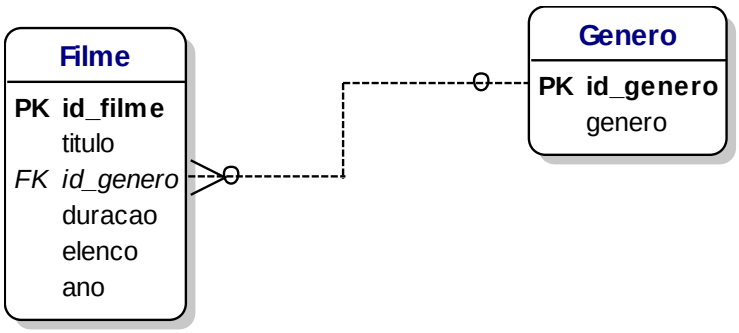
PROJETO LOCADORA – Delphi XE + Firebird

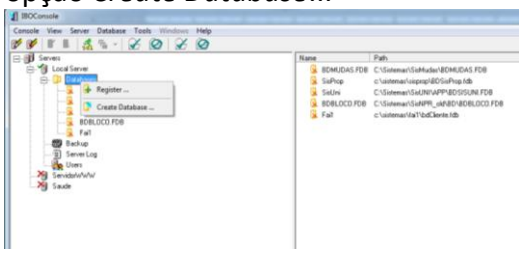
Obs: *Toda estrutura do projeto está localizada em C:\Sistemas\Locadora*
Os executáveis ficarão em C:\Sistemas\Locadora\APP, para tanto, crie também já a pasta APP.

Modelo ER

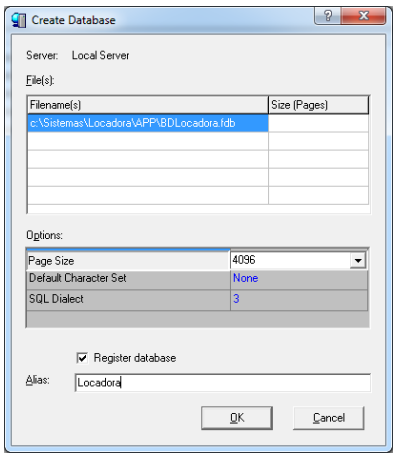


Criando o Banco usando IBConsole

Após conectar ao BD, clicar com botão direito do mouse sobre *Databases* e escolher a opção *Create Database*...



Será apresentada a tela de criação do Banco de Dados:



Em filename preencha:
c:\Sistemas\Locadora\APP\BDLocadora.fdb

Em Alias, preencha como Locadora.

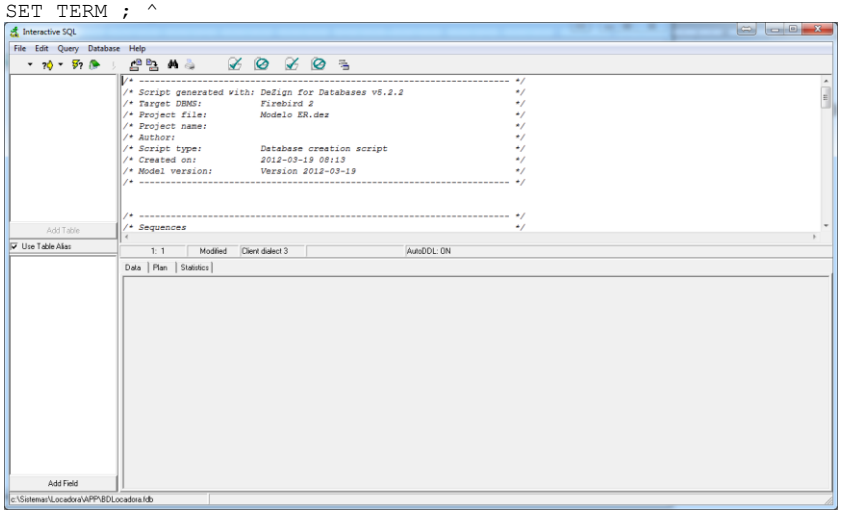
OBS: os arquivos do sistema serão colocadas na subpasta APP

