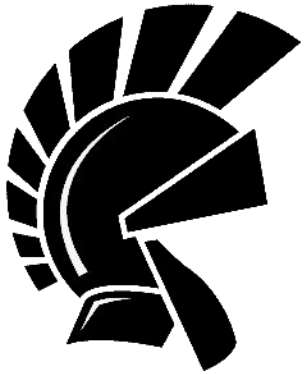


Curso de Delphi

6.ERP Delphi Firebird

Mateus Schwede - UB Social

Conexão Delphi + MySQL

1. Instalar XAMPP
    a. Iniciar XAMPP Apache e MySQL, inserir BD, no Phpmyadmin, com código de 'bancoMySQL.sql'
2. Baixar libmySQL.dll 32bits
    a. Colá-la em xampp/mysql/bin (Windows)
3. Criar projeto Delphi, inserir FDConnection e FDPhyMySQLDriverLink
    a. FDPhyMySQLDriverLink, propriedade VendoLib, informar url de libMysql.dll(C:\xampp\mysql\bin\libmySQL.dll)
    b. FDConnection, 2 cliques, informar:
        - (Driver ID: MySQL)
        - Database: agenda
        - User_Name: root
        - Server: localhost
        - Port: 3306

bancoMySQL.sql

```
CREATE DATABASE agenda;
USE agenda;

CREATE TABLE pessoa (
        id INTEGER AUTO_INCREMENT PRIMARY KEY,
        nome VARCHAR(50) NOT NULL
);

INSERT INTO pessoa(nome) VALUES ("ub1"),("ub2"),("ub3");
```

CRUD MySQL

1. FDConnection
   (Duplo click nele, informar BD (FB se Firebird) e parâmetros)

2. FDPhyMySQLDriverLink
   VendorLib: C:\xampp\mysql\bin\libmySQL.dll

3. FDTable
   TableName (Informar nome da table)
   Active: Marcar true

4. DataSource
   DataSet: nomeFDTable

5. DBGrid
   DataSource: nomeDataSource

6. DBNavigator
   DataSource: nomeDataSource

Delphi + Firebird

Firebird e FlameRobin

1. Instalar Firebird 2.5.0 (32bits) e, na instalação, selecionar opção “mover fpclient.dll para pasta System”
    a. Caso necessário em algum momento, o Firebird usa porta 3050

2. Instalar FlameRobin 32bits
    a. (Registrar novo Server) Configurar BD com User(SYSDBA) Senha(masterkey), Charset(Win1252) e demais parâmetros pessoais

Conexão e criação BD

1. Baixar apostila Delphi BD
2. No FlameRobin, localhost botão direito ‘Create New Database’
   a. Name: Loja
   b. Database Path: Selecionar pasta do projeto, informar nome do novo arquivo ‘dbLoja’
   c. User: SYSDBA
   d. Password: masterkey
   e. Charset: Win1252
3. Copiar código do arquivo ‘bancoFirebird.sql’, no FlameRobin > Tables > (Botão direito) Create New > Colar código > Execute Statement > Commit (F5)
   a. Salvar arquivo como ‘dbLoja.fdb’ na pasta do projeto
   b. Duplo click sobre a table, aparecerá mensagem de erro (Clicar em cancelar), serão mostrados os dados da table

Firebird + Delphi

1. No novo projeto VCL, ir em New > Other > Database > Data Module
    a. Salvar como Data Module como ‘uDM’
    b. Salvar Unit principal como ‘uMenu’

2. Data Module (Propriedade Name: ‘dm’)
    a. FDConnection (Duplo click para abrir conexão)
        i. Driver: FB
        ii. Database: Selecionar arquivo dbLoja.fdb
        iii. User: SYSDBA
        iv. Password: masterkey
        v. Protocol: local
        vi. Charset: Win1252
        vii. Testar conexão (Se erro, então informar Port 3050)
    b. Connected: True
    c. LoginPrompt: False

