

ÍNDICE:

Como construir um sistema utilizando Delphi?.....	02
Como criar uma nova aplicação no Delphi?.....	03
Como salvar a aplicação no Delphi?.....	03
Como alterar as propriedades da tela?	04
Como executar a aplicação no Delphi?	05
Como fechar a aplicação?	05
Como abrir a aplicação?	06
Como inserir uma imagem na tela?	07
Como criar uma tela de splash?	08
Como criar uma lista de ações?	09
Como criar um menu de opções?	10
Como inserir uma lista de imagens?	11
Como criar uma barra de ferramentas?	12
Como criar uma barra de status do sistema?	13
Como inserir data e hora na barra de status do sistema?	13
Como pedir confirmação ao fechar tela na aplicação?.....	14
Como criar uma nova tela na aplicação?.....	14
Como remover uma tela da aplicação?.....	15
Como preparar o banco de dados para ser usado no sistema?	16
Como criar uma tela de login do sistema?	19
Como alterar a Main Form do sistema?	21
Como criar uma tela padrão?	22
Como utilizar uma tela padrão do sistema?	23
Como utilizar um DBGrid?	24
Como utilizar um DBNavigator?	25
Como inserir campos de uma tabela na tela?	26
Como chamar uma tela no Delphi?	27
Como criar uma tela manualmente na aplicação?	28
Como criar relacionamento Master/Detail na aplicação?	29
Como utilizar componentes do tipo LookUp?	30
Como criar consultas em tabelas do banco de dados?	31
Como utilizar filtro em uma tabela?	32
Como usar a linguagem SQL no Delphi?	33
Como criar um relatório com o QuickReport?	35
Como criar um relatório no Rave Reports?	36
Como criar um relatório com a linguagem Rave Reports?	38
Como validar dados e tratar erros no Delphi?	41
Como criar campos calculados?	42
Como criar campos Lookup?	43
Como exportar / importar dados com Delphi/Excel?	44
Como utilizar uma Unit não vinculada a forms?	45
Como utilizar uma DLL?	46
Como preparar um sistema para ser utilizado em rede?	47
Como criar um sistema de ajuda?	48
Como criar discos de instalação para a aplicação?	51
Como compactar o banco de dados do Access no Delphi?	53
Como criar uma rotina de Backup/Restore do sistema?	54
Como criar um gráfico no Delphi?	56
Como criar um relatório de produtos em falta.....	58
Como realizar algumas melhorias no sistema?	59
Como criar um objeto em tempo de execução?	60
Como estruturar o banco de dados “Banco.MDB”	61

Como construir um sistema utilizando Delphi?

Para cada comando a ser executado nesta apostila será utilizada a barra de menu do Delphi, ou, quando existir, ou for mais conveniente, uma combinação de teclas, ou ainda, uma tecla de função. Por exemplo: para visualizar a tela **Object Inspector**, podemos utilizar a barra de menus, **View -> Object Inspector**, ou simplesmente teclar **[F11]**.

Em alguns casos, quando não for utilizada nem a barra de menus, ou uma tecla de função, ou combinação de teclas, o procedimento será solicitado através de uma linha de texto, como por exemplo, “✓ Selecione a guia **Additional** na paleta de componentes do Delphi e insira um componente **Image**”.

Procedimentos a serem executados:

- ✓ O primeiro passo a ser dado para se construir um sistema, em qualquer linguagem de programação, é fazer a análise do mesmo junto às pessoas envolvidas, para obter informações a respeito dos documentos, formulários, procedimentos, enfim, tudo o que faz parte do sistema atual, seja ele informatizado ou manual, e que deverá fazer parte do novo sistema. Anote tudo o que for coletado e faça um esboço num editor de textos, dos menus do sistema, suas telas, consultas e relatórios a serem desenvolvidos.
- ✓ Defina o banco de dados do sistema, com suas tabelas e relacionamentos. Procure eliminar todas as redundâncias de dados.
- ✓ Abra o Windows Explorer (tecle **[F] + [E]**) e crie a pasta:

C:\Sistema

- ✓ Crie, neste momento, no Microsoft Access, o banco de dados que será utilizado pelo sistema. Sua estrutura se encontra nas páginas **61 e 62** desta apostila.

Após executarmos os procedimentos anteriores, já podemos iniciar a construção do nosso sistema utilizando o **Delphi**.

Bom trabalho!

Ronaldo Lavestein – Casa Branca - SP

Como criar uma nova aplicação no Delphi?

Este tópico deverá ser realizado uma única vez para cada sistema a ser criado.

Procedimentos a serem executados:

- ✓ **File** -> **New** -> **Application**


Como salvar a aplicação no Delphi?

Este tópico deve ser executado para a aplicação ser gravada em disco.

Dica: Nunca deixe para salvar somente quando terminar tudo.

Atenção: Organize o seu trabalho, utilizando as pastas criadas anteriormente.

Procedimentos a serem executados:

- ✓ **File** -> **Save All** ou clique no botão 

Escolha a pasta **C:\Sistema\Programas** na Combobox **Salvar em** para indicar o local onde a aplicação será gravada.



Figura 1 - Salvando a unit do menu principal

- ✓ Digite o nome da Unit da tela principal, por exemplo, **UMenuPrin.pas**
- ✓ Digite o nome do projeto, por exemplo, **Sistema.dpr**

Como alterar as propriedades da tela?

Este tópico deve ser realizado, preferencialmente, antes de se introduzir qualquer componente na tela.

Procedimentos a serem executados:

- ✓ Selecione a tela **Form1** através da janela **Object TreeView** e altere suas propriedades na guia **Properties** da janela **Object Inspector**

OBS: O ícone **HandShak.ico** encontra-se na pasta:

C:\Arquivos de Programas\Arquivos Comuns\Borland Shared\Images\Icons

Propriedades:	Conteúdo	Significado
+ BorderIcons biMaximize: <i>False</i> (clique no sinal +)		botão Maximizar desabilitado
BorderStyle: <i>bsSingle</i>		borda simples, tamanho fixo
Caption: <i>Sistema Comercial versão 1.0</i>		legenda da barra de títulos.
Color: <i>clBtnFace</i>		cor de fundo da tela
Height: <i>480</i>		altura em pixels
Icon...: Load (carregue o ícone HandShak.ico)		Ícone da barra de títulos da janela
Name: <i>FrmMenuPrin</i>		nome da tela
Position: <i>poScreenCenter</i>		posiciona a janela no centro da tela
Width: <i>640</i>		largura em pixels
WindowState: <i>wsNormal</i>		abre a tela no tamanho original

Como executar a aplicação no Delphi?

Este tópico deve ser realizado para fazer o programa funcionar.

Procedimentos a serem executados:

- ✓ Escolha a opção **Run -> Run**, na barra de menus, ou simplesmente, tecla **[F9]** para executar a aplicação



Figura 2 - Tela principal após a execução do sistema

O arquivo executável (.exe), e outros, criados no momento da execução do sistema, serão gravados dentro da pasta onde se encontra o código fonte (*.pas e *.dpr), a não ser que sejam especificadas outras pastas através do comando **Project -> Option -> Directories / Conditionals** da barra de menus do Delphi.

Os arquivos de extensão ***.dcu**, ***.~***, ***.exe** podem ser apagados, pois os mesmos são reconstruídos após executarmos o comando **[F9]**.

Como fechar a aplicação?

Este tópico deverá ser executado para fechar a aplicação. Lembre-se de salvá-la antes disso.

Procedimentos a serem executados:

- ✓ **File -> Close All**

Como abrir a aplicação?

Execute este tópico antes de iniciar seu trabalho.

Procedimentos a serem executados:

- ✓ **File -> Open Project**
- ✓ Escolha, como exemplo, na caixa de diálogo a aplicação **Sistema.dpr**, que encontra-se na pasta **C:\Sistema\Programas**.

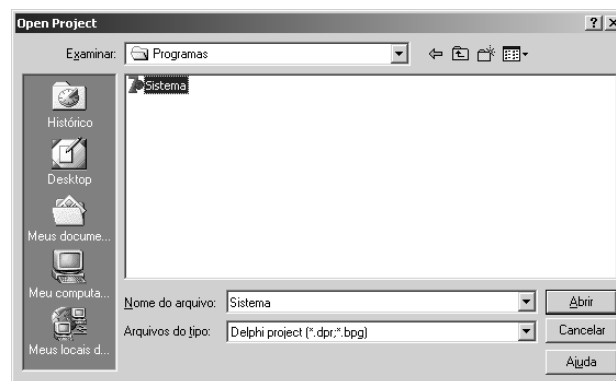



Figura 3 - Abrindo o projeto Sistema.dpr

Como inserir uma imagem na tela?

Este tópico serve para qualquer situação em que se queira inserir uma ou mais imagens na tela, mas neste exemplo, mostrarei como inserir uma imagem de fundo na tela.

Procedimentos a serem executados:

- ✓ Selecione a guia **Additional** na paleta de componentes do Delphi e insira um componente **Image** 
- ✓ Selecione o objeto **Image1** que foi inserido na tela e altere as propriedades abaixo, na janela **Object Inspector**:

OBS: Escolha para a propriedade **Picture:HandShak.bmp**, que se encontra na pasta:
C:\Arquivos de Programas\Arquivos Comuns\Borland Shared\Images\Splash\256Color

Propriedades:

Conteúdo

Align:	AIClient (Alinha a imagem para exibi-la na tela inteira).
Autosize:	True (redimensionamento automático)
Picture...	clique no botão Load e procure a figura desejada na pasta
Name:	ImgFundo (nome do objeto Image)
Stretch:	True (estica a imagem, até ficar do tamanho do componente)

Como criar uma tela de Splash?

Tela de splash ou de abertura é a primeira tela exibida quando o programa é executado. Sua exibição dura apenas alguns instantes.

Procedimentos a serem executados:

- ✓ **File** -> New -> Form (cria uma nova tela)
- ✓ **File** -> Save All (Salve a Unit como **USplash** na pasta **Programas**)
- ✓ Altere as **propriedades** da tela **Form1**, conforme tabela a seguir:

Name	FrmSplash	BorderStyle	bsNone
Height	250	Position	poScreenCenter
Width	400	Color	clGray

- ✓ Acrescente um objeto **Image** à tela e altere suas **propriedades**:

Height	210	Top	3
Width	385	Left	3
Stretch	True	Picture...	HandShak.bm p

- ✓ Acrescente um objeto **Label** à tela e altere suas **propriedades**:

Font.Color	clWhite	Font.Size	26
Transparent	True	Left	45
Caption	Sistema Comercial 1.0	Top	90

- ✓ Selecione, na barra de menus, a opção **Project** -> View Source para exibir o código fonte da aplicação.
- ✓ Procure a linha de comando **Application.CreateForm(TFrmSplash, FrmSplash)** e insira os comandos abaixo após a mesma :

```
FrmSplash.Show; //exibe a tela FrmSplash  
FrmSplash.Refresh; // dá um refresh na tela  
Sleep(2000); // (aguarda 02 segundos) – inclua a unit  
SysUtils na cláusula Uses.
```

aplicação até este ponto, execute-a, teclando **[F9]**.

- ✓ Para
testar a

Como criar uma lista de ações?

Uma lista de ações facilita a manutenção do sistema, pois cada comando é digitado uma única vez e pode ser utilizado em diversos pontos do sistema, por exemplo, uma opção de menu e um botão que permitem sair do sistema.

Procedimentos a serem executados:

- ✓ Selecione na paleta de componentes do Delphi a guia **Standard** e insira o componente **ActionList** na tela **FrmMenuPrin**.
- ✓ Selecione o objeto **ActionList1** na tela e dê um **duplo-clique** nele.
- ✓ Pressione a tecla **[Insert]** seis vezes para inserir seis ações que farão parte das opções dos menus *Cadastro* e *Sair*. Caso deseje apagar uma ação tecle **[Delete]** após selecioná-la.
- ✓ Selecione cada ação criada e altere suas propriedades **Caption** (legenda da ação), **Name** (nome da ação), **ShortCut** (tecla de atalho que executa a ação) e **Hint** (dica), conforme a tabela abaixo:

Propriedades

Ações	Caption	Name	ShortCut	Hint
Action1	<i>&Clientes</i>	<i>ActCliente</i>	<i>F2</i>	<i>Cadastro de Clientes</i>
Action2	<i>&Fornecedores</i>	<i>ActFornec</i>	<i>F3</i>	<i>Cadastro de Fornecedores</i>
Action3	<i>F&uncionários</i>	<i>ActFuncio</i>	<i>F4</i>	<i>Cadastro de Funcionários</i>
Action4	<i>Ca&tegorias</i>	<i>ActCatego</i>	<i>F5</i>	<i>Cadastro de Categorias</i>
Action5	<i>&Itens</i>	<i>ActItens</i>	<i>F6</i>	<i>Cadastro de Itens</i>
Action6	<i>&Sair</i>	<i>ActSair</i>	<i>F7</i>	<i>Sair do Sistema</i>

- ✓ Selecione todas as ações e altere a propriedade **Category** para **Cadastro**.
- ✓ Feche a janela de edição do **ActionList1**
- ✓ O comando de cada ação do **ActionList1** devem ser digitados no evento **OnExecute** das mesmas. Como exemplo, dê duplo-clique no objeto **ActionList1** e selecione a ação **ActSair**. Na janela **Object Inspector**, guia **Events**, selecione evento **OnExecute** e digite o comando a seguir em sua procedure:

Close; {Fecha a tela atual}

Como criar um menu de opções?

Este tópico ensina como criar menus de opções.

Procedimentos a serem executados:

- ✓ Insira um componente **MainMenu** (guia **Standard**) na tela principal
- ✓ Selecione o objeto **MainMenu1** e dê um **duplo-clique** no mesmo
- ✓ Digite as opções abaixo em sua propriedade **Caption**



- ✓ Acrescente as demais opções do menu através da propriedade **Action** (propriedade que exibe as ações do **ActionList**) e deixe seu menu conforme mostra a próxima figura.

Dicas:

Para criar os separadores, digite sinal de subtração (-) na propriedade **Caption** e tecle **[Enter]**

Para inserir uma opção em branco tecle **[Insert]**

Para apagar uma opção Tecle **[Delete]**

A opção **Produtos** deve ser digitada na propriedade **Caption**

Para criar o submenu **Produtos**, tecle **[Ctrl] + [→]** sobre a opção.



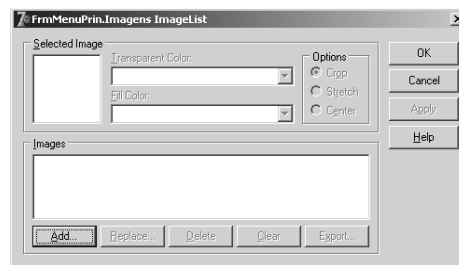
Obs.: Somente após digitarmos os comandos das opções do menu no evento **OnExecute** de cada ação da **ActionList1**, é que as mesmas serão habilitadas. Observe que só a opção **Sair** é que está habilitada até agora.

Como inserir uma lista de imagens?

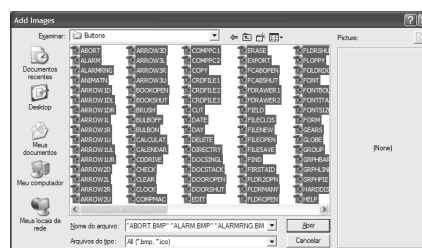
Este tópico tem o propósito de ensinar como criar um “depósito” de imagens para ser utilizado posteriormente na aplicação.

