

Escola Técnica Estadual “São Paulo” - ETESP

Linguagem C# - Visual Studio 2019

Banco de Dados

Programas são desenvolvidos para manipular dados e gerar informações, contudo, na maioria das vezes estes dados podem mudar no decorrer do tempo (dados cadastrais de um cliente e/ou fornecedor, o estoque de mercadorias, o valor unitário, preço de venda, o peso de um paciente...), por esta razão devemos evitar, o máximo possível, a inclusão de “dados/infomações” dentro da escrita de códigos de programação. Estes dados devem estar armazenados externamente e, quando necessário, o programa deverá acessá-los, manipulá-los e, se for o caso, fazer a atualização deles, para posterior utilização.

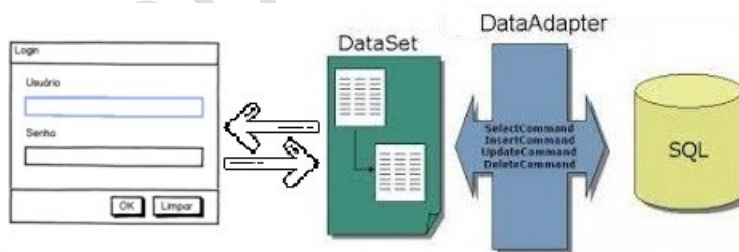
ADO.Net é a atual tecnologia de acesso a dados presente no ambiente “.Net”. Esta tecnologia possui dois modelos de conexão: o modelo conectado e o modelo desconectado.

MODELO CONECTADO

É a maneira mais comum de acessar uma base de dados. A proposta consiste em, todas as vezes que precisamos de alguma informação desta base, nos conectamos a ela e carregamos a informação necessária. Esse modelo nos permite sempre termos a informação o mais atualizada possível e é ideal para sistemas em que os dados mudam com muita frequência e que precisam das informações atualizadas. O problema deste modelo é o fato de que todas as vezes em que for necessário recuperar uma informação, será preciso se conectar a base de dados. Isso causa um tráfego maior na rede deixando-a sobrecarregada.

MODELO DESCONECTADO

Há aplicações em que as informações não mudam com tanta frequência e para estes casos poderemos utilizar uma maneira diferente de obtê-las, economizando recursos, estamos falando do “modelo desconectado”.



No modelo desconectado, conecta-se uma vez ao banco, recupera os dados para a memória e utiliza-se este “banco de dados” na aplicação, evitando-se, desta forma, fazer (ou manter) a conexão com o banco de dados por tempo excessivo. Caso seja necessária alguma atualização, a modificação poderá ser feita nesse “banco na memória” e depois atualiza-se o “banco de dados físico” com essas novas informações. Essa atualização pode ser feita a qualquer momento.

Percebe algum problema nisso?

O maior problema que ocorre no modelo desconectado é a preocupação com consistência dos dados. Por exemplo: Dois usuários, no início do dia, carregam na memória dos seus PCs um subconjunto igual do banco de dados e saem para trabalhar. Ao final do dia, os dois chegam para atualizar os dados no banco de dados. Se os dois tiverem modificado os mesmos registros, mas com dados diferentes... de quem será a preferência? De quem chegou primeiro? Do dado mais novo? Isso depende da política da empresa e deve ser bem pensado em modelos de acesso desse tipo.

Prof. Roberto de Castro

PRINCIPAIS CONCEITOS

- **Connection** (string de conexão) → utilizado em qualquer tipo de acesso ao SGBD, sendo ele conectado ou não, estabelece a conexão com o banco de dados.
- **Dataset** → armazena dados de um banco de dados, em memória, de maneira que possam ser acessados pela aplicação.
- **Data Adapter** → Mecanismo “intermediário” entre o “Banco de Dados físico/real” e o “Banco de Dados Virtual” (funciona como um “conector”). Utilizado para ler/gravar dados em um banco de dados.
- **DataBind** → Vínculo/exibição dos dados a um controle, por exemplo: Label.

PROVEDORES (PROVIDERS)

Os provedores de dados oferecem os mecanismos (funcionalidades) necessários para se conectar ao banco de dados. Os mais comuns, que já vêm com o .NET são:

- SQL Server .NET Data Provider → Utilizado na conexão com bancos de dados SQL Server. Para acessá-lo utilizamos o namespace “System.Data.SqlClient” na escrita do código.
- OleDb .NET Data Provider → É usado comumente para acesso a bancos de dados em Access. Para acessá-lo utilizamos o namespace “System.Data.OleDb”.
- Oracle .NET Data Provider → Usado para conexão com bancos de dados Oracle. Deve ser baixado e instalado. Após a instalação estará presente no namespace “System.Data.OracleClient”.

Desenvolvimento de Aplicações

Vamos colocar em prática os conceitos vistos até o momento??