Cada DataModule representa um setor de base de dados da empresa

Firebird + Delphi

3. FDTable
    a. Connection: (Já pega connetcion automaticamente)
    b. TableName: Selecionar Clientes
    c. Name: tbClientes
    d. Active: False (Conectaremos manualmente via código)
    e. (Duplo click sob o componente, abrirá janela de field, click com botão direito e selecionar ‘add all fields’)

4. DataSource
    a. Name: dsClientes
    b. DataSet: tbClientes

5. Criar um FDTable e respectivo DataSource para todas as demais tables do BD

Firebird + Delphi

6. Criar MainMenu com acesso para nova Unit (uClientesTeste)
7. File > Use Unit > uDM
    a. (Todas Units precisam acessar o DataModule para conexão com DB)
8. Na Unit uClientesTeste
    a. Evento OnShow, inserir código ‘dm.tbClientes.Open;’
    b. Evento OnClose, inserir código ‘dm.tbClientes.Close;’
    c. DBGrid
        i. DataSource: dsClientes
    d. DBNavigator
        i. DataSource: dsClientes

ERP
Delphi Firebird

Form Clientes

1. Criar Unit VCL Form uClientes, link com uDM (Use Unit)
   a. No uDM, tbClientes, desabilitar propriedade AutoEdit
   b. No MainMenu do uMenu, inserir opção para acessar uClientes
   c. No uClientes, OnShow e OnClose: dm.tbClientes.Open; dm.tbClientes.Close;
2. No uDM, seleciona janela com all fields da tbClientes e arrastá-los, com o mouse, para dentro a uClientes
   a. Nesta janela, a cada form alterar propriedade DisplayLabel, com o nome organizado
      i. Edit Código: Marcar property ReadOnly
      ii. Telefone: editMask: Phone (Selecionar um deles e personalizar), desmarcar Save Literal Characteres
      iii. CEP: editMask: CEP (Personalizar últimos somente 3 dígitos), desmarcar Save Literal Characteres
3. Inserir DBNavigator, deixar visíveis somente nbFirst, nbPrior, nbNext, nbLast
   a. DataSource: dm.dsClientes
4. Criar bitBtn Inserir, Editar, Excluir, Cancelar, Confirmar e Sair (Cancelar e Confirmar disables)
   a. Para inserir ícones nos botões, basta fazer upload do mesmo na propriedade Glyph

Procedure habilitar/desabilitar botões

- Abaixo de private {Private declarations} informar Procedure TratarBotoes;
- Abaixo dos procedures dos demais botões, informar procedure:

```
procedure TfrmClientes.TratarBotoes;

begin

  btnInserir.Enabled := not btnInserir.Enabled;

  btnEditar.Enabled := not btnEditar.Enabled;

  btnExcluir.Enabled := not btnExcluir.Enabled;

  btnConfirmar.Enabled := not btnConfirmar.Enabled;

  btnCancelar.Enabled := not btnCancelar.Enabled;

  DBNavigator1.Enabled := not DBNavigator1.Enabled;

end;
```

Botão inserir

```
procedure TfrmClientes.btnInserirClick(Sender: TObject);
var
  prox: Integer;
begin
  TratarBotoes;
  dm.tbClientes.Last;
  prox := dm.tbClientes.FieldByName('cdcliente').AsInteger + 1;
  dm.tbClientes.Append;
  dm.tbClientes.FieldByName('cdcliente').AsInteger := prox;
  DBEdit2.SetFocus;
end;
```

Botão editar

```
procedure TfrmClientes.btnEditarClick(Sender: TObject);

begin

  TratarBotoes;

  dm.tbClientes.Edit;

end;
```

Botão cancelar

```
procedure TfrmClientes.btnCancelarClick(Sender: TObject);

begin

  dm.tbClientes.Cancel;

  TratarBotoes;

end;
```

Botão confirmar

```
procedure TfrmClientes.btnConfirmarClick(Sender: TObject);
begin
  if dm.tbClientes.FieldByName('DCCLIENTE').AsString = '' then
  begin
    MessageDlg('Informe o nome!', mtWarning, [mbOk], 0);
    DBEdit2.SetFocus;
    Exit;
  end;
  dm.tbClientes.Post;
  TratarBotoes;
end;
```

