITURA", COMO RESOLVER ?

Ocorre na tentativa de alteração de registros com o comando MODIFY SAME quando o valor do registro é alterado por outro usuário, entre a leitura e a gravação do registro a ser alterado. Verifique se o registro é bloqueado(lock) para leitura

"EXCEDIDO MÁXIMO DE TENTATIVAS DE BLOQUEIO", COMO EVITÁ-LO ?

Ocorre quando exceder o máximo de tentativas de bloqueio em um banco já bloqueado por outro usuário. Verifique:

- Os parâmetros de maxfiles, maxusers e nfile, de seu sistema operacional, caso o número de usuários de seu banco de dados tenha aumentado.
- Se existe algum usuário utilizando o banco de dados.
- Se algum usuário está interrompendo o programa.
- Se o banco está sendo desbloqueado pelo programa.
- A opção de controle $FILE

"BANCO SEM ARQUIVO DE RECUPERAÇÃO". O QUE FAZER SE O OPENBASE EMITIR ESSA MENSAGEM ?

Ocorre na tentativa de desfazer uma transação (comando UNDO), quando no esquema do banco de dados não se especificou a opção ARQRECUP. Pode ocorrer tal erro ao se tentar utilizar um **delete cascade**. Verifique o esquema do banco de dados.

"MODO INVÁLIDO". EM QUE OCASIÃO ESTE ERRO ACONTECE ?

Ocorre na tentativa de se desfazer uma transação (comando UNDO), quando o banco de dados foi aberto com modo de acesso diferente de 2 (dois). Verifique:

- Os comando de bloqueio e desbloqueio.
- O esquema do banco de dados.

"Nao pode abrir arquivo- 'Erro 1012' ". COMO RESOLVÊ-LO ?

Este erro ocorre quando algum arquivo de dados ou de índice não foi encontrado. Tais arquivos podem estar protegidos, ou a posição dos números sequenciais dos nomes arquivos de índices, já existentes, está incorreta. Neste último caso, execute o utilitário bdcnfg e acerte a posição(ínicio ou fim) do número do índice nos arquivos de índice e compile(define) o esquema do banco de dados, para que o dicionário de dados seja recriado.

"ERRO DE LEITURA E GRAVAÇÃO DO ARQUIVO<arquivo>". COMO RESOLVÊ-LO ?

Este erro, identificado pelo número 1035(leitura) ou 1036(gravação) é um erro indireto do OPENBASE, causado por algum problema físico no disco ou alguma configuração do sistema operacional estar sub-dimensionada.
O arquivo pode estar danificado ou este arquivo pode estar com o tamanho muito grande.
No primeiro caso, recomenda-se utilizar o fsck do unix, ou scandisk do windows que corrigirá a possível área do disco danificada
No caso de o arquivo em questão estar com o tamanho muito grande, recomenda-se alterar os parâmetros em maxfiles, nfile (HP UX) ou o tamanho máximo de arquivos no arquivo /etc/security/limits (aix)
Uma terceira causa deste erro, é o numero máximo de arquivos abertos simultaneamente, do sistema operacional Ter estourado. Neste caso, deve-se
Rever os parâmetros de configuração do sistema operacional ou utilizar a opção **$file**.

"Erro 815 - Erro na abertura do arquivo". O QUE FAZER ?

Se puder perder os dados já contidos no arquivo, simplesmente passe um define no esquema, recriando apenas aquele arquivo. Se os dados já existentes não puderem ser perdidos, utilize BDDESC no arquivo, e redefina o esquema, recriando apenas este arquivo. Depois utilize BDADIC para recuperar os dados do arquivo. Caso a sugestão acima não resolva, o problema pode estar no tamanho do arquivo que está muito grande ou no número de arquivos abertos ter ultrapassado o limite do sistema operacional . Neste caso ocorrerá o erro 1035, erro de leitura no arquivo. Recomenda-se alterar os parâmetros em maxfiles,nfile (hpux) ou o tamanho máximo de arquivos no arquivo /etc/security/limits (aix)

Ao atualizar a minha versão do Opebase, preciso compilar todos os fontes?

Depende: Se a mudança no OPENBASE afetar apenas as bibliotecas(libfacbib, libbd, libbdcli, facaux.lib, cliaux.lib, rotaux.lib) apenas o programa principal que faz a linkedição com estas novas bibliotecas, deve seve ser compilado.
Se a mudança no Openbase afetar a opus, opuscli, opuswin.exe ou opuswcli.exe aí devem-se recompilar o(s) programa(s).

Ao utilizar o comando chexm, não consigo acessar o conteúdo de um determinado ítem de um arquivo, estando no meu programa, posicionado em um ítem de outro arquivo, que é chave estrangeira em relação ao primeiro arquivo ?

Se sua base de dados estiver sendo acessada remotamente verifique se o arquivo bdsera esta carregado na máquina servidor.
Se o acesso for local, existe algum problema nos índices dos arquivos em questão. Utilize o utilitário bdindc para refazer os índices destes arquivos.

"Erro 813 -<arquivo <nome do arquivo> inválido tamanho no registro<num> no dicionário(num1>". COMO RESOLVÊ-LO ?

Este erro ocorre, quando algum item de algum arquivo no esquema do banco de dados é alterado de tamanho ou tipo ou a este item é atribuído a qualidade de chave; o esquema do banco é compilado(define), o dicionário de dados é regerado, mas o arquivo de dados que teve o item alterado, não foi recriado. Com isso, o dicionário de dados contém um determinado tamanho de registro diferente do tamanho do registro no arquivo de dados, já que como o arquivo não foi recriado, ele não alterou o tamanho do registro. Para corrigir este erro, antes de compilar o esquema(define), descarregue o arquivo que foi modificado(bddesc), recrie tal arquivo após compilar o esquema do banco (define) e por fim, recarregue o arquivo(bdadic).