Construiremos três tipos de aplicações com acesso a Banco de Dados.

No primeiro exemplo, utilizando os “assistentes do Visual Studio” e uma base de dados “Access”, reforçaremos alguns conceitos teóricos (acesso desconectado, DataSet, ConnectionString...). Utilizaremos pouquíssimas linhas de código para desenvolver este tipo de projeto.

Para o segundo exemplo, também utilizando o conceito de “acesso desconectado”, sem a utilização dos assistentes e com “algumas linhas de código” para manipular os dados no banco.

O último modelo de acesso a banco de dados, utilizaremos as técnicas de programação orientada a objeto e acesso ao banco utilizando o método de acesso “conectado”.

Ao longo do desenvolvimento dos projetos veremos a utilização de menus de opções, formulários MDI.

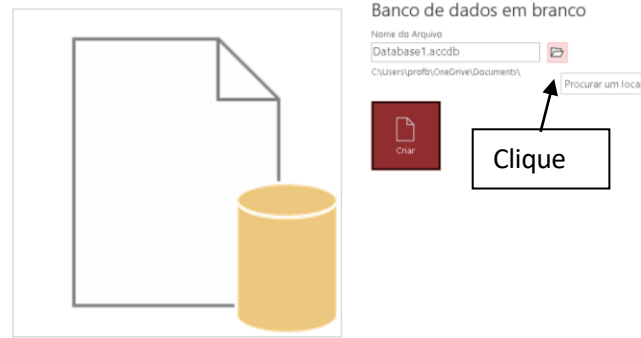
Pronto?

O primeiro passo será a criação de nossa “base de dados”. Utilizaremos o “Access” para gerar a estrutura de nossa tabela.

Importante: Vamos criar uma pasta chamada “BancoDados” e gravar o banco dentro desta pasta! Defina o local de sua preferência.

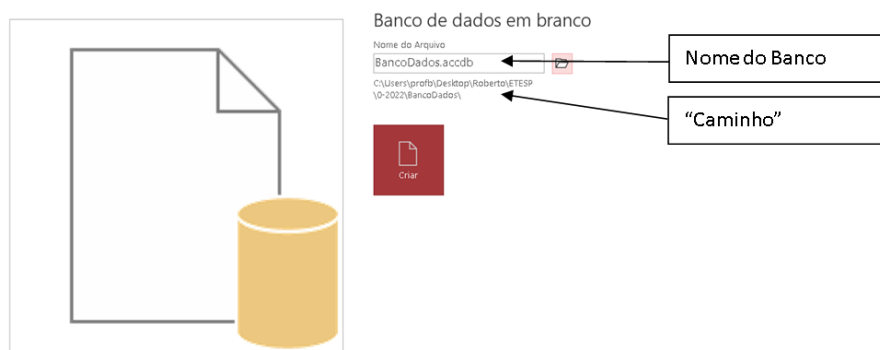
Apresento abaixo, um “breve” passo-a-passo...

1. Na primeira tela exibida pelo “Access” clique em “Banco de Dados em branco”;
2. Na próxima tela, **ANTES DE CLICAR** na opção “criar”, dê um clique na opção “Procurar um local...”, para indicar o caminho onde você irá gravar o Banco de Dados (veja a imagem abaixo):



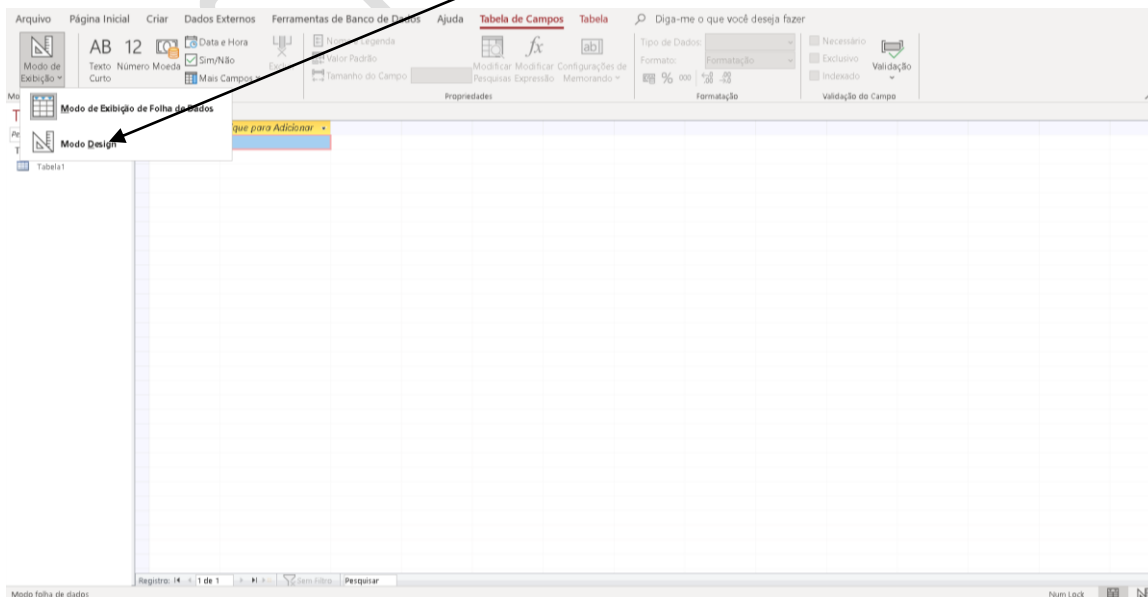
Vá até o local onde você criou a pasta “BancoDados”, clique em “abrir”, digite o nome do banco de dados (para este exemplo o nome do banco será “BancoDados”) e clique em “OK”.