Procedimentos a serem executados:

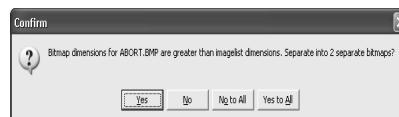
- ✓ Insira um componente **ImageList** da guia **Win32** na tela principal.
- ✓ Dê um duplo-clique sobre **ImageList1** para acessar a tela a seguir



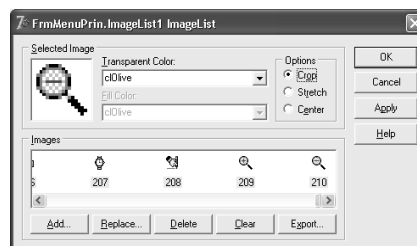
- ✓ Clique em **Add...**, abra a pasta *C:\Arquivos de Programas\Arquivos Comuns\Borland Shared\Images\Buttons*, tecle **[Ctrl] + [A]**, e clique em **Abrir**



- ✓ Clique sobre o botão **Not to All**



- ✓ Selecione a opção **Crop** e clique **OK** para terminar.



- ✓ Para as imagens aparecerem no menu, altere a propriedade **Images** dos objetos **MainMenu1** e **ActionList** para **ImageList1** e escolha, para cada ação da **ActionList1**, o nº de sua respectiva imagem através da propriedade **ImageIndex**.

Como criar uma barra de ferramentas?

Este tópico tem o propósito de ensinar como criar uma barra de ferramentas com botões representando cada opção do menu principal.

Procedimentos a serem executados:

- ✓ Insira um componente **ToolBar** da guia **Win32** da paleta de componentes
- ✓ Altere a propriedade **ShowHint** para **True** (permite exibir as dicas de **Hint**)
- ✓ Altere a propriedade **Images** do objeto **ToolBar1** para **ImageList1**.
- ✓ Clique com o botão direito do mouse sobre a barra de ferramentas e escolha **New Button** para inserir um novo botão ou **New Separator** para inserir um separador. Em nosso exemplo insira respectivamente **05 botões**, **01 separador**, e **01 botão**.
- ✓ Para cada botão altere a propriedade **Action**, de acordo com a respectiva opção do menu.



Obs.: Somente após digitarmos os comandos no evento **OnExecute** de cada ação da **ActionList1**, é que os botões serão habilitados. Até o presente momento, somente o botão Sair está habilitado. Execute a aplicação e confira o resultado.

Como criar uma barra de status do sistema?

Este tópico tem o propósito de ensinar como criar uma barra de status do sistema para apresentar informações ao usuário como, data e hora, usuário, dentre outras.

Procedimentos a serem executados:

- ✓ Insira um componente **StatusBar** da guia **Win32** na tela principal.
- ✓ Dê um duplo-clique sobre o mesmo na tela.
- ✓ Tecle **[Insert]** 03 vezes para criar 03 painéis (divisões) na barra de status
- ✓ Selecione o primeiro painel e altere a propriedade **Width** para **80** (largura).
- ✓ Selecione o segundo painel e altere a propriedade **Width** para **300**
- ✓ Feche a janela da barra de status.

Como inserir data e hora na barra de status do sistema?

Este tópico tem o propósito de ensinar como inserir a data e a hora atual do sistema na barra de status do menu principal.

Procedimentos a serem executados:

- ✓ Para que a hora e a data sejam atualizadas constantemente, **insira** um componente **Timer** da guia **System** na tela principal (**FrmMenuPrin**).
- ✓ Altere a propriedade **Interval** do **Timer** para **1000** para que o seu único evento, o **OnTimer**, seja executado de 01 em 01 segundo.
- ✓ Dê um **duplo-clique** sobre o componente **Timer1** para acessar o evento **OnTimer** e digite os comandos para exibir a hora e a data atual do sistema:

```
StatusBar1.Panels[0].Text := '' + FormatDateTime('hh:nn:ss',now); // 11:08:45
StatusBar1.Panels[1].Text := '' + FormatDateTime('dddd', "dd" de "mmmm"
de "yyyy',now); // domingo, 14 de agosto de 2005
```

Como pedir confirmação ao fechar tela na aplicação?

Este tópico deve ser utilizado sempre que se desejar perguntar ao usuário se ele deseja realmente fechar uma tela do sistema.

Procedimentos a serem executados:

- ✓ Digite os seguintes comandos no evento **OnClose** da tela desejada (em nosso exemplo a tela **FrmMenuPrin**):

```
// Sintaxe: MessageDlg('Mensagem', tipo da tela, [botões], índice ajuda);  
if MessageDlg('Deseja fechar a aplicação?', mtConfirmation,  
    [mbYes, mbNo], 0) = mrYes then  
begin  
    // Dm.tab_Usuarios.Close; // remova o comentário após criar a tela DM  
    Action := caFree; //ação do objeto TCloseAction para fechar a tela  
end  
else  
    Action := caNone ; //ação do objeto TCloseAction para não fechar a tela
```

Obs: Outras opções de Caixa de Diálogo **MessageDlg** para usar em outra ocasião:

Tipos de tela: mtConfirmation, mtWarning, mtError, mtInformation, mtCustom

Botões: mbOk, mbCancel, mbYes, mbNo, mbAll, mbRetry, mbYesToAll, mbNoToAll, mbAbort

Respostas aos botões: mrOk, mrCancel, mrYes, mrNo, etc.

Como criar uma nova tela na aplicação?

Este tópico deve ser utilizado para cada nova tela a ser inserida no sistema.

Procedimentos a serem executados:

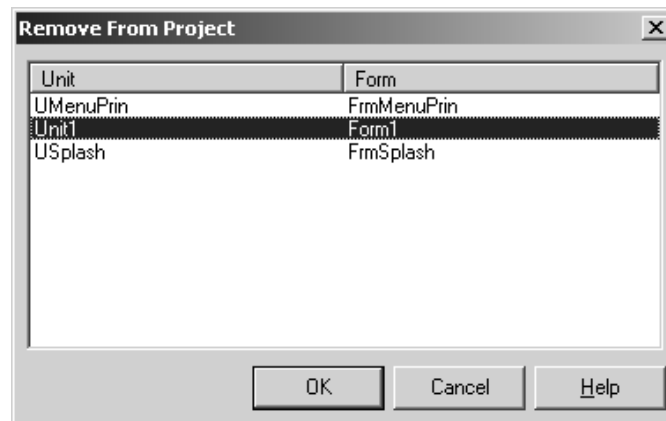
- ✓ **File** -> New -> Form

Como remover uma tela da aplicação?

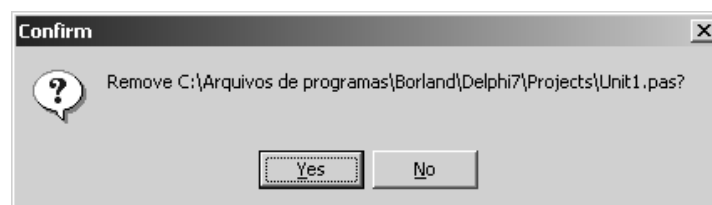
Este tópico deve ser utilizado para cada nova tela a ser inserida no sistema.

Procedimentos a serem executados:

- ✓ **Project** -> *Remove From Project*
- ✓ Escolha a **Unit** da **Form** que será removida e clique em **OK**



- ✓ Responda **Yes** à pergunta da tela de confirmação para remover ou responda **No** para cancelar a operação.



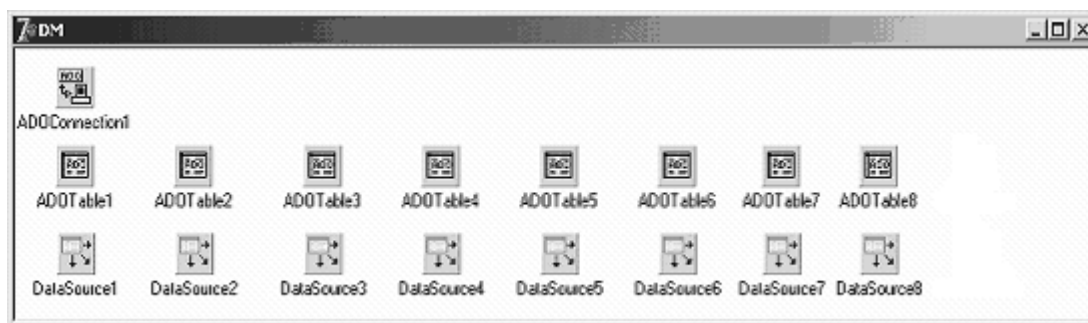
Obs: Caso precise inserir uma **Form/Unit**, já existentes, ao projeto, utilize o comando da opção de menu **Project-> Add to Project** e escolha a unit da form a ser inserida.

Como preparar o banco de dados para ser usado no sistema?

Este tópico tem o propósito de ensinar como preparar o banco de dados e suas tabelas para serem utilizados pelo sistema.

Procedimentos a serem executados:

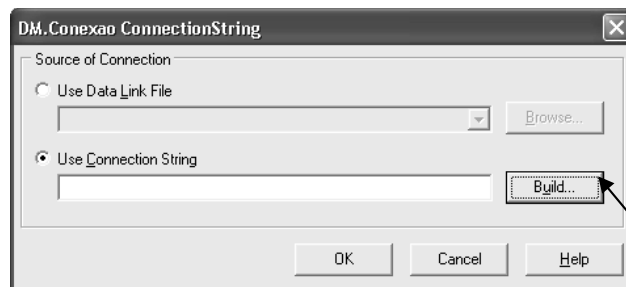
- ✓ **File -> New -> DataModule** (tipo especial de tela para agrupar os objetos de banco de dados). Salve a Unit da tela como **UDM.pas**
- ✓ Altere a propriedade **Name** para **DM** e salve a unit da tela como **UDM**.
- ✓ Selecione a guia **ADO** na paleta de componentes do Delphi e insira um componente **ADOConnection** (conexão com o banco de dados mdb).
- ✓ Insira um componente **ADOTable** (tabela) para cada tabela do banco de dados (para o nosso exemplo **devem ser inseridos 08 ADOTable**).
- ✓ Selecione a guia **DataAccess** na paleta de componentes do Delphi e insira um componente **DataSource** (origem dos dados) para cada tabela do banco de dados (para o nosso exemplo **devem ser inseridos 08 DataSource**).



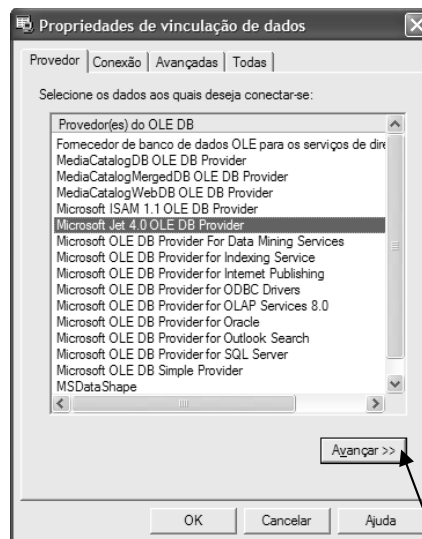
Componentes da tela DataModule (DM) do sistema

- ✓ Altere as propriedades de cada componente conforme as tabelas a seguir:

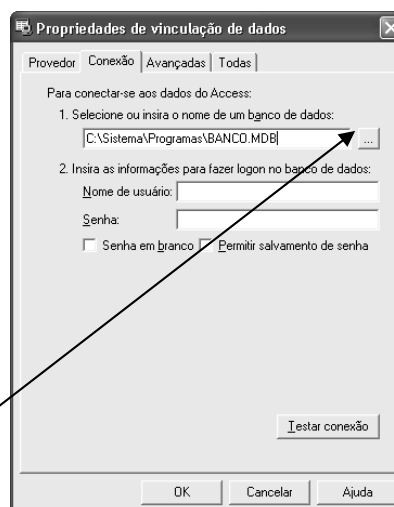
ADOConnection1	Conteúdo	Significado
LoginPrompt	<i>False</i>	Desabilita tela de login (não pede usuário e senha) ao fazer conexão com o banco de dados.
Name	<i>Conexao</i>	Nome do componente AdoConnection
ConnectionString..	<i>Clique em ... e siga as instruções a seguir</i>	String para conexão ao banco de dados



Clique em **Build...**



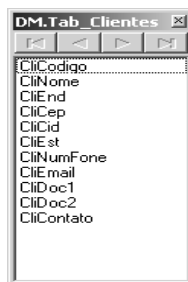
Escolha **Microsoft Jet 40.0 OLE DB Provider** e clique em **Avançar >>**



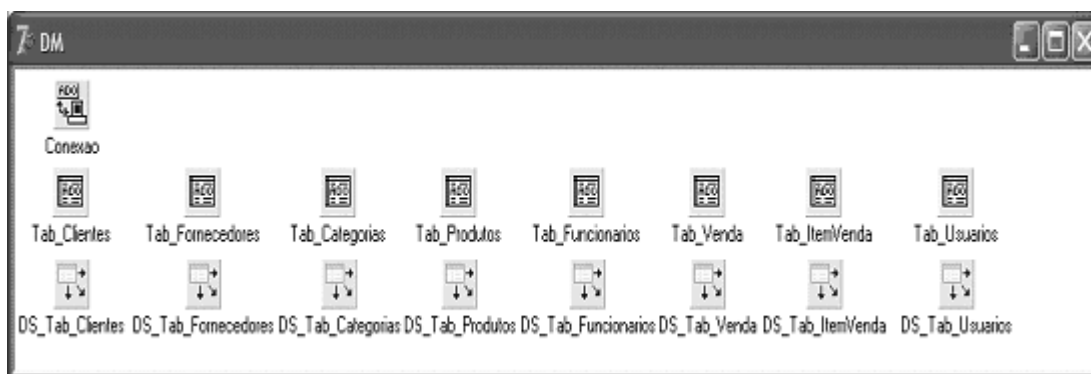
- 1 – Selecione o banco de dados (**C:\Sistema\Programas\Banco.mdb**)
- 2 – Remova o **Nome do usuário** e desmarque **Senha em branco**
- 3 – Clique no botão **Testar Conexão** para testar se a conexão foi bem sucedida.
- 4 – Clique **Ok** nesta e nas próximas, até retornar à tela **DM**.

ADOTable1	Conteúdo	Significado
Connection	<i>Conexao</i>	Conexão ao Banco de dados onde se encontra a tabela desejada.
Name	<i>Tab_Clientes</i>	Nome do componente ADOTable
TableName	<i>Escolha Clientes</i>	Nome da tabela no banco de dados

- ✓ Dê duplo-clique no componente **tab_Clientes**. Tecle **[Ctrl] + [F]** para inserir os campos da tabela. O resultado deverá ficar como o da tela a seguir:



- ✓ Selecione cada campo e altere suas propriedades, quando necessário. Por exemplo, altere a propriedade **DisplayLabel** do campo **CliCodigo** para **Código do Cliente**. Isto fará com que todas as telas que usarem este campo apareça como Código do Cliente em seu rótulo, e não mais **CliCódigo**. Outra propriedade é a **EditMask**, usada com campos do tipo *Texto* e *Data/Hora*, que define a máscara do campo (Ex.: **CliCep** = 99.999-999). Para campos do tipo **Moeda**, altere a propriedade **Currency** para **True**, para que lhe seja aplicado o formato monetário. Quando precisar acessar informações de um campo em uma tabela, use a sintaxe: **DataModule.TabelaCampo.Propriedade**. Por exemplo, para atribuir o conteúdo do campo **CliNome** à variável **Nome**, use **Nome:=Dm.Tab_ClientesCliNome.Value;** (não precisa fazer isso agora).
- ✓ Selecione o componente **DataSource1** e escolha **tab_Clientes** na propriedade **DataSet**. Altere a propriedade **AutoEdit** (edição automática nos dados) para **False**. Altere a propriedade **Name** para **Ds_Tab_Clientes**. Faça o mesmo para as demais tabelas. O resultado final deverá ser como o mostrado na tela a seguir:



Como criar uma tela de login do sistema?

