**Eficiência x Eficácia**

**Mário Leite**

...

Criar um programa para calcular uma simples **Média Escolar** de um aluno ou definir os autovetores da **Equação de Schrödinger** (da Mecânica Quântica), embora possa parecer esquisito fazer essa comparação, na verdade, ambas as situações passam pela mesma análise prévia: “Lógica da Programação”. Não interessa o nível de complexidade de um programa se o programador não *planejar*, *pensar* e *aplicar a lógica* para criar a soluça que agrade ao usuário; não adianta conhecer a linguagem da “moda”, pois, o problema pode não ser resolvido de maneira **eficaz** e satisfatória para o usuário. E a Lógica pode ser estendida a qualquer particularidade do sistema como, por exemplo, no controle de dados com um SGBDR, tendo como base uma 4GL (SQL a mais usada) para *inclusão*, *edição*, *exclusão* e, principalmente, *pesquisas*. De qualquer forma, tudo passa pela lógica de como o programador vai planejar a solução que, neste caso, começa com o **MER** (**M**odelo **E**ntidade-**R**elacionamento). A **figura 1** mostra um esquema desse modelo ***Cliente-Compra-Produto*** de um banco (no caso, Sql Server CE) onde os símbolos **#** e **\*** representam a chave primária e a chave estrangera, respectivamente. O código do *click* no botão **[Exibir]** é mostrado abaixo, para criar uma listagem das compras dos produtos efetuadas pelos clientes, tabulada dentro de uma *ListBox*. Observe que, mesmo não sendo atributos da tabela “Compras”, o nome do produto e o nome do cliente saíram na listagem; isto porque foi seguida a LÓGICA mostrada no MER, baseando nas chaves estrangeiras desta tabela. E poderia até mostrar apenas os produtos de determinada característica bastando, apenas, a inclusão da cláusula restritiva **WHERE** na instrução SQL, aumentando a Eficiência; MAS, a **Eficácia** continuaria na lógica ditada pelo MER e não pela codificação.

-------------------------------------------------------------------------------------------------------------------------

Para adquirir *pdf de* livros sobre Programação, entre em contato: **marleite@gmail.com**

-------------------------------------------------------------------------------------------------------------------------

**using** System.Data.SqlServerCe; //biblioteca para acesso a bancos **SQL Server CE**

**private** **void** btnExibir\_Click(**object** sender, **EventArgs** e)

{

**string** strConn = "Data Source=" + "D:\\Livros\\Livro10\\Dados\\SistCompras.sdf";

**SqlCeConnection** objConn = new **SqlCeConnection**(strConn); //cria o objeto de conexão

**try**

{

objConn.Open(); //faz a conexão: tenta abrir o banco de dados

}

**catch** (**Exception**)

{

MessageBox.Show("Falha na conexão!");

**return**; //sai da rotina e retorna ao início

}

**DataTable** objTab = new **DataTable**(); //cria objeto de tabela

//Define a instrução SQL

**string** strSelect =

"**SELECT** Compras.compNum **AS** Compra, Compras.compData **AS** Data, " +

" Compras.compQuant **AS** Unidades," +

" Compras.compValor **AS** Valor, Produtos.prodDesc **AS** Produto, " +

" Clientes.cliNome **AS** Cliente" +

" **FROM** Compras" +

" **INNER JOIN** Produtos **ON** Compras.prodCod=Produtos.prodCod" +

" **INNER JOIN** Clientes **ON** Compras.cliCod=Clientes.cliCod";

**SqlCeDataAdapter** obgAdapt = **new** **SqlCeDataAdapter**(strSelect, objConn);

obgAdapt.Fill(objTab);

objConn.Close(); //desfaz a conexão: fecha o banco de dados

//Apresenta os dados

lstCompras.Items.Clear(); //Limpa a ListBox previamente

**var** colunas = **String**.Format("{0,6}{1,9}{2,14}{3,12}{4,10}{5,43}\n",

"Compra", "Data", "Unidades", "Valor(R$)", "Produto", "Cliente");

lstCompras.Items.Add(colunas);

**var** campoCompra = 0;

**string** campoData;

**string** campoUnidades;

**double** campoValor;

**string** campoProduto;

**string** campoCliente;

**foreach** (**DataRow** linha **in** objTab.Rows)

{

campoCompra = **Convert**.ToInt32(linha["Compra"]);

**string** strCompra = campoCompra.ToString();

**if** (campoCompra<10) {

strCompra = "0" + campoCompra.ToString();

}

campoData = linha["Data"].ToString();

campoUnidades = linha["Unidades"].ToString();

campoValor = (**double**)linha["Valor"];

campoProduto = linha["Produto"].ToString().Trim();

campoCliente = linha["Cliente"].ToString().Trim();

lstCompras.Items.Add(**String**.Format("{0,-9}{1,-15}{2,-9}{3,-11}{4,-43}{5,-35}",

strCompra, campoData, campoUnidades, (**double**) campoValor,

campoProduto, campoCliente));

}

}