O “Access” retornará para a mesma tela anterior, porém, com o do nome do banco de dados e o “caminho” onde será gravado. Observe:



Se tudo estiver “ok”, clique na opção “Criar” (botão vermelho).

3. Altere o “modo de exibição” para “Modo Design”.



Prof. Roberto de Castro

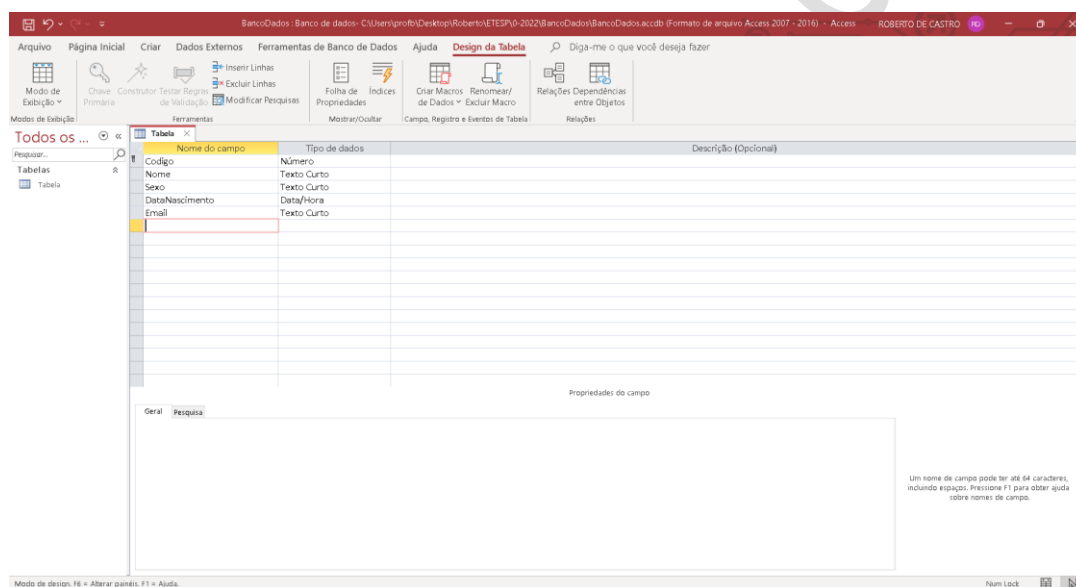
4. O nome da tabela será “Tabela”. Clique em “ok”.

5. Estrutura da tabela:

Campo	Tipo	Observação
Codigo	Número	Campo Chave
Nome	Texto Curto	Tamanho 50; Requerido
Sexo	Texto Curso	Tamanho 1; Requerido; Regra de validação “M” ou “F”
DataNascimento	Data/Hora	Requerido
EEmail	Texto Curto	Tamanho 255

IMPORTANTE: Na escrita dos “nomes dos campos” evite a utilização de acentos, “ç” e espaços entre os nomes compostos, por exemplo: “Data de Nascimento”, substitua por “DataNascimento”.