Este tópico tem o propósito de ensinar como criar uma tela de login



Figura 4 – Layout da Tela de Login

Procedimentos a serem executados:

- ✓ **File** -> New -> Form (cria uma nova tela)
- ✓ **File** -> Save All (Salve a Unit como **ULogin** na pasta *Programas*)
- ✓ Altere as propriedades da tela **Form**, conforme tabela a seguir:

Name	FrmLogin	BorderStyle	bsDialog
Height	225	Position	poScreenCenter
Width	400	Caption	Login

- ✓ Acrescente um objeto **Image** e altere suas propriedades:

Height	84	Left	4
Width	113	Top	4
Stretch	True	Picture...	HandShak.bmp

- ✓ Acrescente 01 objeto **GroupBox** e altere suas propriedades:

Caption	Logando no Sistema	Name	GrpBxLogin
Left	4	Top	96
Height	88	Width	382

- ✓ Acrescente 03 **Labels** (**01** à tela e **02** ao GrpBxLogin) e mude as propriedades:

	Label1	Label2	Label 3
Caption	SISTEMA COMERCIAL Versão 1.0	Usuário	Senha
Alignment	taCenter	taLeftJustif y	taLeftJustify
AutoSize	False	True	True
WordWrap	True	False	False
Font...	Comic Sans Ms, 14, Bordô	-	-
Left, Top, Width, Height	140, 4, 208, 50	18, 20,-,-	18, 54,-,-

- ✓ Acrescente ao GroupBox 02 objetos **Edit** e altere suas propriedades:

	Edit1	Edit2
Text	(em branco)	(em branco)
Name	EdtApelido	EdtSenha
PassWordChar	#0	*
CharCase	ecUpperCase	ecLowerCase
Left, Top, Height, Width	64, 19, 21, 177	64, 53, 21, 177

- ✓ Acrescente ao **GroupBox** 02 objetos **BitBtn (Additional)** e altere suas **propriedades**:

	BitBtn1	BitBtn2
Kind	bkOk	bkCancel
Caption	&OK	&Cancelar
Name	BtnOk	BtnCancelar
Left, Top, Height, Width	272, 18, 25, 89	272, 49, 25, 89

- ✓ No evento **OnClick** do botão **BtnCancelar**, digite *Application.Terminate*;
- ✓ No evento **OnClick** do botão **BtnOk**, digite:

```

Dm.tab_Usuarios.Open; // abre a tabela tab_Usuarios
FrmMenuPrin.StatusBar1.Panels[2].Text := ' Usuário:' +
FrmLogin.EdtApelido.Text + ' - ' + Dm.tab_UsuariosUsuDepto.AsString;
If not (Dm.tab_Usuarios.Locate('UsuApelido', FrmLogin.EdtApelido.Text,
[loPartialKey ])) or (Dm.tab_UsuariosUsuSenha.Value <>
FrmLogin.EdtSenha.Text) then
begin
    MessageDlg('Nome ou senha do usuário inválidos.'+#13+#13
    + 'Se você esqueceu sua senha, consulte ' + #13
    + 'o administrador do sistema', mtError, [mbOK], 0);
    EdtSenha.Clear; // limpa o objeto EdtSenha
    EdtSenha.SetFocus; //Ajusta o foco para o objeto EdtSenha
end
else
begin
    FrmLogin.Hide; //Esconde a tela
    FrmMenuPrin.ShowModal; //chama a tela FrmMenuPrin no modo modal
    FrmLogin.Release; //Remove a tela da memória
    FrmLogin := Nil; //Atribui conteúdo nulo para a variável de tela FrmLogin
end;

```

Obs.: Se, ao tentar compilar o programa, for exibido um erro dizendo que **loPartialKey** não foi declarado, tecle **[F1]** e peça ajuda sobre **loPartialKey**. Quando o Delphi exibir a ajuda, anote o nome da **Unit** à qual pertence este parâmetro e inclua o nome da mesma (no caso, **DB**) na cláusula **Uses** do formulário em questão.

Como alterar a Main Form do sistema?

Este tópico tem o propósito de ensinar como alterar a Main Form do sistema, ou seja, fazer com que seja mudada a tela que é apresentada em primeiro lugar quando o sistema é executado.

Se você tentou executar o sistema até este ponto, percebeu que a tela de login não apareceu. Uma das maneiras de fazer com que isto aconteça, é alterar a **Main Form** para a tela **FrmLogin**. Para isto siga os procedimentos abaixo:

Procedimentos a serem executados:

- ✓ Selecione a opção de menu **Project -> Options**

- ✓ Selecione a guia **Forms** e altere a opção **Main Form** para a tela desejada, em nosso exemplo escolha **FrmLogin** e clique no botão **OK**.
- ✓ Tecle **[F9]** para executar a aplicação

Como criar uma tela padrão?

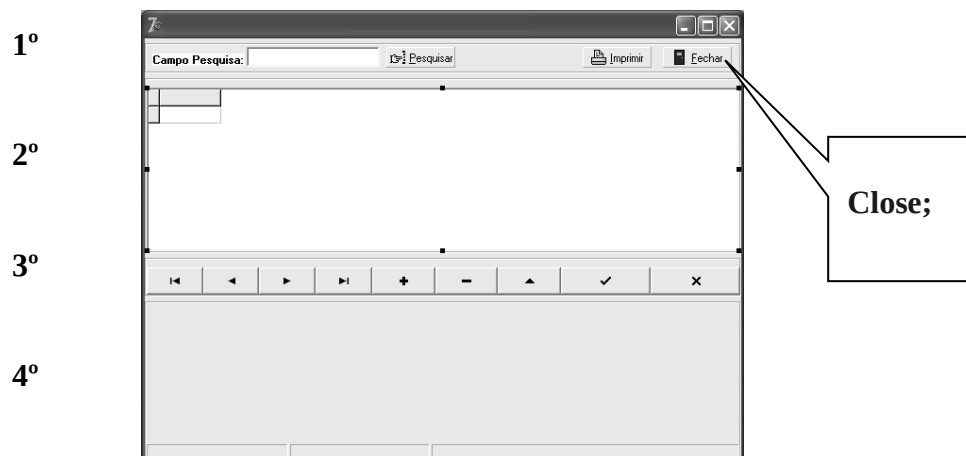
O uso de uma tela padrão além de deixar as telas do sistema com um layout padronizado, agiliza o processo de criação das mesmas, pois, todos os objetos, propriedades e eventos, comuns a todas as telas, não são criados novamente, e sim, herdados da tela padrão, através do conceito de herança de formulários.

Procedimentos a serem executados:

- ✓ Crie uma nova form (**File -> New -> Form**), altere sua propriedade **Name** para **FrmCadPadrao**. Altere as demais propriedades, conforme tabela vista na **página 04** desta apostila. Salve a unit como **UCadPadrao**.
- ✓ Observe o layout da tela na figura a seguir e acrescente os objetos, na ordem em que são citados na lista de objetos abaixo:

Lista de objetos: (Deixe os tamanhos dos **GroupBox** proporcionais aos da figura)

- ✓ Acrescente 04 **GroupBox** (guia *Standard*) com **Align = alTop**
- ✓ 1º **GroupBox** com 01 **Label**, 01 **Edit** (**Name=ValorCampo**), 03 **SpeedButton**.
- ✓ 2º **GroupBox** com 01 **DBGrid** (**Data Controls**) com **Align = alClient**.
- ✓ 3º **GroupBox** com 03 **DBNavigator** (**Data Controls**) com **Align = alLeft**, um ao lado do outro. Use a propriedade **+VisibleButtons** para **exibir/ocultar** os botões do **DBNavigator**, pois o primeiro **DBNavigator** deverá conter apenas os botões **nbFirst**, **nbPrior**, **nbNext**, **nbLast**, o segundo deverá conter apenas os botões **nbInsert**, **nbDelete** e **nbEdit** e o terceiro deverá conter apenas os botões **nbPost** e **nbCancel**.
- ✓ 4º **GroupBox** vazio.
- ✓ 01 **StatusBar** (Win32) com 03 painéis com **Width = 150**.

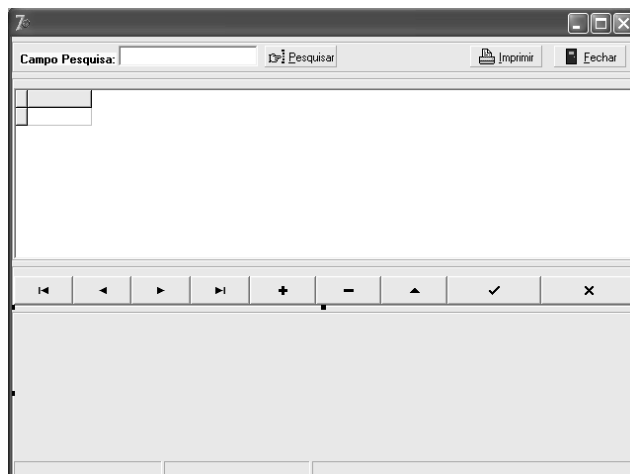


Como utilizar uma tela padrão do sistema?

Utilize este procedimento sempre que for criar uma tela baseada na tela padrão do sistema. Em nosso exemplo será utilizada uma tela padrão para criação das telas do menu de Cadastros.

Procedimentos a serem executados:

- ✓ **File** -> New -> Other
- ✓ Selecione a guia que tenha o nome do sistema atual (**Sistema**)
- ✓ Selecione o nome da tela padrão (**FrmCadPadrao**, em nosso exemplo)
- ✓ Altere a propriedade **Caption** para *Cadastro de Clientes*
- ✓ Altere a propriedade **Name** para **FrmCadCliente**
- ✓ Salve a tela como **UCadCliente**
- ✓ Ajuste as propriedades e eventos dos objetos que compõe a tela. **Obs:** Os componentes **DBGrid** e **DBNavigator** serão explanados no próximo tópico.



No evento **OnShow** da tela digite: **Dm.Tab_Clientes.Open;** // abre a tabela de clientes



No evento **OnClose** da tela digite: **Dm.Tab_Clientes.Close;** // fecha a tabela de clientes

OBS: Repita a execução dos tópicos das **páginas 23 – 28** para criar as demais telas do menu Cadastros (*Clientes, Fornecedores, Categoria, Itens (Produtos), Funcionários*) e do menu Movimentos (*Vendas e Itens da Venda*), fazendo as devidas adaptações, de acordo com a tabela utilizada, pois nos exemplos estão sendo utilizadas apenas a tabela de **Clientes**. A tela Itens da Venda deverá ser chamada por um botão na tela de Vendas.

Como utilizar um DBGrid?

Utilize este procedimento para apresentar os dados de um banco de dados em formato de grade, em linhas e colunas.

Procedimentos a serem executados:

- ✓ **File->Use Unit** e escolha a tela onde se encontram as tabelas. Selecione o componente **DBGrid**, e altere a propriedade **DataSource** (origem dos dados) de acordo com a tabela a ser apresentada na grade. Escolha para este exemplo **Ds_Tab_Clientes**.
- ✓ Dê um duplo-clique sobre o **DBGrid** e clique no botão *Add all Fields* 
- ✓ Selecione os campos que não farão parte da grade e clique no botão *Delete Selected*  para removê-los. Caso a grade faça parte de uma tela padrão, só será possível remover o campo através da janela **Object Treeview**.

Para deixar o DBGrid Zebrado:

No evento **OnDrawColumnCell** do **DBGrid** digite os seguintes comandos:

```
If Odd(dm.tab_Clientes.RecNo) and (dm.tab_Clientes.State <> dsInsert) then
begin //Lembre-se de colocar a unit DB na cláusula uses na unit da tela.
  DBGrid1.Canvas.Brush.Color := clMoneyGreen; // muda a cor do pincel
  DBGrid1.Canvas.FillRect(Rect); // Preenche o fundo com a cor especificada
  DBGrid1.DefaultDrawDataCell(Rect,Column.Field,State);// desenha as células da grade
end;
```

Para ordenar os dados da grade ao clicar no título do campo: a ordenação será feita através da propriedade de tabela **IndexFieldNames**, que define o nome do índice para a classificação, conforme o campo clicado na grade (**Column.FieldName**)

- ✓ Crie **na tela padrão** uma variável global chamada **Ascendente** do tipo **Boolean** para que todas as telas de cadastro passem a utilizá-la.
- ✓ No evento **OnShow**, da **tela padrão**, digite: **Ascendente := False;**
- ✓ Em **FrmCadCliente**, no evento **OnTitleClick** do **DBGrid** digite:

```
Ascendente:= not Ascendente ;
If Ascendente then
  Dm.tab_Clientes.IndexFieldNames := Column.FieldName + ' ASC'
else
  Dm.tab_Clientes.IndexFieldNames := Column.FieldName + ' DESC';
```

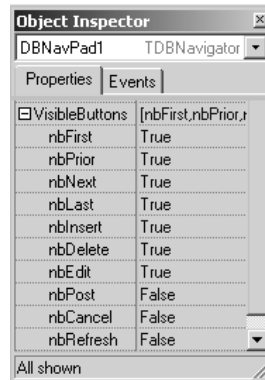
OBS: Não digite os comandos na unit da *tela padrão*, **somente** na de cadastro.

Como utilizar um DBNavigator?

Utilize este procedimento para navegar entre os registros da tabela.











Procedimentos a serem executados:

- ✓ Selecione os componentes **DBNavigator** e altere a propriedade **DataSource** (origem dos dados) de acordo com a tabela a ser utilizada na tela. No exemplo anterior foi usado o datasource **Ds_Tab_Clientes**.
- ✓ Para **exibir / ocultar** qualquer botão utilize a propriedade **VisibleButtons**:



- ✓ **VisibleButtons = True** exibe o botão. **VisibleButtons = False** oculta.

Significado dos botões do DBNavigator e seus comandos equivalentes: utilize os comandos indicados abaixo, caso queira utilizar botões comuns ao invés do DBNavigator.

	= equivale ao comando <Tabela>.First , ou seja, vai para o primeiro registro.
	= equivale ao comando <Tabela>.Prior , ou seja, vai para o registro anterior.
	= equivale ao comando <Tabela>.Next , ou seja, vai para o próximo registro.
	= equivale ao comando <Tabela>.Last , ou seja, vai para o último registro.
	= equivale ao comando <Tabela>.Insert , ou seja, insere um registro em branco.
	= equivale ao comando <Tabela>.Delete , ou seja, apaga um registro.
	= equivale ao comando <Tabela>.Edit , ou seja, edita um registro para alteração.
	= equivale ao comando <Tabela>.Post , ou seja, grava dados no registro.
	= equivale ao comando <Tabela>.Cancel , ou seja, cancela alteração no registro.
	= equivale ao comando <Tabela>.Refresh , ou seja, atualiza dados nos controles.

<Tabela> deve ser o nome da tabela na qual deseja-se manipular os registros.

Como inserir campos de uma tabela na tela?