Com o projeto Locadora selecionado, abra o Interactive SQL (Menu Tool), para executar o comando SQL de criação do Banco de Dados.

```
/* ------------------------------------------------------------------------ */
/* Sequences                                                                */
/* ------------------------------------------------------------------------ */

CREATE GENERATOR GEN_GENERO;
SET GENERATOR GEN_GENERO TO 0;

CREATE GENERATOR GEN_FILME;
SET GENERATOR GEN_FILME TO 0;
```

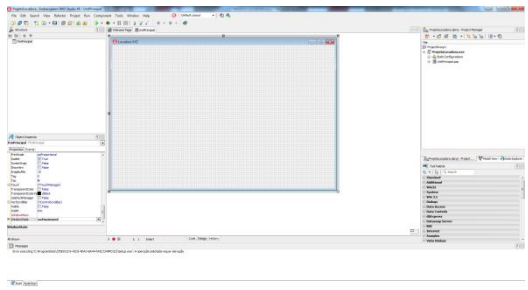
Fai Faculdades
Gestão de Tecnologia da Informação
Prof: André Klunk

```
/* ---------------------------------------------------------------------- */
/* Tables                                                                 */
/* ---------------------------------------------------------------------- */

/* ---------------------------------------------------------------------- */
/* Add table "GENERO"                                                     */
/* ---------------------------------------------------------------------- */

CREATE TABLE GENERO (
    ID_GENERO INTEGER NOT NULL,
    GENERO VARCHAR(40) NOT NULL,
    CONSTRAINT PK_GENERO PRIMARY KEY (ID_GENERO)
);

/* ---------------------------------------------------------------------- */
/* Add table "FILME"                                                      */
/* ---------------------------------------------------------------------- */

CREATE TABLE FILME (
    ID_FILME INTEGER NOT NULL,
    TITULO VARCHAR(100) NOT NULL,
    ID_GENERO INTEGER,
    DURACAO TIME,
    ELENCO VARCHAR(250),
    CONSTRAINT PK_FILME PRIMARY KEY (ID_FILME)
);

/* ---------------------------------------------------------------------- */
/* Foreign key constraints                                                */
/* ---------------------------------------------------------------------- */

ALTER TABLE FILME ADD CONSTRAINT GENERO_FILME
    FOREIGN KEY (ID_GENERO) REFERENCES GENERO (ID_GENERO);

/* ---------------------------------------------------------------------- */
/* Triggers                                                               */
/* ---------------------------------------------------------------------- */

SET TERM ^ ;
CREATE TRIGGER TRG_GENERO1 FOR GENERO
BEFORE INSERT AS BEGIN
  IF (NEW.ID_GENERO IS NULL) THEN NEW.ID_GENERO = GEN_ID(GEN_GENERO, 1);
END
^
 ^
SET TERM ; ^

SET TERM ^ ;
CREATE TRIGGER TRG_FILME1 FOR FILME
BEFORE INSERT AS BEGIN
  IF (NEW.ID_FILME IS NULL) THEN NEW.ID_FILME = GEN_ID(GEN_FILME, 1);
END
^
 ^
SET TERM ; ^
```



Execute o comando (opção Execute Query) e pronto, o banco de dados está criado.

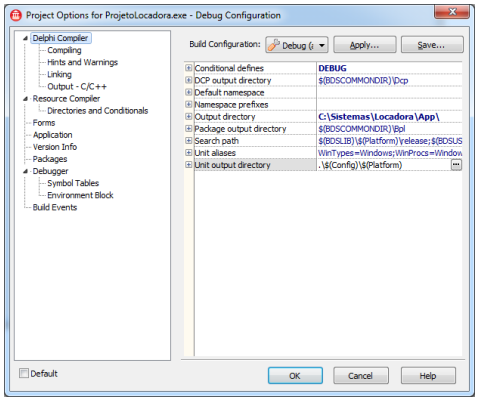
Partindo então para o Delphi, vamos criar nosso formulário principal:



- Caption: Locadora XYZ
- FormStyle: fsMDIChild
- Name: frmPrincipal
- WindowState: wsMaximized

Salve a Unit como UnitPrincipal e o Projeto como Projeto Locadora em *C:\Sistemas\Locadora*

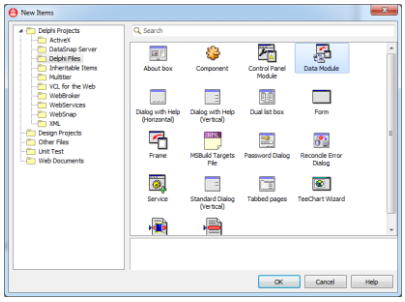
Para que o executável seja criado dentro da subpasta APP siga o seguinte procedimento no Delphi:
- Menu Project/Options, sendo exibida a seguinte tela:



- No lado esquerdo, selecione a opção “Delphi Compiler”.
- no centro, na opção Output directory substitua o conteúdo pelo caminho da subpasta pretendida: **C:\Sistemas\Locadora\App\.**
- Pronto, executando a aplicação (F9), o executável será criado dentro da pasta APP, juntamente com o arquivo do banco de dados DBLocadora.fdb.

CRIANDO O DATA MODULE

Para facilitar, usaremos o Data Module para concentrar os componentes não visuais de Acesso ao Banco de Dados na nossa aplicação:
- Menu File/New/Other, sendo exibida a tela:

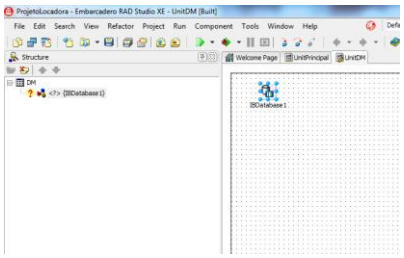


- No lado esquerdo, selecione Delphi Files
- No Centro, opção Data Module.

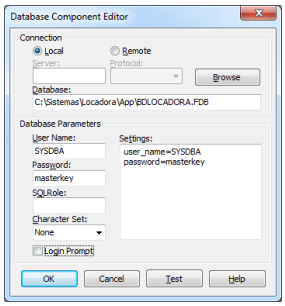
- Criado, altere a propriedade name para DM.

- Salve como UnitDM

Com o Data Module criado, vamos acrescentar os componentes de acesso ao BD.
- Da paleta *Interbase*, vamos usar inicialmente o componente IBDatabase, responsável pela conexão ao BD.



- Para configurar, dê um duplo clique sobre o componente, abrindo a seguinte janela:

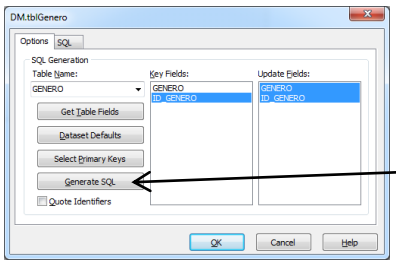


- Em DataBase, vamos selecionar o arquivo do banco de dados, localizado em "*C:\Sistemas\Locadora\App\BDLOCADORA.FDB*". Obs, fz-se necessário na opção de busca do arquivo, escolher no tipo, a opção "Todos os arquivos".
- User_Name = SYSDBA (obrigatoriamente maiúsculo)
- Password = masterkey (minúsculo).
- Desmarcar a opção "Login Prompt".

Depois de feitas as conexões, mude a propriedade Connected para True.

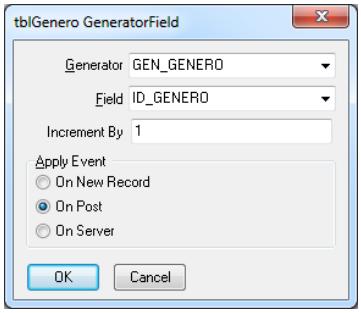
- Próximo passo, vamos acrescentar ao DM o componente IBTransaction também presente na paleta Interbase, alterando suas propriedades: Com Duplo Clique, marque a opção Read Commited.
- No Object Inspector (propriedades no lado esquerdo) selecione na opção "Default Database" o componente IBDatabase1.
- Em active, marque a opção TRUE.