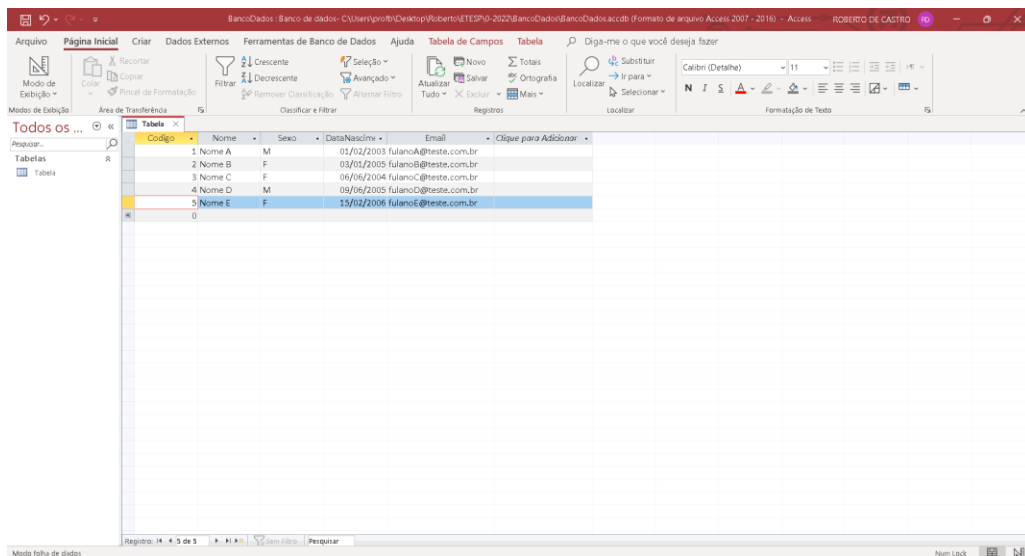
Ao final da digitação dos campos, teremos a imagem:



Agora, vamos cadastrar alguns dados. Altere o modo de exibição para “Folha de Dados” (O Access solicitará para gravar a sua estrutura, clique em SIM) e cadastre pelo menos 5 registros...

Dica: Incluímos algumas “validações” nos campos, por exemplo: nome é obrigatório (requerido), o sexo aceitará somente as letras “M” ou “F”. Faça os testes, deixe o nome em “branco”, digite a letra “L” no sexo e perceba como o “Access” se comportará.

Após cadastrar os cinco registros, a imagem deverá ser:

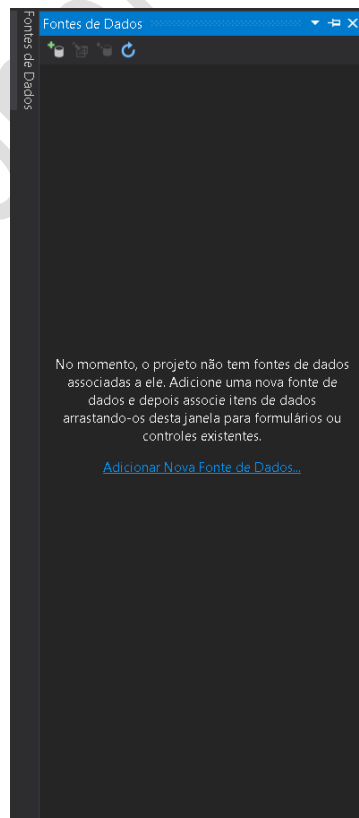


6. Nosso trabalho no "Access" está concluído. Clique no menu "Arquivo" → "Fechar" e finalize o "Access".

Iniciaremos agora, a construção de nossa aplicação no Visual Studio.

Após carregar o Visual Studio e criar uma "nova aplicação Windows Forms", precisamos tornar "visível" a caixa de opções "Fonte de Dados" (Data Source).

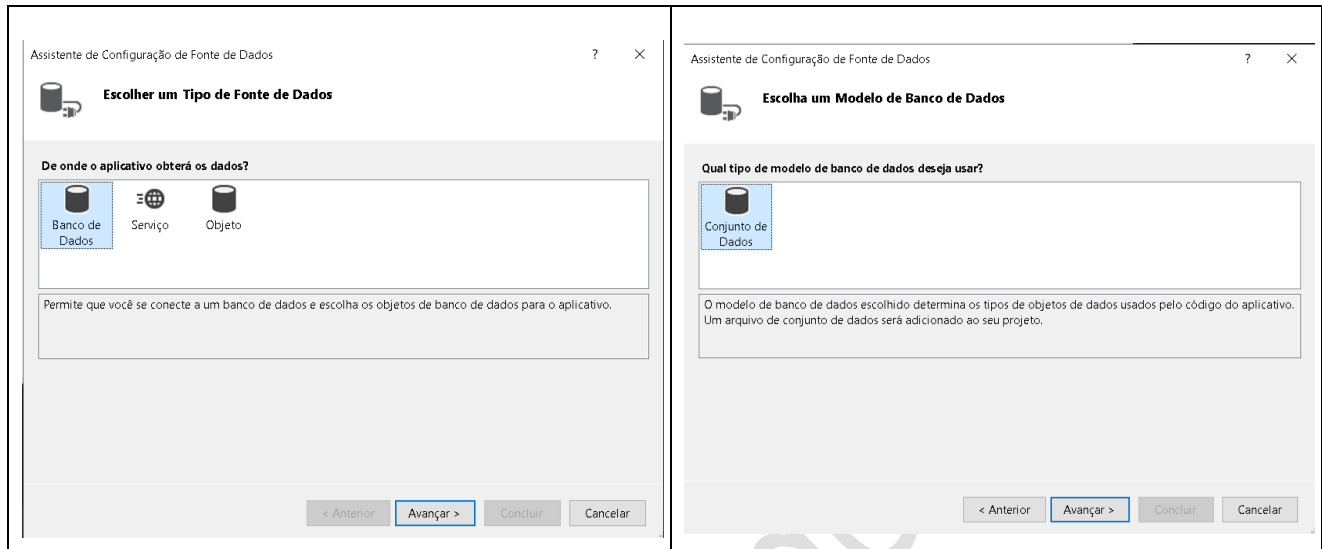
Clique em View (Exibir) → Outras Janelas (Other Windows) → Fontes de Dados (Data Source). A "janela" será exibida:



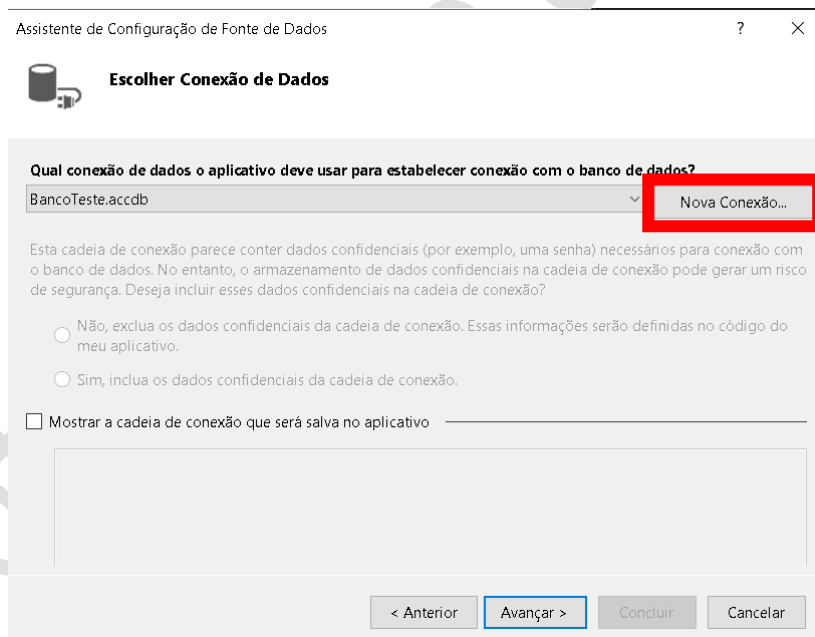
Prof. Roberto de Castro

A aplicação não identificou nenhuma fonte de dados no projeto atual e solicita para “Adicionar Nova Fonte de Dados”. Clique no link que está em “azul”.

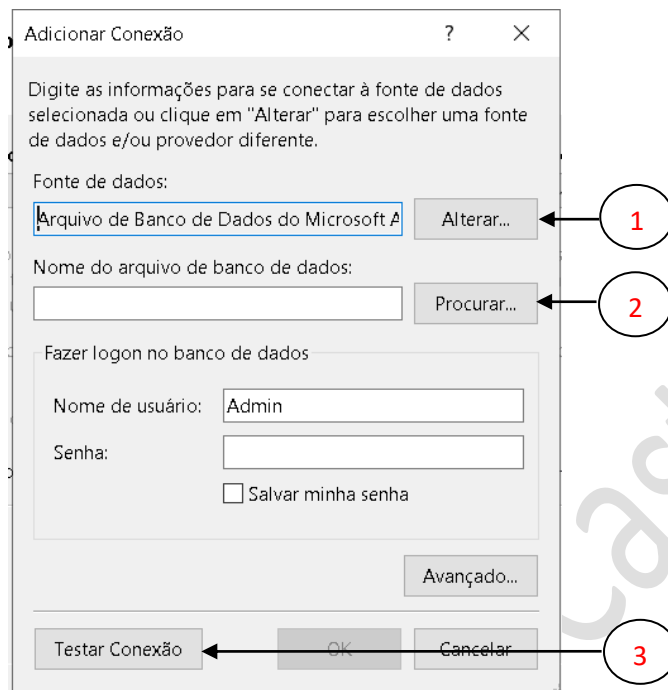
Nas duas proximas etapas, clique em avançar, pois o “padrão” já está selecionado:



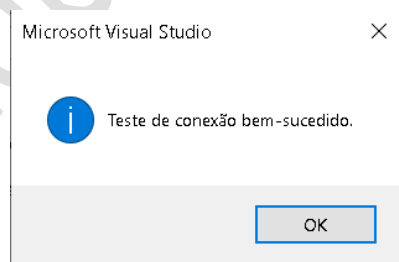
Na próxima tela, clique em “Nova Conexão”:



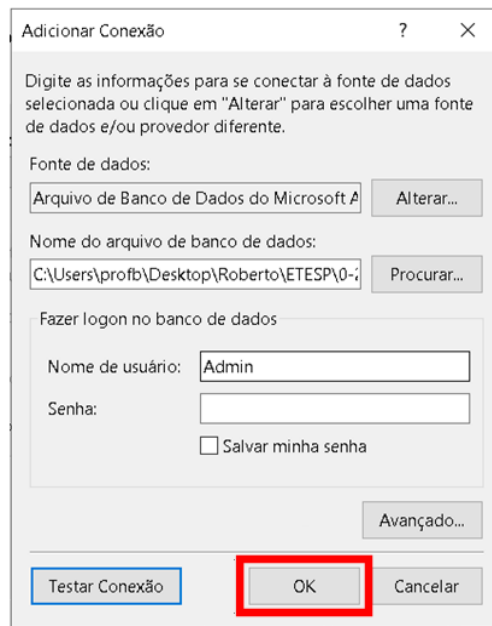
A seguir executaremos três etapas...



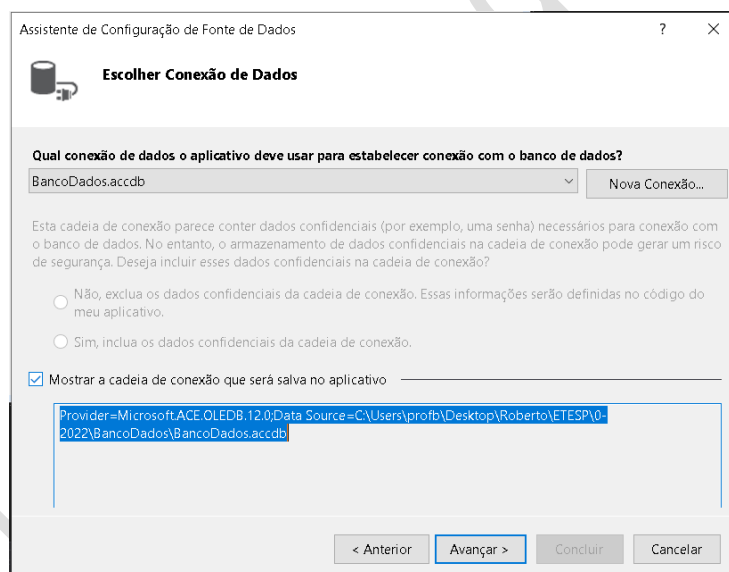
- 1) A primeira é informar o tipo de “fonte de dados” (provider). Nossa fonte de dados, para este nosso projeto, será o “Access”. Por padrão, esta opção já está assinalada, porém, caso seja necessário alterar, basta clicar em “Alterar”, o que, no momento, não é o caso, vamos manter o padrão.
- 2) A segunda etapa é informar o “nome do banco de dados”. Clique em procurar e “mostre” o caminho onde o banco está gravado.
- 3) A terceira e última etapa, é fazer o “Teste de Conexão” para verificar a integridade do banco. Clique em “Testar Conexão”. Se o banco estiver “ok”, o sistema exibirá a mensagem:



Clique em “ok” para fechar a mensagem exibida e clique novamente em “ok” para fechar a janela do “Assistente de Conexão”:

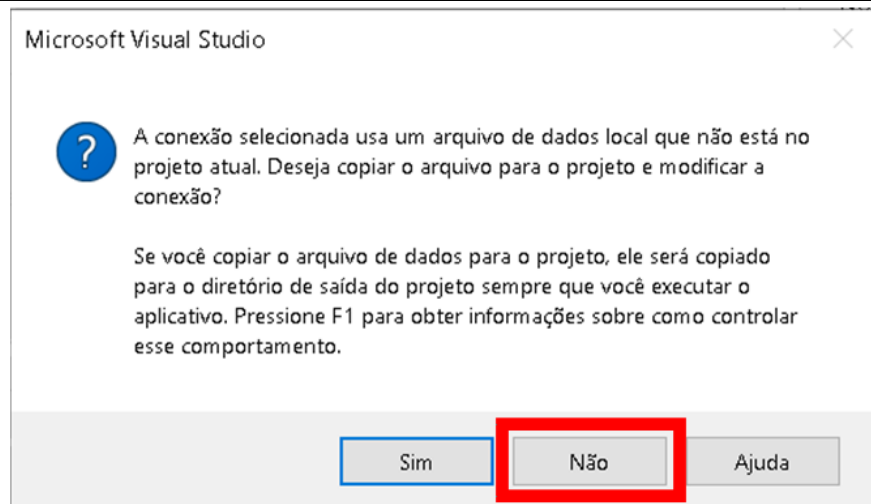


O assistente será “fechado” e retornaremos à tela inicial. Neste momento selecionamos a caixa de seleção: “Mostrar a cadeia de conexão....” para verificar o que foi realizado até o momento:



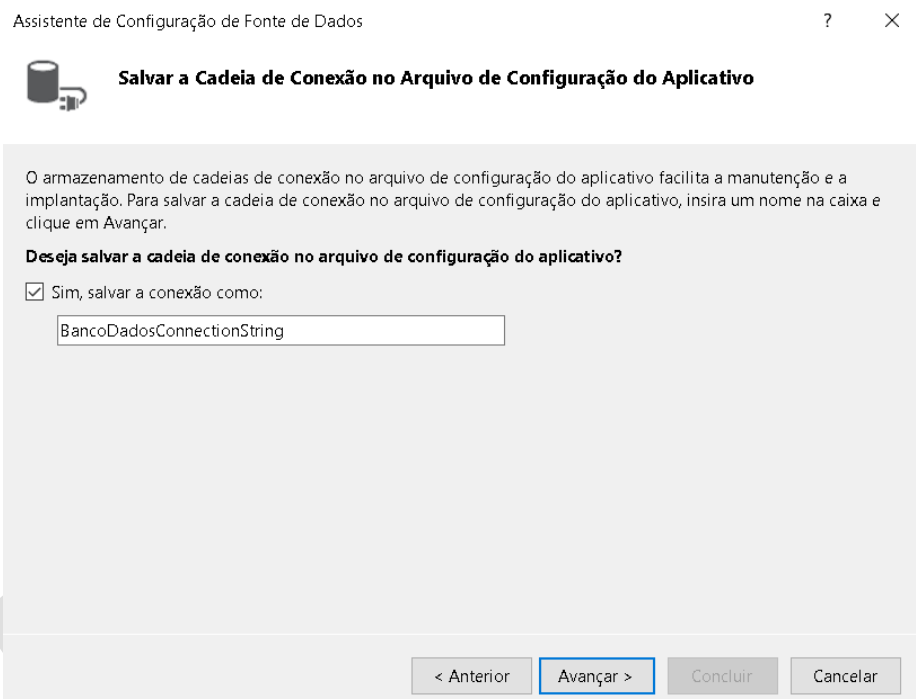
Em resumo, este texto, chamado de “Connection String” (texto de conexão), contém: o nome do provedor (provider), o caminho e o nome do banco de dados. Mais adiante veremos como alterar esta “connection string” caso ocorra alguma alteração (por exemplo: caminho do banco foi alterado, nome do banco foi alterado....). No momento, clique em “Avançar”.

ATENÇÃO: O assistente exibirá nova mensagem informando que a conexão utiliza uma base de dados que está “fora” da pasta de projeto da aplicação e pergunta se você deseja “copiar” o arquivo para a pasta do projeto. **Clique em NÃO**, pois não desejamos fazer uma cópia (esta opção será útil para o caso de uma aplicação “real” fazendo testes em um grande banco de dados verdadeiro), mas para o nosso caso o que desejamos é acessar o banco SEM FAZER A CÓPIA!!

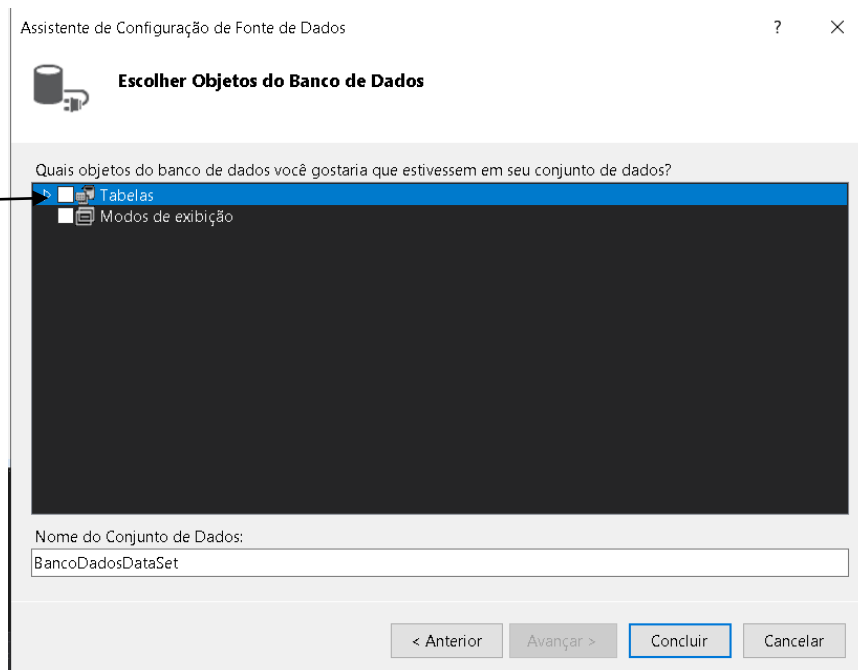


Estamos chegando ao final...