Execute este tópico sempre que for apresentar campos da tabela na tela, utilizando controles *Data Controls* (*DBEdit*, *DBComboBox*, etc).

- ✓ **Automático:** Selecione o **datamodule** onde estão os componentes de banco de dados. Logo em seguida dê um **duplo-clique** sobre o objeto **AdoTable** correspondente à tabela que deseja pegar os campos. Selecione todos os campos que deseja incluir na tela e arraste-os para dentro da tela em que eles ficarão. Como exemplo, selecione a tela datamodule **DM** e o objeto AdoTable **tab_Clientes**, selecione os seus campos e arraste-os para a tela **FrmCadCliente**. Neste caso são utilizados controles DBEdit.
- ✓ **Manual:** Selecione o objeto **Data Control** desejado, **DBEdit**, por exemplo, e altere as propriedades **DataSource** (origem dos dados) e **DataField** (campo) de acordo com a informação a ser exibida ou editada da tabela.

A imagem mostra a interface de design de uma aplicação em Delphi, especificamente a tela 'Cadastro de Clientes' no modo Designer. A janela tem uma barra de título com o nome 'Cadastro de Clientes'. No topo, há uma barra de ferramentas com botões para 'Pesquisar', 'Imprimir' e 'Fechar'. Abaixo, há uma grade de dados com cabeçalhos 'Nome', 'Cidade', 'Endereço' e 'Fone'. Na parte inferior, há campos de formulário rotulados com 'DBE dnt1' a 'DBE dnt10', organizados em seções para 'Nome', 'Endereço', 'Cidade', 'Cep', 'UF', 'Fone', 'Email', 'RG', 'CPF' e 'Contato'.

Tela Cadastro de Clientes no modo Designer

Como chamar uma tela no Delphi?

Execute este tópico sempre que desejar chamar uma tela através de outra tela do sistema.

Procedimentos a serem executados:

- ✓ Selecione o evento do objeto que será usado para chamar a tela (**OnClick** de um botão, ou **OnExecute** de um **ActionList**, por exemplo) e digite um dos dois comandos conforme sintaxe a seguir:
 - o Para criar uma tela **Modal** (não permite acessar as demais telas da aplicação enquanto a tela modal não for fechada):
`Nome_Form.ShowModal;` Ex.: `FrmCadCliente.ShowModal;`
 - o Para criar uma tela **não Modal**: (permite acessar as demais telas da aplicação mesmo se a tela não-modal estiver aberta):
`Nome_Form.Show;` Ex.: `FrmCadCliente.Show;`
- ✓ Para o nosso primeiro exemplo dê um duplo-clique na **ActionList1** da tela **FrmMenuPrin** e escolha a ação **ActCliente**
- ✓ Dê um **duplo-clique** no evento **OnExecute** da mesma e digite **FrmCadCliente.ShowModal;** para chamar a tela e cadastro de clientes.

Nome	Cidade	Endereço	Fone
Carlos	Campinas	R. Alecrins, 18	(19)3278-1258
Paulo	Campinas	R. Votorantim, 315	() -
Ana	Rio de Janeiro	Av. Moraes Sales, 182	() -
Roberto	São Paulo	R. Alamedas, 75	() -
Daniela	Campinas	Av. Pio XII, 215	() -
Luis	Salvador	R. José Paulino, 15	() -
Eduardo	São Paulo	R. Quinze, 2452	() -
Pedro	Belém	R. Lacerda Franco, 85	() -

Nome	Endereço	Cidade
Carlos	R. Alecrins, 18	Campinas

Cep	UF	Fone	Email
13100-000	SP	(19)3278-1258	Carlos123@hotmail.com

RG	CPF	Contato
123.456.678-9	11.222.333-60	Carlos

Tela Cadastro de Clientes após ser chamada

OBS: Prefira utilizar os comandos da maneira como serão mostrados no próximo tópico para chamar as demais telas do sistema. Isso liberará memória do sistema.

Como criar uma tela manualmente na aplicação?

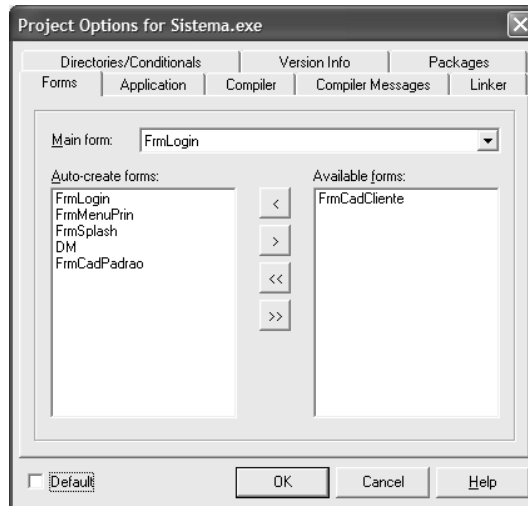
Este tópico vai ensinar os procedimentos para criar uma tela manualmente. Por padrão, o Delphi cria automaticamente seus formulários, mas, dependendo da quantidade deles, a memória pode ficar sobrecarregada, comprometendo o desempenho do sistema. Esse problema é solucionado criando-se manualmente

cada tela somente quando a mesma for utilizada, e liberando-a da memória logo em seguida, após o seu uso.

Procedimentos a serem executados:

Obs. Neste exemplo utilize a tela **FrmCadCliente** e a ação **ActCliente**.

- ✓ Selecione a opção de menu **Project -> Options**
- ✓ Clique na guia **Forms** e selecione a tela a ser criada manualmente em **Auto-Create Forms** (criação automática)
- ✓ Clique no botão [**>**] para passá-la para **Available Forms** (criação manual)
- ✓ Clique no botão [**OK**].



- ✓ Selecione no **ActionList1** o evento **OnExecute** da ação que será usada para chamar a tela e digite os comandos conforme exemplo a seguir.

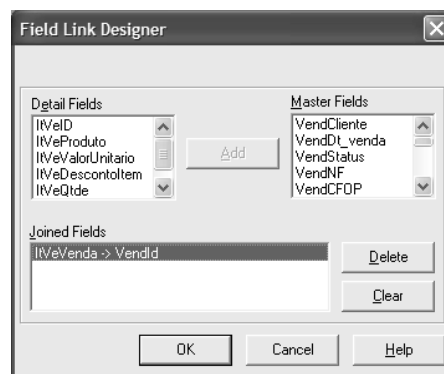
```
FrmCadCliente:= TFrmCadCliente.Create(Self); //criação manual  
FrmCadCliente.ShowModal; //exibe a tela no modo modal  
FrmCadCliente.Release; //libera a tela da memória  
FrmCadCliente:= nil; //atribui o conteúdo nulo para a variável FrmCadCliente
```

Como criar relacionamento Master/Detail na aplicação?

Este tópico vai ensinar os procedimentos para criar um relacionamento **Master/Detail** na aplicação. Como exemplo utilizaremos as tabelas **Tab_Venda** e **Tab_ItemVenda**, onde, **Tab_Venda** será **Master**, e **Tab_ItemVenda** será **Detail**. Em um relacionamento **Master/Detail**, para cada registro da tabela **Master**, são relacionados apenas os registros da tabela **Detail** que tiverem os mesmos valores de chaves primária e estrangeira, respectivamente (**VendID = ItVeVenda**).

Procedimentos a serem executados:

- ✓ Selecione a tela **Dm** ([Shift] + [F12])
- ✓ Selecione a propriedade **MasterSource** da tabela detail **Tab_ItemVenda**, e escolha o nome do **DataSource** da tabela master, **Ds_Tab_Venda**.
- ✓ Selecione a propriedade **MasterFields** (*chave primária da tabela master*).na tabela **Tab_ItemVenda** e digite **VendID**, *que é o campo que fará a ligação entre as tabelas*.
- ✓ Execute a aplicação e chame a tela de **vendas**. A partir de agora, quando a tela de **itens da venda** for chamada, só serão exibidos os registros dos itens correspondentes ao registro da venda selecionada.
- ✓ Dê **duplo-clique** na propriedade **MasterField** para ver a janela de designer do relacionamento. O relacionamento também poderia ter sido feito selecionando-se as chaves primária e estrangeira e clicando no botão **Add**. Faça isto apenas se você não se lembrar do nome da chave primária.



Janela Designer do Relacionamento

Como utilizar componentes do tipo LookUp?

Este tópico vai ensinar os procedimentos para utilizar componentes do tipo **LookUp** que atribuem a um determinado **DataSet**, valores provenientes de outro **DataSet**, através de um relacionamento entre seus campos-chave.

Procedimentos a serem executados:

- ✓ Insira um objeto **DBLookupComboBox**, ou **DBLookupListBox** da guia **DataControls** e altere as seguintes propriedades:
 - **DataSource** = DataSource da tabela onde há chave estrangeira no relacionamento um-para-muitos.
 - **DataField** = nome do campo que é chave estrangeira num relacionamento um-para-muitos.
 - **ListSource** = DataSource da tabela principal do relacionamento e que possui o valor do campo que será exibido na lista do componente **DBLookup**.
 - **ListField** = nome do campo cujo conteúdo será exibido para o usuário na lista do componente **DBLookup** escolhido.
 - **KeyField** = nome do campo chave-primária da tabela principal do relacionamento.
- ✓ Considere a seguinte situação: a tabela **Tab_ItemVenda** possui o campo **ItVeProduto** que é chave estrangeira e relaciona-se com a tabela **Tab_Produtos** através da chave primária **ProdID**. Como o campo **ItVeProduto** armazena apenas o valor do código do produto, fica difícil para o usuário saber qual produto está sendo utilizado. Utilizaremos um dos componentes **DBLookup** para exibir o nome do produto para o usuário.
- ✓ Insira um componente **DBLookupComboBox** da guia **DataControls** na tela **FrmItemVenda** e altere as seguintes propriedades:

- **DataSource** = Ds_Tab_ItemVenda
 - **DataField** = ItVeProduto
 - **ListSource** = Ds_Tab_Produtos.
 - **ListField** = ProdNome
 - **KeyField** = ProdID
- ✓ Siga o mesmo procedimento para o campo **Cliente** na tela **FrmVenda**. Lembre-se de abrir a tabela **Tab_Produtos** e **Tab_Clientes** (no evento **OnShow** das telas).

Como criar consultas em tabelas do banco de dados?

Este tópico vai ensinar os procedimentos para criar uma consulta parcial a uma tabela e uma de chave completa. A Unit do Delphi utilizada deve ser a **DB**.

Procedimentos a serem executados:

Para criar uma **consulta aproximada** a uma tabela de banco de dados, ou seja, uma pesquisa que procura parte do campo desejado, siga os passos:

- ✓ Selecione o evento **OnChange** do componente **TEdit** desejado (neste caso **ValorCampo** na tela de Clientes) e digite o comando **Locate**, como abaixo:

```
Dm.tab_Clientes.Locate( 'CliNome',ValorCampo.Text, [loCaseInsensitive, loPartialKey] );
```

Onde,

*Dm é o nome da tela de **DataModule**,*

*tab_Clientes é o nome da componente **AdoTable** da tabela,*

CliNome é o nome do campo na tabela a ser pesquisado,

*ValorCampo.Text é o objeto **TEdit** onde será digitado o conteúdo a ser pesquisado.*

loCaseInsensitive não faz distinção entre letra maiúscula ou minúscula

loPartialKey pesquisa por parte do campo

Para realizar uma **consulta pela chave completa**, ou seja, o conteúdo só será encontrado depois de digitado todo o conteúdo do campo, utilize o mesmo comando, remova a palavra **loPartialKey** e coloque o comando dentro de uma estrutura de decisão **If**, conforme o exemplo a seguir:

```
If not Dm.tab_Clientes.Locate( 'CliNome',ValorCampo.Text, [loCaseInsensitive]) then  
    MessageDlg('Cliente não cadastrado!', mtError, [mbOk], 0);
```

Digite o comando acima no evento **OnClick** do botão **Pesquisar**, por exemplo. Neste caso, lembre-se de colocar comentário (//) no comando do evento **OnChange** do **TEdit ValorCampo**, utilizado anteriormente. Para fazer uma consulta com dois campos utilize a sintaxe: **Tabela.Locate('Campo1;Campo2', VarArrayOf([Conteúdo1, Conteúdo2]), [loPartialKey]);**

Como utilizar filtro em uma tabela?

Este tópico vai ensinar os procedimentos para utilizar filtros em tabelas de banco de dados para selecionar apenas os registros que o usuário deseja.

Procedimentos a serem executados:

- ✓ **Sintaxe para utilizar filtro:**

With <DataSet> **do**

begin

Filtered := false; // desliga o filtro

Filter := 'digite o filtro desejado entre apóstrofes';

Filtered := true; // ativa o filtro de acordo com critério estabelecido em Filter

End;

- ✓ Para exibir seus clientes, filtrados por cidade, crie a procedure abaixo na unit da tela **FrmCadCliente**:

```
procedure TFrmCadCliente.FiltraCliente;
Var Cidade : String;
      Ok : Boolean;
begin
  Cidade := 'ALL';
  Ok := InputQuery('Filtra Clientes por Cidade', 'Digite o nome da cidade:
(ALL remove o filtro)',Cidade);
  If Ok then
    With dm.Tab_Clientes do
      begin
        Filtered := false; // Desativa o filtro
        if Cidade <> 'ALL' then
          begin // A função QuotedStr coloca apóstrofes no string.
            Filter := 'CliCid = ' + QuotedStr(Cidade); // monta o filtro
            Filtered := true; // Ativa o filtro
          end;
        end;
      end;
    end;
  end;
```

- ✓ Crie um botão que será usado para filtrar os clientes por cidade e, no evento **OnClick** do mesmo, chame a procedure **FiltraCliente**.

Como usar a linguagem SQL no Delphi?

Este tópico vai ensinar os procedimentos para a utilização dos comandos da linguagem **SQL** (Structure Query Language) no Delphi. Através desta linguagem você pode criar e alterar tabelas, inserir, alterar, apagar registros, bem como fazer pesquisas simples e complexas em tabelas de bancos de dados.

Procedimentos a serem executados:

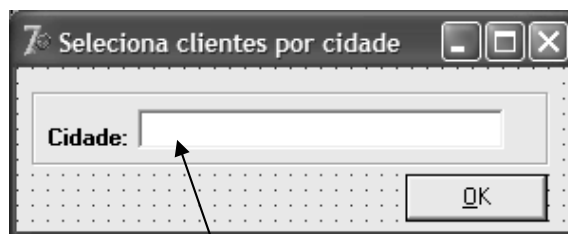
- ✓ Utilize algum componente para banco de dados que aceite instruções **SQL**. Em nosso exemplo, como estamos utilizando componentes **ADO**, selecione na guia **ADO** da paleta de componentes, o componente **ADOQuery**, inserindo-o na tela **FrmCadCliente**.
- ✓ Altere a propriedade **Name** para **QueryClientes**
- ✓ Defina a conexão do banco de dados que contém as tabelas desejadas. Para isto altere a propriedade **Connection**, escolhendo a conexão. Em nosso exemplo, escolha **Conexao**
- ✓ Clique na propriedade **SQL...** e digite o comando: **Select * From Clientes Where CliCid = :Cidade Order By CliNome** para selecionar todos os registros da tabela de Clientes da cidade fornecida como parâmetro de pesquisa pelo usuário através da aplicação, em ordem pelo nome do cliente. Em nosso exemplo serão selecionados todos os campos de cada registro (*), mas, se preferir selecionar apenas alguns campos, basta apenas digitar no lugar do asterisco o nome de cada campo separado por vírgula.
- ✓ Dê um duplo-clique no objeto **QueryClientes** e selecione todos os campos teclando **[Ctrl] + [F]**.