Botão excluir

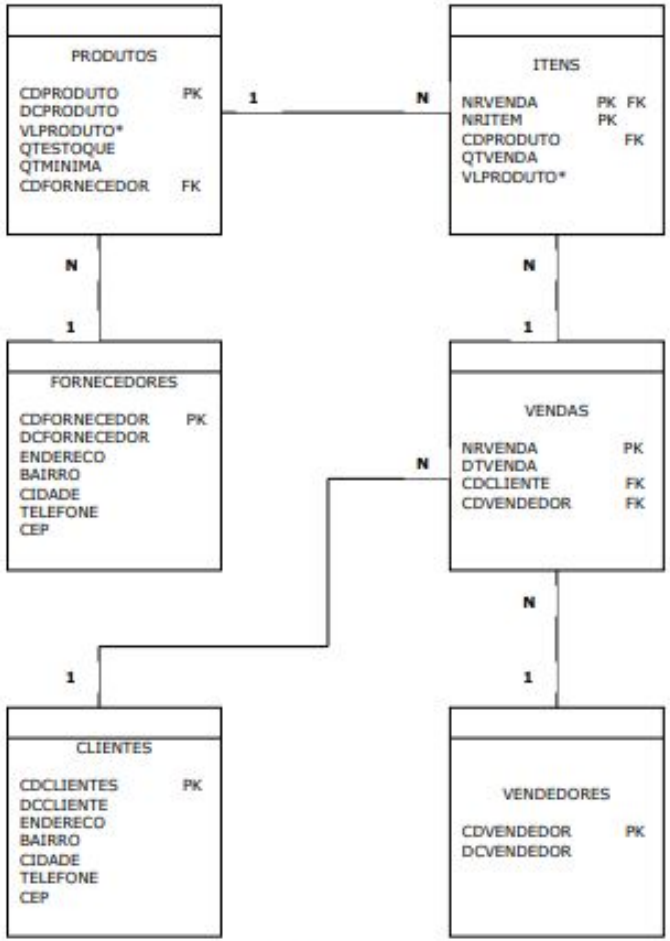
```
procedure TfrmClientes.btnExcluirClick(Sender: TObject);
begin
  if MessageDlg('Excluir?',mtConfirmation,[mbYes,mbNo],0) = mrYes then
  begin
    dm.tbClientes.Delete;
  end;
end;
```

Botão sair

Informar, em uses Data.DB

```
procedure TfrmClientes.btnSairClick(Sender: TObject);
begin
  if dm.tbClientes.State IN [dsInsert,dsEdit] then
  begin
    MessageDlg('Cancele ou confirme',mtWarning,[mbOk],0);
    Exit;
  end;
  Close;
end;
```

Firebird FK



PK = Primary Key = Chave Primária
FK = Foreign Key = Chave Estrangeira

*pode parecer redundância mas não é, uma vez que poderá haver alteração do valor do produto lançado no momento da venda em relação ao período atual.

Preparo

- Para cadastrar produtos, precisa-se de um fornecedor cadastrado
  - Integração entre tabelas
  - Realizar, anteriormente, cadastro de fornecedor

Integração FK

- Ao invés de mostrar o código do fornecedor, mostrará o nome. Mas, na base de dados, será salvo seu código, conforme exigido na table
- Form uProdutos:
  - OnShow: dm.tbProdutos.Open; dm.tbFornecedores.Open;
  - OnClose: dm.tbProdutos.Close; dm.tbFornecedores.Close;
- Realizar tela de cadastro de produto, excluir o campo de código do fornecedor
  - Ao seu lugar, inserir LookupComboBox
    - DataSource: dsProdutos (Onde salvará o registro)
    - DataField: CDFORNECEDOR (Campo que o registro será salvo)
    - ListSource: dm.dsFornecedores (Origem da informação que será mostrada)
    - ListField: DCFORNECEDOR (Campo que será mostrado)
    - KeyField: CDFORNECEDOR (Campo que relaciona as tables)

UB Social

ubsocial.github.io