Acrescentaremos agora da paleta Interbase o componente IBDataset, responsável pela conexão a primeira tabela, configurando-o da seguinte forma e sequência:
- Database: IBDatabase1
- Name: tblGenero
- SelectSQL: "*Select * from Genero*"
- Clique com o botão direito sobre o tblGenero, opção "Dataset Editor".



- Em Key Field, selecione apenas o campo que é chave primária (id_genero), mantendo em Update Fields todos os campos selecionados.

- Clique no botão Generate SQL, para gerar os códigos de inserção, alteração e exclusão automaticamente.



Depois, no Object Inspector, vamos alterar a propriedade **Generator Field**, para que seja gerado código automaticamente no campo id_genero ao inserir um registro. Basta configurar como a imagem ao lado.

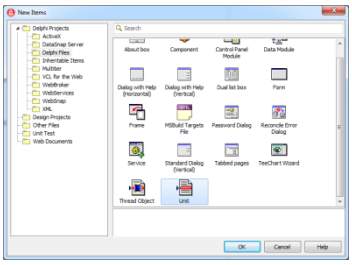
- Prontos, colocaremos a opção Active como True.

Para finalizar a primeira parte do Data Module, vamos adicionar da paleta Data Access, o componente **DataSource**, responsável pela conexão dos componentes do banco de dados com os componentes visuais dos formulários.
No DataSource, apenas configure:
Dataset = tblGenero
Name = dsGenero

Fai Faculdades
Gestão de Tecnologia da Informação
Prof: André Klunk

CRIANDO A UNIT COM OS PROCEDIMENTOS COMUNS DO SISTEMA

Para facilitar nosso trabalho e principalmente, evitar redundância do mesmo código, vamos acrescentar ao nosso projeto uma única específica contendo os procedimentos e funções rotineiras. Para tal, vamos em File/New/Other.



- No lado direito, opção Delphi Files

- No centro, último item: Unit

- Salve a unit como UnitRotinasGerais.pas.

A unit deve ficar exatamente assim, com as seguintes rotinas implementadas:

```
unit UnitRotinasGerais;

interface

uses sysutils, IniFiles,Windows, Forms, Classes, Controls, StdCtrls,
  ExtCtrls, dialogs, DBCtrls, Buttons;

procedure HabilitaCampos (Form: TForm);
procedure DesabilitaCampos (Form: TForm);

implementation

procedure HabilitaCampos (Form: TForm);
var
i : Integer;
begin
for i := 0 to Form.ComponentCount - 1 do
Begin
  if Form.Components[i] is TCustomEdit then
    (Form.Components[i] as TCustomEdit).Enabled:=True;
  if Form.Components[i] is TDBComboBox then
  (Form.Components[i] as TDBComboBox).Enabled:=True;
  if Form.Components[i] is TDBLookupComboBox then
  (Form.Components[i] as TDBLookupComboBox).Enabled:=True;
  if Form.Components[i] is TComboBox then
  (Form.Components[i] as TComboBox).Enabled:=True;
  if Form.Components[i] is TDbCheckBox then
    (Form.Components[i] as TDbCheckBox).Enabled:=True;
  if Form.Components[i] is TGroupBox then
    (Form.Components[i] as TGroupBox).Enabled:=True;
  if Form.Components[i] is TDBRadioGroup then
    (Form.Components[i] as TDBRadioGroup).Enabled:=True;
End;
end;

procedure DesabilitaCampos (Form: TForm);
var
i : Integer;
begin
for i := 0 to Form.ComponentCount - 1 do
Begin
  if Form.Components[i] is TCustomEdit then
    (Form.Components[i] as TCustomEdit).Enabled:=False;
  if Form.Components[i] is TDBComboBox then
  (Form.Components[i] as TDBComboBox).Enabled:=False;
  if Form.Components[i] is TDBLookupComboBox then
  (Form.Components[i] as TDBLookupComboBox).Enabled:=False;
  if Form.Components[i] is TComboBox then
  (Form.Components[i] as TComboBox).Enabled:=False;
  if Form.Components[i] is TDbCheckBox then
    (Form.Components[i] as TDbCheckBox).Enabled:=False;
  if Form.Components[i] is TGroupBox then
    (Form.Components[i] as TGroupBox).Enabled:=False;
  if Form.Components[i] is TDBRadioGroup then
    (Form.Components[i] as TDBRadioGroup).Enabled:=False;
```

```
End;
end;
end.
```