Obs:A sintaxe para utilização de um objeto **AdoQuery** com os comandos SQL é a seguinte:

```
With <nome do ADOQuery> do
Begin
  Close;
  Parameters[nº de ordem do parâmetro].Value := <valor>;
  Open;
  // <outros comandos do Delphi, se houver>
  Close;
End;
```

* Logo a seguir será mostrado um exemplo.

- ✓ Crie a tela abaixo (**name = FrmPesCliCid**) para ser chamada no evento **OnClick** do botão **Imprimir** (*FrmPesCliCid.ShowModal;*)



- ✓ No evento OnClick do botão **OK** acima digite:

```
With FrmCadCliente.QueryClientes do
Begin
  Close; // fecha a query
  Parameters[0].Value := EdtCidade.Text; //define parâmetro de pesquisa
  Open; //abre a query e executa os comandos SQL
End;
FrmPesCliCid.Close;
```

- ✓ Utilize os comandos abaixo para permitir apenas digitação de letras e **backspace**. Acesse o evento **OnKeyPress** do objeto **EdtCidade** e digite:

```
// se última tecla for letra, espaço ou Backspace
If not (Key in ['A'..'Z','a'..'z', #32, #8]) then
  Key := #0; // ignora a última tecla digitada
```

Obs: Caso você utilize os comandos **Insert**, **Delete**, **Update**, em sua query, execute o método **ExecSql**, ao invés de **Open**.

Dica: Procure se aprofundar no estudo da linguagem SQL, principalmente seus principais comandos como Select, Update, Insert, Delete, Create Table, bem como técnicas de Join, agrupamentos, etc.

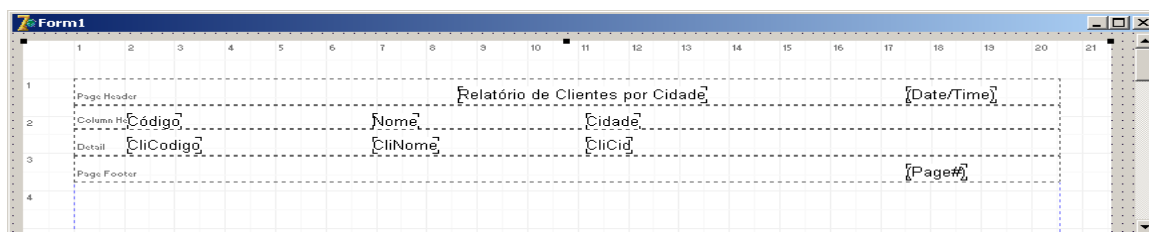
Caso queira, você pode criar uma tela **Datamodule** para organizar seus objetos **AdoQuery**, evitando que os mesmos fiquem espalhados pelas telas da aplicação.

Como criar um relatório com o QuickReport?

Este tópico vai ensinar os procedimentos para a criação de um relatório utilizando o QuickReport. **Obs.:** Para instalar o QuickReport no Delphi 7, escolha **Components ->Install Packages...**, clique no botão **Add** e selecione o arquivo **C:\Arquivos de programas\Borland\Delphi7\Bin\dcqlrt70.bpl**

Procedimentos a serem executados:

- ✓ Clique em **File->New->Form** para criar a tela do relatório.
- ✓ Mude o **name** para **FrmRelCliCid** e salve a unit como **UQRRelCliCid**.
- ✓ Clique em **File->Use Unit** e selecione a unit **UCadCliente**
- ✓ Selecione a guia **QReport** e insira um componente **QuickRep**
- ✓ Escolha na propriedade **DataSet** a query **FrmCadCliente.QueryClientes**
- ✓ Dê duplo-clique no componente e defina as propriedades do relatório.
Marque todas as bandas, **exceto Title e Summary**.
- ✓ Acrescente **04** componentes **QRLabel** (rótulo em relatório) e altere respectivamente suas propriedades **Caption** para : *Relatório de Clientes por Cidade, Código, Nome, Cidade*.
- ✓ Acrescente **03** componentes **QRDBText** (conteúdo de campos de banco de dados a serem exibidos no relatório) e altere a propriedade **DataSet (tabela)** para **FrmCadCliente.QueryClientes**, e a propriedade **DataField** respectivamente para **CliCodigo**, **CliNome** e **CliCid**.
- ✓ Acrescente **02** componentes **QRSysData** e altere suas propriedades **Data** para **qrsDateTime**, **qrsPageNumber** . Sua tela deverá ficar assim:






- ✓ Para testar o relatório **clique com o botão direito do mouse** sobre o **QuickRep** e escolha **Preview**. Salve o relatório como **URelCliCid**.
- ✓ Para chamar o relatório digite os comandos abaixo no botão **Imprimir** da tela **FrmCadCliente**:

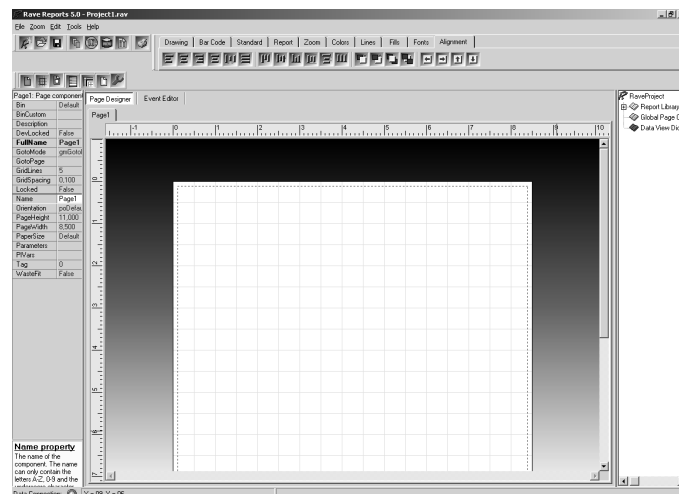
```
FrmPesCliCid.ShowModal;  
FrmRelCliCid.QuickRep1.Preview;
```

Como criar um relatório no Rave Reports?

Este tópico vai ensinar os procedimentos para a criação de um relatório utilizando o Rave Reports.

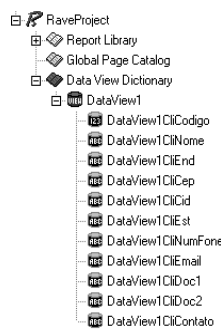
Procedimentos a serem executados:

- ✓ Selecione a tela **FrmCadCliente**.
- ✓ Acesse a guia **Rave** e insira um componente **RvDataSetConnection** , que é responsável pela conexão com o banco de dados. Mude a propriedade **Name** para **RvDtCnCliente**. Configure a propriedade **DataSet** selecionando o nome da tabela ou query. Neste exemplo, escolha a query **QueryClientes**.
- ✓ Insira um componente **RvSystem** , que será responsável pela exibição do relatório, na tela ou na impressora. Mude **Name** para **RvSysCliente**.
- ✓ Insira um componente **RvProject** , que será responsável pelo gerenciamento do relatório. Mude **Name** para **RvPrjCliente**. Ajuste as propriedades **Engine** para **RvSysCliente**.
- ✓ Dê um duplo clique no componente **RvPrjCliente**, ou, escolha **Tools->Rave Designer**, para abrir o **Rave** em sua área de Design.



Ambiente de Design do Rave Report

- ✓ Acesse **File-> New** para criar um novo relatório no Rave
- ✓ Acesse **File -> New Data Object** e escolha **Direct Data View**. Selecione a conexão **RvDtCnCliente** na área **Active Data Connection** e clique no botão **Finish**.



Tree Panel após selecionarmos RvDtCnCliente

- ✓ Vá até a guia **Report** e adicione um componente **Region Component**, onde devem ficar obrigatoriamente todas bandas do relatório. Use a guia **zoom** para ajudá-lo a visualizar melhor e faça com que o **Region** ocupe toda a área de impressão (área serrilhada de cor vermelha).
- ✓ Adicione um componente **Band** (guia Report) ao **Region**, clique na propriedade **BandStyle** e marque as opções **Body Header, First (1) e New Page (P)**. Esta será a banda de cabeçalho do relatório. Adicione dentro dela 04 **Text Component** (guia Standard) e altere a propriedade **Text** dos mesmos para **Relatório de Clientes, Nome, Endereço, Cidade**. Use a propriedade **Font** e ajuste a seu critério, tamanho, estilo, dentre outros atributos da fonte.
- ✓ Adicione um componente **DataBand** (guia Report), e ligue a propriedade **DataView** ao **DataView1** onde o Rave buscará os dados do relatório.
- ✓ Adicione os campos do relatório, teclando **[CTRL]** enquanto arrasta cada campo do **Tree Panel** para dentro de **DataBand1**. Faça isto para os campos **Cli_Nome, Cli_End** e **Cli_Cid**.
- ✓ Selecione o **Band1** e ligue a propriedade **ControllerBand** ao **DataBand1** que será a banda de controle do cabeçalho do relatório.

▼ Region1: Band1 (BGRDrGb 1PC)		
Nome	Endereço	Cidade
◆ Region1: DataBand1 (Master 1PC)		
[CliNome]	[CliEnd]	[CliCid]


Visão final de como deve ficar os objetos no Rave

- ✓ Teste o relatório com **[F9]**. Feche a tela de visualização após o teste.
- ✓ Salve o projeto como **RelClientes.rav**, volte ao Delphi e ajuste a propriedade **Project File** para **RelClientes.rav**.
- ✓ No evento **OnClick** do botão **Imprimir**, da tela **FrmCadCliente**, substitua o comando **FrmRelCliCid.QuickRep1.preview** do **QuickReport** pelo comando **RvPrjCliente.Execute;** do **Rave Reports**.

Como criar um relatório com a linguagem Rave Reports?

Este tópico vai ensinar os procedimentos para a criação de um relatório com a linguagem do Rave Reports.

Procedimentos a serem executados:

- ✓ Acesse a guia **Rave** e insira um componente **RvSystem** , que será responsável pela exibição do relatório, na tela ou na impressora. Mude o **Name** para **RvSysCliente**.
- ✓ Substitua o comando **RvPrjCliente.Execute**; no evento **OnClick** do botão **Imprimir** da tela **FrmCadCliente** pelos comandos a seguir:

```
If QueryClientes.RecordCount > 0 then // se qtde de registros = 0
begin
  With RvSysCliente do
    begin
      SystemPrinter.Units      := unCM; //unidades em centímetro
      SystemPrinter.UnitsFactor := 2.54; // Fator para conversão polegada
      SystemPrinter.Orientation := poPortrait; // Modo Retrato
      SystemPreview.RulerType   := rtBothCm; // medidas da régua em cm
      //SystemSetups := SystemSetups - [ssAllowSetup]; remove tela de setup
      SystemPreview.FormState := wsMaximized; // tela relatório maximizada
      Execute; // executa o relatório
    end;
  end
else
  ShowMessage ('Nenhum Registro Encontrado.');
```

- ✓ No evento **OnBeforePrint** do objeto **RvSysCliente** digite:

```
With RvSysCliente .BaseReport do
begin
  FontName := 'Arial'; //define a fonte como Arial
  FontSize := 11; //define o tamanho da fonte para 11
  Bold := false; // desabilita o estilo de fonte negrito
  SetPaperSize(DMPAPER_A4, 0, 0); //ajusta tamanho do papel
end;
```

- ✓ No evento **OnPrint** do objeto **RvSysCliente** digite:

With RvSysCliente.BaseReport do

begin

QueryClientes.First; // vai para o primeiro registro da query

While (not QueryClientes.Eof) do // enquanto não for fim de arquivo

begin

Lin := 1; // Declarar variável **Lin** com o tipo **Real** e escopo **global**

Cab_RelCadCliente; // chama a procedure cabeçalho do relatório

While (not QueryClientes.Eof) and (Lin <= 29) do

begin

Det_RelCadCliente; // chama a procedure detalhe do relatório

QueryClientes.Next; //vai para o próximo registro

end;

if not QueryClientes.Eof then

NewPage; // insere uma nova página ao relatório

end;

Lin := Lin - 0.2;

MoveTo(0.7,Lin); //move o cursor para a coluna e linha indicados

LineTo(20.5,Lin); //traça uma linha até posição *coluna x linha* indicada.

Lin := Lin + 0.5;

end;

- ✓ Crie as **procedures** a seguir para imprimir cada banda do relatório:

procedure TFrmCadCliente.Det_RelCadCliente; // Detalhe do relatório

begin

with RvSysCliente .BaseReport do

begin

Gotoxy(0.7,Lin); //tabula coluna e linha no relatório

Print (QueryClientesCliNome.AsString); // Imprime Nome do cliente

Gotoxy(6.5,Lin);

Print (QueryClientesCliNumFone.AsString);

Gotoxy(10,Lin);

Print(QueryClientesCliCid.AsString);

Lin := Lin + 0.5;

end;

end;

```

procedure TFrmCadCliente.Cab_RelCadCliente; // Cabeçalho
begin
  with RvSysCliente .BaseReport do
    begin
      Gotoxy(0.7,Lin); // Declarar variable Lin com o tipo Real e escopo global
      Print('Data:' + FormatDateTime('dd/mm/yyyy " – Hora:" hh:mm:ss',now));
      Bold := False;
      Gotoxy(18,Lin); // tabula coluna e linha de impressão
      Print('Pág.:'+Macro(midCurrentPage)+'/'+Macro(midTotalPages));
      Bold := True; //define estilo da fonte para negrito
      Gotoxy(08,Lin);
      Print('Empresa XYZ'); // Escreve dados no relatório
      Lin := Lin + 0.5;
      Gotoxy(08,Lin);
      Print('Relatório de Clientes');
      Lin := Lin + 0.5;
      Gotoxy(0.7, Lin);
      Lin := Lin + 0.5;
      Gotoxy(6,Lin);
      MoveTo(0.7,Lin);
      LineTo(20.5,Lin);
      Lin := Lin + 0.5;
      Gotoxy(0.7,Lin);
      Print ('Nome');
      Gotoxy(6.5,Lin);
      Print ('Fone');
      Gotoxy(10,Lin);
      Print('Cidade');
      Lin := Lin + 0.2;
      MoveTo(0.7,Lin);
      LineTo(20.5,Lin);
      Lin := Lin + 0.5;
      Bold := False;
    end;
  end;

```


Como tratar erros no Delphi?

Este tópico vai ensinar os procedimentos tratar erros em uma aplicação feita em Delphi.

Procedimentos a serem executados:

- ✓ Digite o comando **Try**
- ✓ Digite a seqüência de comandos desejada
- ✓ Utilize a palavra **Finally** e logo a seguir os comandos que deverão ser executados independente do que ocorreu anteriormente, ou seja, que serão sempre executados. Geralmente usamos este recurso para fechar tabelas, habilitar controles de dados, etc.
- ✓ Ou então, utilize a palavra **Except** e logo a seguir os comandos que deverão ser executados caso seja levantada alguma exceção, ou seja, tenha ocorrido um erro. Geralmente utilizamos este recurso para alertar o usuário a respeito do erro e fazer os ajustes necessários ao sistema.
- ✓ **Exemplos:**

```
Function divide(A, B : Integer);
begin
  try
    divide := A div B;
  except
    On EdivByZero do
      begin
        Divide := 0 ; // este bloco só será executado se ocorrer um erro de divisao por 0
        MessageDlg('Divisão por zero corrigida', mtError, [mbOk],0);
      end;
    end;
  end;
end;
```

```
procedure VarreClientes;
begin
  dm.tab_Clientes.DisableControls; // desabilita os controles de dados
  try
    while not dm.tab_Clientes.Eof do //executa enquanto não for fim de arquivo de clientes
    begin
      {bloco de comandos qualquer}
      dm.tab_Clientes.Next
    end;
  finally
    dm.tab_Clientes.EnableControls; // sempre será executada essa linha
  end;
end;
```

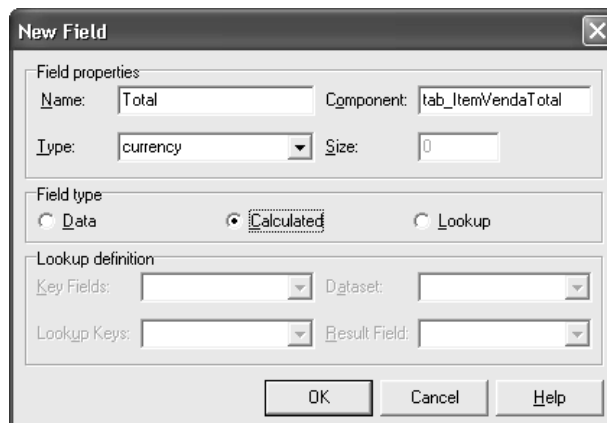
OBS: Os comandos acima são apenas ilustrativos de exemplos. Para utilizá-los no sistema, faça as devidas adaptações, de acordo com o local onde julgar necessário. Em vários tópicos a seguir utilizaremos exemplos de uso dos mesmos. Não digite nada ainda.

Como criar campos calculados?

Este tópico vai ensinar os procedimentos para criar campos do tipo calculado, cujos cálculos são feitos toda vez que um registro da tabela é alterado.

Procedimentos a serem executados:

- ✓ Visualize a tela de **DataModule** do Sistema, ou qualquer tela que possua o objeto **ADOTable**. **[Shift] + [F12]**
- ✓ Dê duplo-clique sobre o objeto **TADOTable** desejado. Para nosso exemplo, escolha **tab_ItemVenda**
- ✓ Tecle **[Ctrl] + [N]** para inserir um novo campo.
- ✓ Em **Fields Properties** digite o nome do campo na caixa de texto **Name** e escolha o tipo do mesmo em **Type**.
- ✓ Em **Field Type** marque o tipo do campo como **Calculated**.
- ✓ Para que seja criado um campo **Total** para calcular o total do produto vendido, deixe sua tela conforme o modelo a seguir:




- ✓ Clique sobre o botão **[OK]**
- ✓ Para que o campo calculado funcione inclua o cálculo no evento **OnCalcFields** da tabela para a qual ele foi criado. No evento **OnCalcFields** da tabela **tab_ItemVenda**, digite:

```
dm.tab_ItemVendaTotal.Value := dm.tab_ItemVendaltVeQtde.Value *  
(dm.tab_ItemVendaltVeValorUnitario.Value -  
dm.tab_ItemVendaltVeDescontoItem.Value);
```

- ✓ Na tela **FrmCadItemVenda** dê um duplo-clique sobre o **DBGrid1**. Insira um



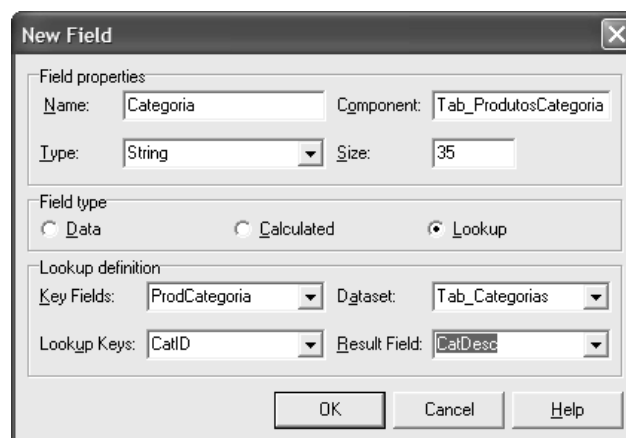
novo campo, clicando sobre o botão . Selecione a coluna **TColumn** e mude a propriedade **Fieldname** da mesma para **Total**.

Como criar campos Lookup?

Este tópico vai ensinar os procedimentos para criar campos lookup, que exibem valores de campos de registros de tabelas primárias de um relacionamento.

Procedimentos a serem executados:

- ✓ Visualize a tela de **DataModule** do Sistema, ou qualquer tela que possua o objeto **ADOTable**. Para isto tecle **[Shift] + [F12]**
- ✓ Dê duplo-clique sobre o objeto **ADOTable** desejado. Para nosso exemplo, escolha **tab_Produtos**
- ✓ Tecle **[Ctrl] + [N]** para inserir um novo campo.
- ✓ Em **Fields Properties** digite o nome do campo na caixa de texto **Name**, escolha o tipo do mesmo em **Type** e digite o tamanho em **Size**.
- ✓ Em **Field Type** marque o tipo do campo como **Lookup**.
- ✓ Em **Lookup Definition** escolha em **Key Fields** a chave estrangeira. Em **Dataset** escolha o nome da tabela primária. Escolha o nome da chave primária da tabela escolhida anteriormente em **Lookup Keys**. E finalmente escolha o campo que aparecerá como resultado em **Result Field**. Para que seja exibido o nome da categoria do produto na tabela de produtos deixe sua tela conforme o modelo a seguir:



- ✓ Clique sobre o botão **[OK]**
- ✓ Acrescente o campo criado ao **DBGrid** da tela de **Produtos**, ou, arraste-o junto aos demais campos na tela, como já foi feito com os outros campos da mesma. Crie campos **Lookup** nas tabelas **Tab_ItemVenda** (*Produto*), **Tab_Venda** (*Cliente*) e adicione-os aos respectivos *DBGrids* e telas.

Como exportar / importar dados com Delphi/Excel?

Este tópico vai mostrar os comandos necessários para exportar / importar dados do Delphi para o Excel e vice-versa.

Procedimentos a serem executados:

- ✓ Acrescente **ComObj** na cláusula **uses** da unit da tela ***FrmCadCliente***.
- ✓ Em um botão na tela ***FrmCadCliente*** e digite os comandos abaixo:

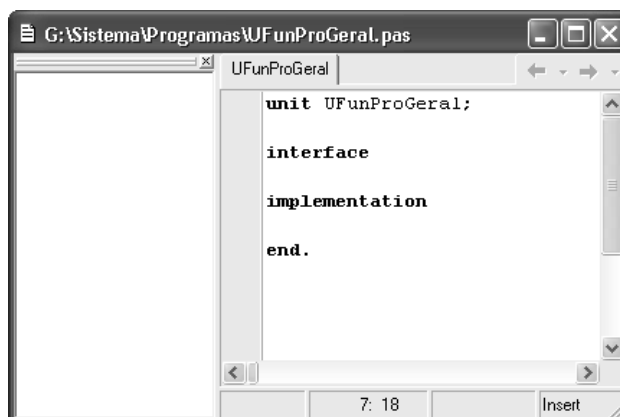
```
procedure TFrmCadCliente.BtnPlanilhaClick(Sender: TObject);
var
  Pasta : Variant; // este tipo aceita qualquer tipo de informação, inclusive Objeto OLE
  Linha : Integer;
begin
  dm.tab_clientes.Filtered := False;
  Linha := 2;
  Pasta := CreateOleObject('Excel.Application'); // cria uma aplicação do Excel
  Pasta.WorkBooks.Add(1); // adiciona uma pasta do Excel
  Pasta.Caption := 'Cadastro de Clientes'; // Título da planilha
  Pasta.Visible := False; // Deixa a planilha invisível
  Pasta.Cells[1,1] := 'Cliente'; // insere o conteúdo 'Cliente' na célula A1
  Pasta.Cells[1,2] := 'Cidade'; // insere o conteúdo 'Cidade' na célula A2
  Pasta.Cells[1,3] := 'Fone'; // insere o conteúdo 'Fone' na célula A3
  dm.tab_Clientes.DisableControls; // desabilita os controles de dados
  Try
    While not dm.tab_Clientes.Eof do //executa enquanto não for fim de arquivo de clientes
      begin
        Pasta.Cells[Linha,1] := dm.Tab_ClientesCliNome.Value;
        Pasta.Cells[Linha,2] := dm.Tab_ClientesCliCid.Value;
        Pasta.Cells[Linha,3] := dm.Tab_ClientesCliNumFone.Value;
        Linha := Linha + 1; // Incrementa a variável Linha em 01
        dm.tab_Clientes.Next; //vai para o próximo registro da tabela de clientes
      end;
    Pasta.Columns.AutoFit; // Faz auto ajuste das colunas do Excel
    Pasta.WorkBooks[1].Sheets[1].Protect(DrawingObjects:=True, Contents:=True, Scenarios:=True,
    Password:='1234'); // Coloca Senha de Proteção na Planilha 01
    If SaveDialog1.Execute then // O componente SaveDialogs está na paleta Dialogs
      Pasta.WorkBooks[1].SaveAs(SaveDialog1.FileName); // Salva a Planilha (Salvar como)
    Pasta.Visible := True; //Deixa a planilha visível
  Finally
    dm.tab_Clientes.EnableControls; // sempre será executada essa linha
    Pasta := Unassigned;
  end;
end;
```

Como utilizar uma Unit não vinculada a forms?

Este tópico vai ensinar os procedimentos para utilizar units não vinculadas a forms para criar um arquivo de funções e procedimentos comuns à aplicação.

Procedimentos a serem executados:

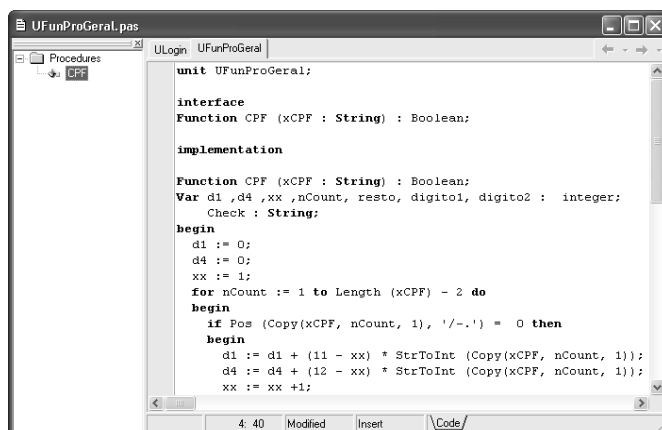
- ✓ **File -> New -> Unit** para acrescentar uma unit nova não vinculada a tela na aplicação. Para acessá-la de outros locais, use **[Ctrl] + [F12]**.
- ✓ **File -> Save** para salvar a **Unit1.pas** como **UFunProGeral.pas**
- ✓ Uma tela aparecerá, como a do modelo a seguir:



A seção **Unit** possui o nome da unit gravada no disco (*.pas)

Na seção **interface** declare as funções e procedimentos de uso comum.

Na seção **Implementation** implemente tudo o que foi declarado na **interface**.



Para que as funções e procedimentos criados na **nova Unit** possam ser utilizados, informe o nome da mesma na cláusula **uses** de cada **unit** em que os mesmos forem solicitados.

Aproveite este momento para pesquisar ou criar procedures e funções que possam ser utilizadas em vários pontos do sistema e acrescente-as à Unit **UFunProGeral.pas**

Como utilizar uma DLL?

Este tópico ensinará como criar uma DLL - biblioteca de ligação dinâmica.

Procedimentos a serem executados:

- ✓ **File -> Close All**
- ✓ **File->New->Other->DLL Wizard**
- ✓ Apague o comentário após a linha **Library Project2;**
- ✓ Digite as funções que farão parte da DLL e deixe-a da seguinte maneira:

```
Library DLLCurso;  
Uses SysUtils, Classes;  
{ $R *.res }  
Function Triplo(N: Integer): Integer; stdcall; // permite exportar para C, C++, etc.  
begin  
    Result := N * 3;  
end;  
Function Dobro (N: Integer): Integer; stdcall;  
begin  
    Result := N * 2;  
end;  
Exports Triplo Index 1, Dobro Index 2; // Permite acesso às funções declaradas  
end.
```

- ✓ Salve o projeto como "C:\Sistema\DLLCurso" e compile-o ([Ctrl] + [F9]). Após isso será criado um arquivo com o nome do projeto, só que com a extensão **DLL**.
- ✓ Para usar as DLL, crie uma aplicação nova com **02 botões** e **01 Edit**, e declare suas funções antes da seção **Implementation**, no seguinte formato:

```
Function Triplo(N: Integer): Integer; stdcall;  
Function Dobro(N: Integer): Integer; stdcall;
```

- ✓ Dentro da seção **Implementation**, após a diretiva {\$R *.DFM}, digite:

```
Function Triplo(N: Integer): Integer; external 'C:\Sistema\DLLCurso.dll';  
Function Dobro(N: Integer): Integer; external 'C:\Sistema\DLLCurso.dll';
```

- ✓ Chame cada função no evento **OnClick** dos botões, passando como parâmetro o número de que se deseja calcular o triplo ou o dobro.

Ex.: Edit1.Text := IntToStr(**Dobro**(100)); // retorna 200

 Edit1.Text := IntToStr(**Triplo**(100)); // retorna 300

Como preparar um sistema para ser utilizado em rede?

Este tópico vai ensinar os procedimentos para preparar o sistema para ser utilizado pelos usuários em uma rede.

Procedimentos a serem executados:

- ✓ Abra a aplicação, acione a tela **DM** da mesma, teclando **[Shift] + [F12]**, e apague o conteúdo da propriedade **ConnectionString** do objeto **Conexao**.
- ✓ Acrescente as próximas linhas de comando no programa principal (*.dpr), logo após a criação da tela **DM** do sistema:

```
Dm.Conexao.Connected := False;  
Dm.Conexao.ConnectionString :=  
'Provider=Microsoft.Jet.OLEDB.4.0;Data Source=' +  
ExtractFilePath(Application.ExeName)+  
'Banco.mdb;Persist Security Info=False;';  
Dm.Conexao.Connected := True;
```

A primeira linha de comando fecha a conexão com banco de dados.

A segunda linha de comando cria uma “string” para a conexão com o banco de dados, sendo que, “**Provider=Microsoft.Jet.OLEDB.4.0**”, indica que o provedor de banco de dados utilizado é o **Microsoft Jet OLEDB 4.0**, o que permite a utilização de um arquivo *.mdb do **Microsoft Access** como banco de dados da aplicação juntamente com **ADO**. A outra parte do “string” **Data Source=' + ExtractFilePath(Application.ExeName)** diz que o endereço do banco de dados será o mesmo da pasta em que estiver o executável da aplicação (*.exe). O restante do “string” “**Banco.mdb;Persist Security Info=False**” indica o nome do banco de dados (*.mdb) da aplicação e que não será utilizada nenhuma informação de segurança, como usuário e senha para o banco de dados.

A terceira linha de comando abre a conexão do objeto **ADOConnetion** com o banco de dados para ser utilizada no sistema.