FORMULÁRIO DE CADASTRO DOS GÊNEROS

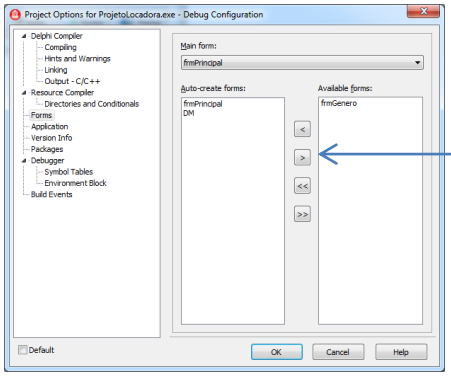
Passaremos agora para a criação do formulário de Cadastro dos Gêneros de filmes da nossa Locadora! Para isso, no Delphi opção File/New/Form – Delphi.

- Salve como UnitGenero

Configure as seguintes propriedades desse novo formulário.

- BorderIcons: somente o biSystemMenu
- Caption: Cadastro de Gêneros
- FormStyle: fsMDIChild
- KeyPreview: true (para poder configurar mais tarde o Enter como Tab)
- Position: poDesktopCenter

Aproveitar, já vamos tirar a criação automática do formulário, indo em Project/Options...



- Lado direito, opção Forms.
- selecione o formulário frmGenero e clique na seta apontando pra direita para enviar na seção Available forms.
- Feito isso, vamos fazer com que o formulário seja de fato destruído quando clicamos no botão fechar. Para tal, selecione o formulário, e no evento OnClose, adicionamos o seguinte código:

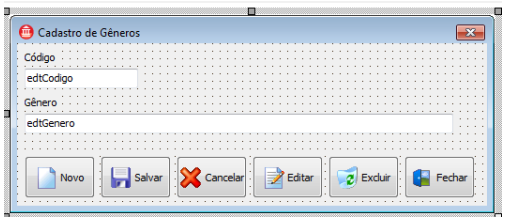
```
procedure TfrmGenero.FormClose(Sender: TObject; var Action: TCloseAction);
begin
  Action := Cafree;
  frmGenero := Nil;
end;
```

Para chamar o formulário de cadastro de Gêneros, voltamos ao formulário principal, onde vamos inserir um Menu Cadastros com duas opção: Gêneros e Filmes.

No evento onClick do menu Gêneros, chamaremos o formulário de cadastro com o seguinte código:

```
procedure TfrmPrincipal.Gneros1Click(Sender: TObject);
begin
  if frmGenero = nil then
    frmGenero := TfrmGenero.Create(Self);
  frmGenero.Show;
end;
```

O formulário de cadastro de Gêneros deve ficar com a seguinte aparência:



Neste formulário colocaremos:

- 2 Label
- 2 dbEdit (da paleta Data Controls)
- 6 bitBtn(paleta Additional)

OBS: Lembre que os botões Salvar e Cancelar devem ter a propriedade inicial **Enabled = False**.