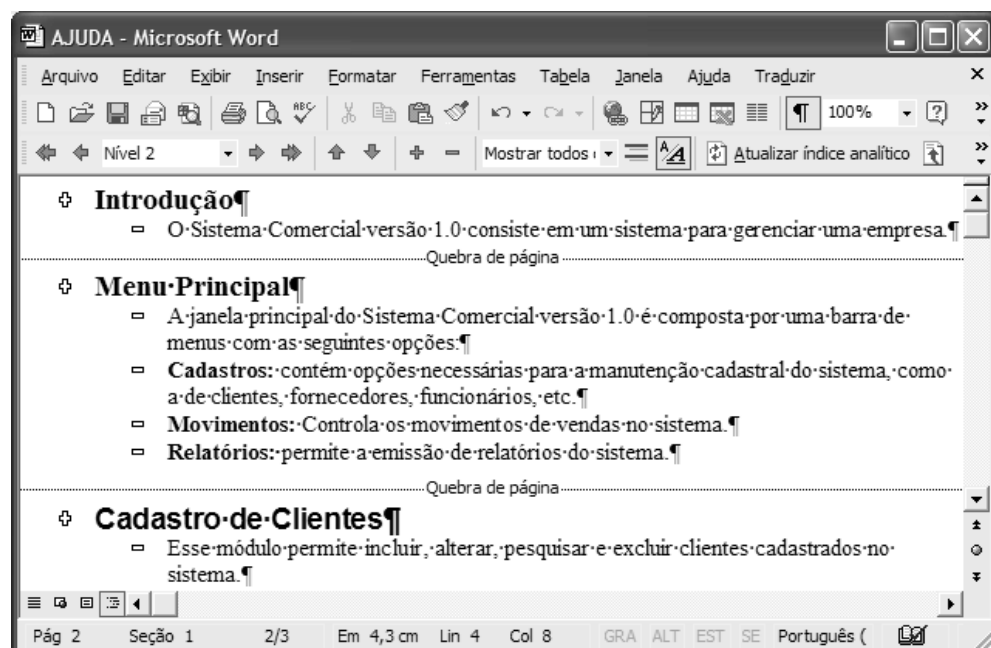
- ✓ Compile o sistema e grave o executável da aplicação (*.exe) juntamente com o arquivo de banco de dados do Access (*.mdb) em uma mesma pasta no computador considerado como “servidor” da aplicação.
- ✓ Para encerrar, crie um atalho apontando para o executável (*.exe) da aplicação, em cada máquina considerada como “cliente” do “servidor” da aplicação na rede.

Como criar um sistema de ajuda?

Este tópico vai ensinar os procedimentos para criar um sistema de ajuda que será acionado pela tecla **[F1]**.

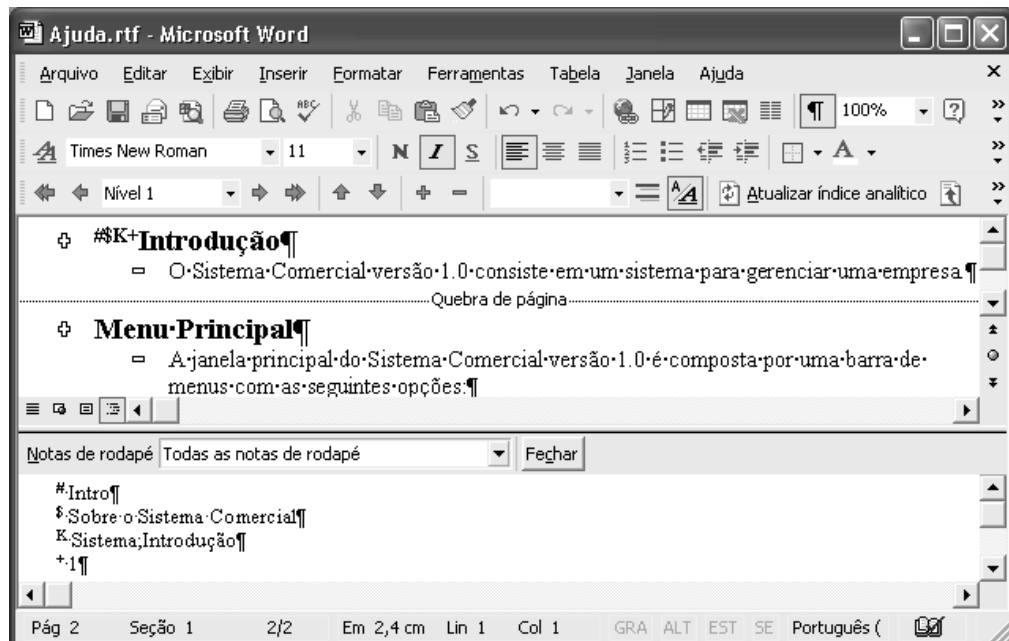
Procedimentos a serem executados:

- ✓ Carregue algum editor de textos que aceite trabalhar com o tipo **Rich Text Format (*.RTF)**, como o Microsoft Word, por exemplo.
- ✓ Digite o texto de ajuda agrupado por tópicos, deixando cada tópico ocupando uma página separada (quebra de página = **[Ctrl] + [Enter]**, conforme exemplo a seguir).



- ✓ Identifique cada um dos tópicos de acordo com os atributos abaixo:
 - # (contexto)**: Este atributo serve para identificar o tópico de ajuda. Ele não será visível no arquivo de ajuda.
 - \$ (título)**: define o título do tópico que será exibido na janela de índice do sistema de ajuda.
 - K (chave)**: palavra-chave associada ao texto exibida na janela de índice do sistema de ajuda. Se houver mais de uma palavra, separe-as por ponto-e-virgula (;).
 - + (ordem)**: valor numérico que define o número de ordem da apresentação no sistema de ajuda.
- ✓ Cada identificação deve ser feita como uma nota de rodapé, portanto coloque o cursor antes de cada título principal (¶ **Introdução**, por exemplo) e acione o comando **Inserir -> Referência -> Notas...** e escolha

notas de rodapé. Na caixa de texto **Personalizada** digite o símbolo identificador e clique no botão **Inserir**. Na **nota de rodapé** digite o texto do identificador, conforme a tela a seguir:



- ✓ Observe que no exemplo anterior foi identificado apenas o tópico de **Introdução**, portanto, faça o mesmo para os outros dois tópicos restantes, utilizando os mesmos procedimentos.
- ✓ Defina os links do seu sistema de ajuda. Por exemplo, no tópico **Menu Principal**, na opção **Cadastros**, selecione a palavra **clientes**, aplique **Formatar - > Fonte**, e escolha o estilo de sublinha **traço duplo**.
- ✓ Exatamente após a palavra clientes, digite a palavra-chave **cadcli** (nome do contexto #), selecione-a, e aplique **Formatar->Fonte** com efeito **oculto** e salve o seu documento como **Ajuda.rtf**.
- ✓ Na pasta *C:\Arquivos de programas\Borland\Delphi7\Help\Tools*, execute o programa **HCW.exe** (Help WorkShop).
- ✓ Crie o arquivo de conteúdo no Help WorkShop: **File->New-> Help Contents** e digite na caixa **Default filename (and Windows)** o nome do arquivo **Ajuda.hlp**. Para inserir um cabeçalho, clique sobre o botão **Add Above**, selecione **Heading** e digite na caixa **Title:** *Sistema Comercial versão 1.0* e clique em **OK**.
- ✓ Digite cada tópico do cabeçalho clicando sobre o botão **Add Below**, escolhendo a opção **Topic**. Como modelo, digite na caixa **Title:** *Introdução ao Sistema Comercial*, e na caixa **Topic Id** digite **Intro**, que corresponde à nota de rodapé usada para o marcador # no arquivo **Ajuda.rtf**. Faça o mesmo para os demais tópicos.
- ✓ Salve o arquivo de contexto como **Ajuda.cnt**
- ✓ Crie o arquivo de projeto no Help WorkShop: **File->New-> Help Project**

- ✓ Digite o nome do arquivo **Ajuda** e clique sobre o botão **salvar**.
- ✓ Na nova janela do Help WorkShop clique sobre o botão **Options**, escolha a guia **General** e digite na caixa **Help Title**: *Ajuda do Sistema Comercial*.
- ✓ Selecione **Files**, clique sobre o botão **Change**.
- ✓ Clique sobre o botão **Add**, selecione o arquivo **Ajuda.rtf** e clique em **Abrir**.
- ✓ Selecione **Files**, clique sobre o botão **Browse** em **Contents File**.
- ✓ Selecione o arquivo **Ajuda.cnt** e clique em **Abrir**.
- ✓ Para finalizar clique no botão **OK** e observe as mudanças realizadas, na área de trabalho do Help WorkShop.
- ✓ Clique no botão **Save and Compile** para salvar e compilar o projeto.
- ✓ Teste o arquivo: **File -> Run WinHelp -> View Help**.
- ✓ **Para vincular o Help à sua aplicação** clique em **File->Open** e abra o arquivo **Ajuda.hpj** e clique sobre o botão **Map** e logo em seguida sobre o botão **Add**.
- ✓ Na caixa **Topic Id** digite **Intro**. Na caixa **Mapped numeric value** digite **1** e clique sobre o botão **OK**. Siga o mesmo modelo para as demais opções e **recompile** o projeto. **Feche o Help WorkShop**.
- ✓ No Delphi, em seu sistema, para cada objeto que deverá acionar o respectivo tópico de ajuda através da tecla **[F1]**, selecione-o, e na propriedade **HelpContext** digite o número de identificação do tópico. Como modelo selecione a tela principal do sistema e altere a propriedade **HelpContext** para **1**. Faça o mesmo para os demais tópicos.
- ✓ Indique ao sistema o arquivo de ajuda a ser utilizado clicando sobre **Project -> Options**, e, na guia **Application**, selecione o arquivo **Ajuda.hlp** através do botão **Browse** da caixa de texto **Help File**. Clique sobre o botão **OK**.
- ✓ Execute a aplicação e sobre o objeto específico tecla **[F1]** para testar a ajuda.

Como criar discos de instalação para a aplicação?

Este tópico vai ensinar os procedimentos para criar discos de instalação da aplicação utilizando o programa **InstallShield**. Este programa acompanha as versões Professional e Enterprise do Delphi. Antes de executar os procedimentos deste tópico tenha certeza de que o **InstallShield** já tenha sido instalado em seu computador.

Procedimentos a serem executados:

- ✓ No Windows clique **Iniciar -> Programas -> InstallShield -> Express** para carregar o **InstallShield**
- ✓ No **InstallShield** clique na opção **“Create a new Project”**.
- ✓ **Selecione** a opção **Blank Setup Project** para criar uma instalação a partir do zero, manualmente.
- ✓ Clique no botão **“Browse...”** da caixa de texto **“Project Name and Location”** e na janela **“Select Filename”** escolha a pasta onde será armazenado o arquivo de instalação, digite o nome do arquivo e clique no botão **Salvar**.
- ✓ Ao retornar à tela do **InstallShield**, clique sobre o botão **Create**.
- ✓ Na **1ª Etapa – “Organize your Setup”**, clique sobre a opção **“General Information”** e defina as principais informações sobre o seu projeto, como: nome do produto, versão, código, nome do autor, etc.
- ✓ Clique sobre a opção **“Setup Types”** para definir os tipos de instalação que estarão disponíveis ao usuário, como por exemplo, típica, mínima, personalizada. Desmarque as opções **“&Minimal e Cu&stom”**
- ✓ Na **2ª etapa – “Specify Application Data”**, clique sobre a opção **“Files”** e selecione os arquivos da aplicação (executáveis, banco de dados, ajuda, etc) em **“Source Computer’s files”** dentro de suas respectivas pastas selecionadas em **“Source computer’s folders”**. Logo em seguida arraste-os para o painel inferior **“Destination computer’s files”**
- ✓ Clique sobre **“Files and Features”** para confirmar os arquivos selecionados para cada **“Feature”**.
- ✓ Clique sobre a opção **“Objects/Merge Modules”** para selecionar o tipo da conexão usada em seu projeto, em nosso caso, marque a opção **“ADO Data Control 6.0”**
- ✓ Na **3ª etapa – “Configure the Target System”** clique sobre a opção **“Shortcuts / Folders”** para configurar o computador de destino, no que diz respeito a atalhos e pastas para a instalação dos arquivos.

- ✓ Na 4ª etapa – “**Customize the setup appearance**”, clique sobre a opção “**Dialogs**” e selecione as seguintes opções: “**Splash Bitmap, Install Welcome, Customer Information, Ready to Install, Setup Progress, Setup Complete Success**”.
- ✓ Na 7ª etapa – “**Prepare for release**”, clique sobre a opção “**Build your release**” para dar início à construção da instalação. Para criar uma instalação em um CD, escolha a opção “**CD_ROM**” e pressione **[F7]** para iniciar o processo de criação do disco e ser exibido o log de criação na parte inferior da tela.
- ✓ Teste o arquivo de instalação clicando sobre “**Test your Release**”, em seguida **[Ctrl] + [T]**.
- ✓ Caso tudo esteja correto, faça a cópia para o CD.

Obs: Para instalações mais simples, utilize o modo **Project Wizard** (assistente de projetos), que auxilia, passo a passo, a construção do programa de instalação.

Como compactar o banco de dados do Access no Delphi?

Este tópico vai mostrar como compactar o banco de dados criado no Access para eliminar todo seu espaço excedente, reduzindo assim, o tamanho do arquivo de banco de dados.

Procedimentos a serem executados:

- ✓ Acrescente um **Panel** invisível (**Visible = False**) na tela **FrmMenuPrin**.
- ✓ Na Unit da tela **FrmMenuPrin**, acrescente a unit **ComObj** na sua cláusula **Uses** e crie a **procedure** abaixo:

```
Procedure TFrmMenuPrin.Compactar;  
var dao: OLEVariant;  
begin  
  Panel1.Visible := True;  
  Dm.Conexao.Connected := False;  
  Try  
    Panel1.Caption:='Compactando Tabela';  
    Panel1.Repaint;  
    dao := CreateOleObject('DAO.DBEngine.36');  
    dao.CompactDatabase(ExtractFileDir (Application.ExeName)  
+ '\Banco.mdb', ExtractFileDir(Application.ExeName)  
+ '\Banco2.mdb',",0,"");  
    Panel1.Caption:='Apagando Arquivo Temporário';  
    Panel1.Repaint;  
    If FileExists(ExtractFileDir (Application.ExeName)  
+ '\Banco2.mdb') then DeleteFile(ExtractFileDir  
(Application.ExeName)+'\Banco.mdb');  
    Panel1.Caption:='Renomeando Arquivo';  
    Panel1.Repaint;  
    if FileExists(ExtractFileDir (Application.ExeName)  
+ '\Banco2.mdb') then  
      RenameFile(ExtractFileDir(Application.ExeName)  
+ '\Banco2.mdb',  
      ExtractFileDir (Application.ExeName)+'\Banco.mdb');  
    Panel1.Caption:='Arquivo Banco.mdb Compactado';
```

- ✓ Crie uma opção no menu principal: **Ferramentas -> Compactar** para chamar a procedure **Compactar**: Lembre-se usar a **ActionList1** (**ActComp**) e digite os seguintes comandos no Evento **OnExecute** da ação **ActComp**:

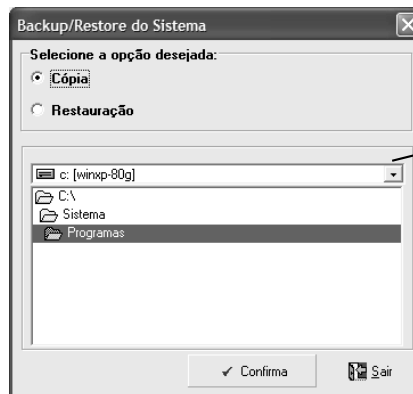
```
If MessageDlg('Antes de confirmar esta operação, feche o banco de dados.' + #13 + #13 + 'Deseja  
efetuar a compactação do Banco de Dados?', mtConfirmation, [mbYes, mbNo],0) = mrNo then  
  Abort;  
Compactar;
```

Como criar uma rotina de Backup/Restore do sistema?

Este tópico vai mostrar como criar uma rotina para cópia de segurança e restauração do banco de dados do sistema.

Procedimentos a serem executados:

- ✓ Crie uma opção no menu principal: **Ferramentas -> Backup/Restore**.
- ✓ Crie a tela a seguir com o **Name = FrmBkpRst**, contendo os objetos:
 - **RadioGroup** (*Standard*) – use as propriedades **Caption** e **Items**
 - **DriveComboBox** e **DirectoryListBox** (*Win 3.1*), **02 SpeedButton**
 - **Animate** (*Win 32*) – **Visible = False**, **CommonAvi = aviCopyFile**



Selecione o **DriveComboBox1** e altere sua propriedade **DirList**, para **DirectoryListBox1** para que os dois fiquem associados.

- ✓ Digite os comandos abaixo no evento **OnClick** do botão **Confirma**:

```
If MessageDlg('Antes de confirmar esta operação, feche o banco de dados' + #13 + #13 + 'Deseja efetuar ' +  
RadioGroup1.Items[RadioGroup1.ItemIndex] + '?', mtConfirmation,  
[mbYes, mbNo],0) = mrNo then  
    Abort;  
Try  
    Animate1.Visible := True;  
    Animate1.Active := True; //ativa a animação  
    If RadioGroup1.ItemIndex = 0 then  
        begin  
            FrmMenuPrin.Compactar;  
            Copia(ExtractFilePath(Application.ExeName)+'\Banco.mdb',  
DirectoryListBox1.Directory);  
        end;  
    If RadioGroup1.ItemIndex = 1 then // se a segunda opção do  
    RadioButton for escolhida  
        begin  
            Dm.Conexao.Connected := False; // Desconecta o banco de dados
```

- ✓ Faça referência à unit **UFunProGeral** na cláusula **Uses** da unit da tela **FrmBkpRst**:
- ✓ Crie uma procedure denominada **Copia**, na unit **UFunProGeral**, conforme mostrada a seguir:

```

Procedure Copia(Origem, Destino: String);
var FileOpInfo: TSHFileOpStruct;
begin
  With FileOpInfo Do
    Begin
      Wnd := Application.Handle;
      wFunc := FO_COPY;
      pFrom := Pchar (Origem+#0+#0);
      pTo := Pchar (Destino);
      fFlags := FOF_WANTMAPPINGHANDLE;
    end;
  SHFileOperation (FileOpInfo);
  ShFreeNameMappings

```

- ✓ Faça referência à unit **ShellAPI** na cláusula **Uses** da unit **UFunProGeral**.
- ✓ Chame a tela **FrmBkpRst** no Evento **OnExecute** da ação **ActBkpRst**.

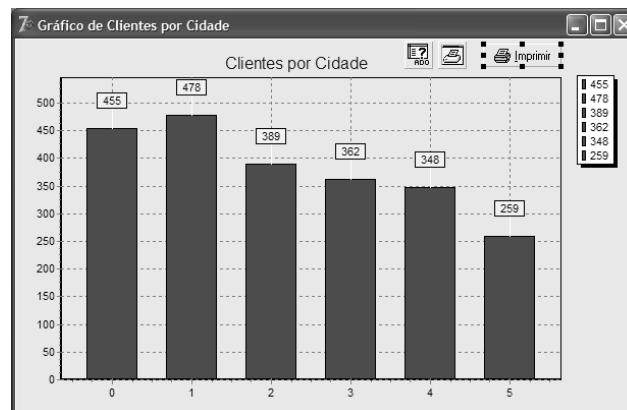
Como criar um gráfico no Delphi?

Este tópico vai mostrar como criar gráficos estatísticos no Delphi, como por exemplo, gráfico de barras, verticais, de pizza, etc.

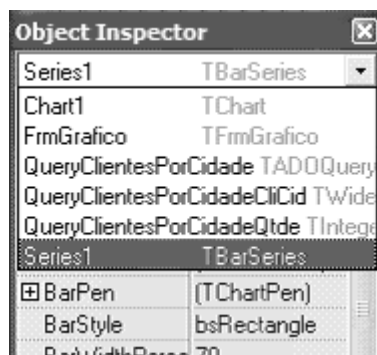
Procedimentos a serem executados:

- ✓ Crie uma tela nova no sistema, altere as propriedades **Name** = *FrmGrafico* e **Caption** = *Gráfico de Clientes por Cidade*.
- ✓ Insira um objeto **Chart** (*Additional*) à tela. Altere as propriedades **Name** = *chtGrafico*, **Align** = *alClient* e dê um duplo-clique sobre o mesmo.
- ✓ Escolha o tipo de gráfico a ser utilizado: **Bar** (Barras). Para isto, clique nas guias **Chart**, **Series** e nos botões **Add** (Desmarque a opção **3D**) e **OK**.
- ✓ Clique no botão **Titles** e altere o título da série para **Cidades**.
- ✓ Selecione a guia **Titles** e digite o título do gráfico: *Clientes por Cidade*, na caixa de texto e selecione o **Alignment** = **Center**.
- ✓ Clique no botão **Close**.
- ✓ Faça referência à unit da tela **DM**. **File** -> *Use Unit (UDM)*
- ✓ Insira um objeto **ADOQuery** à tela e altere as propriedades:
 - **Name**: *QCliCid*
 - **Connection**: *Dm.Conexao*
 - **SQL**: *Select Count(*) as Qtde, CliCid From Clientes Group By CliCid*
- ✓ Dê duplo-clique na query **QCliCid** e tecle **[Ctrl] + [F]** para inserir os campos.
- ✓ Copie o botão **Btn_Imprimir** da tela **FrmCadCliente** para a tela **FrmGrafico**.
- ✓ Insira um componente **PrintDialog** da guia **Dialogs** na tela **FrmGráfico**.

A tela deverá ficar com a seguinte aparência:



- ✓ No **Object Inspector**, selecione o objeto **Series1** do gráfico **chtGrafico**:



- ✓ Altere a propriedade **Name** para *CliCid*
- ✓ No evento **OnShow** da tela **FrmGrafico**, digite os seguintes comandos:

```

CliCid.Clear;
chtGrafico.Title.Text.Clear;
chtGrafico.Title.Text.Add('Gráfico Clientes por Cidade');
QCliCid.Open;
While (not QCliCid.Eof) do
begin
    // Insere dados do Eixo Y do gráfico de barras
    CliCid.AddY(QCliCidQtde.Value, QCliCidCliCid.AsString, clSkyBlue);
    QCliCid.Next;
end;

```

- ✓ Digite os comandos a seguir no evento **OnClick** do botão **Imprimir**:

```

if PrintDialog1.Execute then
begin
    chtGrafico.BackColor := clWhite;
    chtGrafico.Print;
end;

```

- ✓ Acrescente um botão com **Caption** = *Gráfico* à tela **FrmCadCliente** e chame a tela **FrmGráfico** através dele.

Como criar um relatório de produtos em falta

Procedimentos a serem executados:

- ✓ Acesse a guia **ADO**
- ✓ Insira um componente **ADOQuery**
- ✓ Altere as seguintes propriedades:
 - Name:** QComprarProdutos
 - Connection:**ConexaoBanco
 - SQL...:** Select CatDesc, ProdNome, ProdQtdeEst From Categorias, Produtos
Where CatCodigo = ProdCategoria And ProdQtdeEst <= :Qtde
 - Active:** True
- ✓ Dê um duplo-clique no objeto **QComprarProdutos** e tecle **[Ctrl] + [F]**
- ✓ Coloque um botão (**Name** = BtnProdFalta e **Caption** = "Relatório de Produtos em Falta"), e digite no evento **OnClick** do mesmo os seguintes comandos:

```
procedure TFrmCadProduto.SpeedButton4Click(Sender: TObject);
var Qtde : String;
    OK : Boolean;
begin
    OK := InputQuery('Digite a Qtde Mínima', 'Qtde Mínima', Qtde);
    if OK then
        begin
            With QComprarProdutos do
                begin
                    Close;
                    Parameters[0].value := Qtde;
                    Open;
                    RvPrjProdFalta.Execute;
                    Close;
                end;
        end;
    end;
end;
```

- ✓ Crie um relatório no **RAVE** com o seguinte layout

RELATÓRIO DE PRODUTOS PARA COMPRA

Categoria	Nome do Produto	Qtde em Estoque
XXXXXXXXXX	XXXXXXXXXXXXXXXXXX	99
XXXXXXXXXX	XXXXXXXXXXXXXXXXXX	99
XXXXXXXXXX	XXXXXXXXXXXXXXXXXX	99

- ✓ Crie uma opção no menu *Relatórios* para executar e testar o relatório

Como realizar algumas melhorias no sistema?

Procedimentos a serem executados:

- ✓ No evento **OnBeforeAction** do **DbNavigator2** da tela padrão do sistema, digite:

```
if Button in [nbInsert, nbEdit] then
begin
  GroupBox1.Enabled := False; //desabilita o GroupBox1 e todos seus componentes.
  GroupBox2.Enabled := False;
  DbNavigator1.Enabled := False;
  DbNavigator2.Enabled := False;
  DbNavigator3.Enabled := True;
end;
```

- ✓ No evento **OnClick** do **DbNavigator3** da tela padrão do sistema, digite:

```
GroupBox1.Enabled := True;
GroupBox2.Enabled := True;
DbNavigator1.Enabled := True;
DbNavigator2.Enabled := True;
DbNavigator3.Enabled := False;
```

- ✓ No evento **OnBeforeAction** do **DbNavigator2** da tela **FrmCadCliente**, digite:

```
if (Button = nbInsert) then
begin
  Dm.Tab_Clientes.Cancel;
  Dm.Tab_Clientes.Append;
  EdtNome.SetFocus;
end;
```

- ✓ Faça o mesmo procedimento para as demais telas do sistema. Lembre-se de alterar o nome das tabelas e do componente que receberá o foco na tela.

- ✓ Para utilizar a tecla **[Enter]** como **[Tab]** nas telas do sistema, selecione a tela padrão do sistema e altere a propriedade **KeyPreview** para **True**. Logo em seguida, digite os seguintes comandos no evento **OnKeyPress** da tela padrão:

```
if Key = #13 then
begin
  keybd_event(9,0,0,0);
  Key := #0;
end;
```

Como criar um objeto em tempo de execução?

Este tópico vai mostrar como criar um objeto, a partir de uma classe existente, em tempo de execução.

Procedimentos a serem executados:

- ✓ Crie uma nova aplicação para teste
- ✓ Acrescente um botão à tela
- ✓ Defina uma variável do tipo do objeto que deseja criar. Por exemplo, defina uma variável **Edit1** do tipo **TEdit**. (**Var Edit1 : TEdit**) na área de definição global de variáveis, ou seja, antes da seção **Implementation**.
- ✓ Crie uma instância do **Edit1** e defina sua propriedade **Parent** como **Self**, digitando o seguinte código no evento **OnClick** de um **botão**:

```
Edit1:= TEdit.Create(Self); // Diz que controle proprietário do Edit1 é ele mesmo
Edit1.Parent := Self; // Diz que controle de origem do Edit1 é ele mesmo.
Edit1.SetBounds(300,300,175,21); // Ajusta Left, Top, Width, Height do Edit1
Edit1.Font.Color := clNavy;
Edit1.Text := 'Edit criado em tempo de execução.';
```

- ✓ Execute a aplicação para verificar o resultado.
- ✓ Insira um novo botão na tela e utilize os comandos seguintes para remover o objeto da tela e da memória, em tempo de execução:

```
Edit1.Free; // Método que elimina o objeto da memória.
Edit1 := nil; // anula a referência ao ponteiro do objeto.
```

- ✓ **OBS:** Pesquise mais sobre os conceitos de POO – Programação Orientada a Objetos do Delphi, tais como, classe, objeto, encapsulamento, polimorfismo, herança, etc. Conheça também a hierarquia de classes da **VCL** – Biblioteca de Componentes Visuais do Delphi, cuja classe principal é a **TObject**.

Como estruturar o banco de dados “Banco.MDB”

Obs: Todo campo precedido de um asterisco (*) deve ser configurado como chave primária na respectiva tabela.

Tabela: **Clientes**

Nome Campo	Tipo	Tamanho
*CliCodigo	AutoNumeração	
CliNome	Texto	50
CliEnd	Texto	40
CliCep	Texto	9
CliCid	Texto	35
CliEst	Texto	2
CliNumFone	Texto	15
CliEmail	Texto	50
CliDoc1	Texto	15
CliDoc2	Texto	15
CliContato	Texto	50

Tabela: **Fornecedores**

Nome Campo	Tipo	Tamanho
*ForCodigo	AutoNumeração	
ForRazao	Texto	40
ForEnd	Texto	40
ForCid	Texto	35
ForEst	Texto	2
ForCep	Texto	8
ForCont	Texto	35
ForNumFone	Texto	15
ForHomPag	Texto	50

Tabela: **Funcionarios**

Nome Campo	Tipo	Tamanho
*FunCodigo	AutoNumeração	
FunNome	Texto	50
FunEnder	Texto	40
FunCep	Texto	8
FunCid	Texto	30
FunEst	Texto	2
FunNumFone	Texto	15
FunDatAdm	Data/Hora	
FunSalario	Moeda	

Tabela: **Usuarios**

Nome Campo	Tipo	Tamanho
*UsuCodigo	AutoNumeração	
UsuNome	Texto	50
UsuApelido	Texto	15
UsuSenha	Texto	7
UsuDepto	Texto	15
UsuNivel	Número	

Tabela: **Categorias**

Nome Campo	Tipo	Tamanho
*CatCodigo	AutoNumeração	
CatDesc	Texto	15

Tabela: **Produtos**

Nome Campo	Tipo	Tamanho
*ProdID	AutoNumeração	
ProdCategoria	Número	
ProdCodigo	Texto	7
ProdNome	Texto	50
ProdPrecoVenda	Moeda	
ProdQtdeEst	Número	
ProdUnidade	Texto	5

Tabela: **Venda****Nome Campo**

*VendID
VendCliente
VendDt_venda
VendValorProdutos
VendDesconto
VendAcrescimo
VendFrete
VendQtde

Tipo

AutoNumeração
Número
Data/Hora
Moeda
Moeda
Moeda
Moeda
Número

TamanhoTabela: **ItemVenda****Nome Campo**

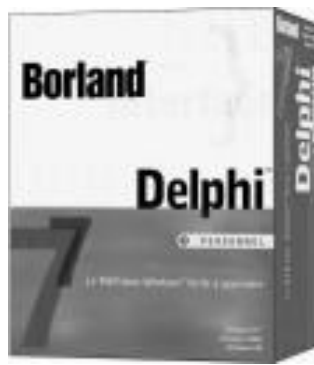
*ItVeID
ItVeVenda
ItVeProduto
ItVeValorUnitario
ItVeDescontoItem
ItVeQtde

Tipo

AutoNumeração
Número
Número
Moeda
Moeda
Número

Tamanho

DELPHI 7 PARA DESENVOLVIMENTO DE SISTEMAS



APOSTILA

CONTEÚDO: Delphi 7 para Desenvolvimento de Sistemas

ELABORADA POR: Ronaldo Lavestein - Casa Branca - SP