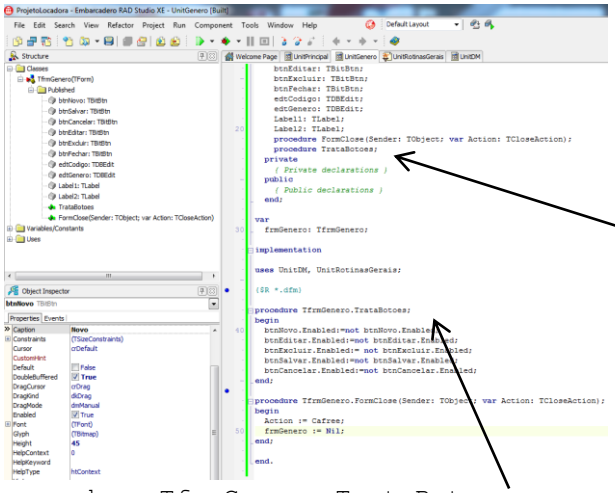
E vamos para a implementação e configuração dos componentes.
Antes de tudo, vamos adicionar a unit do Data Module ao formulário de Gêneros e também a unit Rotinas Gerais. Para tal, vamos no menu **File/Use Unit e selecionamos UnitDM.pas** e depois **UnitRotinasGerais.pas**.
Agora aos componentes:
- etdCodigo:

DataSource = DM.dsGenero
DataField = id_genero
ReadOnly = True (para impedir que o usuário digite o código, uma vez que ele é criado automaticamente pelo generator.
Color = clScrollBar

- etdGenero:

DataSource = DM.dsGenero
DataField = genero
Enabled = False (para que não seja possível edição ao abrir o formulário).

Feito isso, vamos adicionar o procedimento que irá Habilitar ou Desabilitar os botões de cadastro conforme a situação:



- Inicialmente, adicionamos na parte superior a declaração do procedimento, logo acima do termo private:

```
Procedure TrataBotoes;
```

- Após, na parte mais abaixo, lgo depois de {$R *.dfm}, é que inserimos de fato a implementação:

```
procedure TfrmGenero.TrataBotoes;
begin
  btnNovo.Enabled:=not btnNovo.Enabled;
  btnEditar.Enabled:=not btnEditar.Enabled;
  btnExcluir.Enabled:= not btnExcluir.Enabled;
  btnSalvar.Enabled:=not btnSalvar.Enabled;
  btnCancelar.Enabled:=not btnCancelar.Enabled;
end;
```

Evento OnClick dos Botões:

```
procedure TfrmGenero.btnNovoClick(Sender: TObject);
begin
  DM.tblGenero.Append;
  edtGenero.SetFocus;
  TrataBotoes;
  HabilitaCampos(frmGenero);
end;
```

```
procedure TfrmGenero.btnSalvarClick(Sender: TObject);
begin
  DM.tblGenero.Post;
  DM.IBTransaction1.CommitRetaining;
  TrataBotoes;
  DesabilitaCampos(frmGenero);
end;

procedure TfrmGenero.btnCancelarClick(Sender: TObject);
begin
  DM.tblGenero.Cancel;
  TrataBotoes;
  DesabilitaCampos(frmGenero);
end;

procedure TfrmGenero.btnEditarClick(Sender: TObject);
begin
  DM.tblGenero.Edit;
  TrataBotoes;
  HabilitaCampos(frmGenero);
end;

procedure TfrmGenero.btnExcluirClick(Sender: TObject);
begin
  If Application.MessageBox('Tem Certeza de que deseja excluir esse registro
?',
  'Exclusão',mb_YesNo+mb_DefButton2+mb_IconQuestion)
  = Id_Yes then
    Begin
      DM.tblGenero.Delete;
      DM.IBTransaction1.CommitRetaining;
    end;
end;

procedure TfrmGenero.btnFecharClick(Sender: TObject);
begin
  Close;
end;
```

No evento OnShow do Formulário, adicionaremos o código para abrir a tabela Generos.

```
procedure TfrmGenero.FormShow(Sender: TObject);
begin
  DM.tblGenero.Open;
end;
```

No evento OnClose do formulário, adicionaremos acima do código já existente, o comando para fechar a tabela Generos.

```
procedure TfrmGenero.FormClose(Sender: TObject; var Action: TCloseAction);
begin
  DM.tblGenero.Close;

  Action := Cafree;
  frmGenero := Nil;
end;
```

E vamos pros testes....