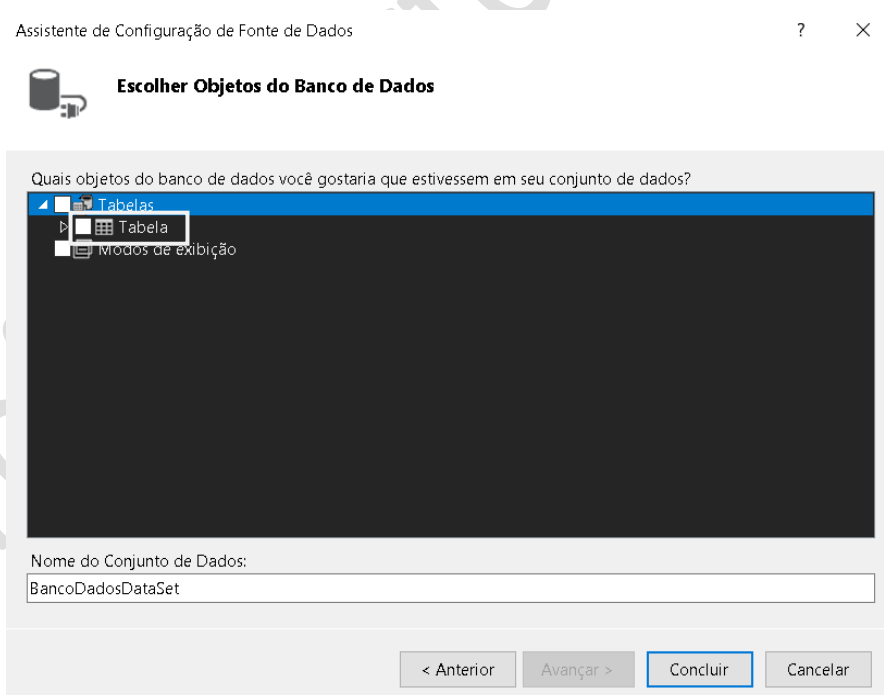
Na próxima tela o assistente informa que a “Connection String” será salva no arquivo de configuração (veremos este arquivo em breve) e exibe o nome desta “Connection String” (BancoDadosConnectionString). Vamos manter este nome e clicar em “Avançar”.



Muita atenção na próxima etapa, pois antes de clicar no botão CONCLUIR, **DEVEMOS** (veja que eu escrevi DEVEMOS) selecionar quais serão as informações que a nossa aplicação irá tratar (o assistente criará os comandos SQL com base nesta seleção que devemos fazer, daí a importância).



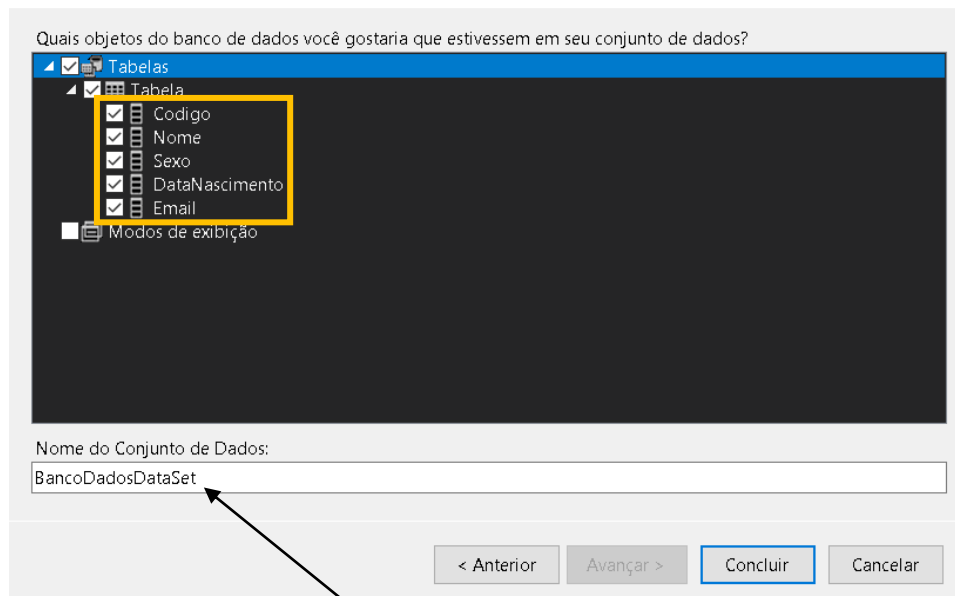
Dê um clique na “seta” à esquerda da palavra “Tabelas” (ou Tables se inglês). O assistente exibirá TODAS as tabelas existentes em nosso banco de dados, porém, para este nosso projeto, como o banco possui somente UMA tabela, será exibida somente uma...



Clique novamente na “seta” à esquerda da palavra “Tabela”, para o assistente exibir os “campos” existentes nesta tabela:



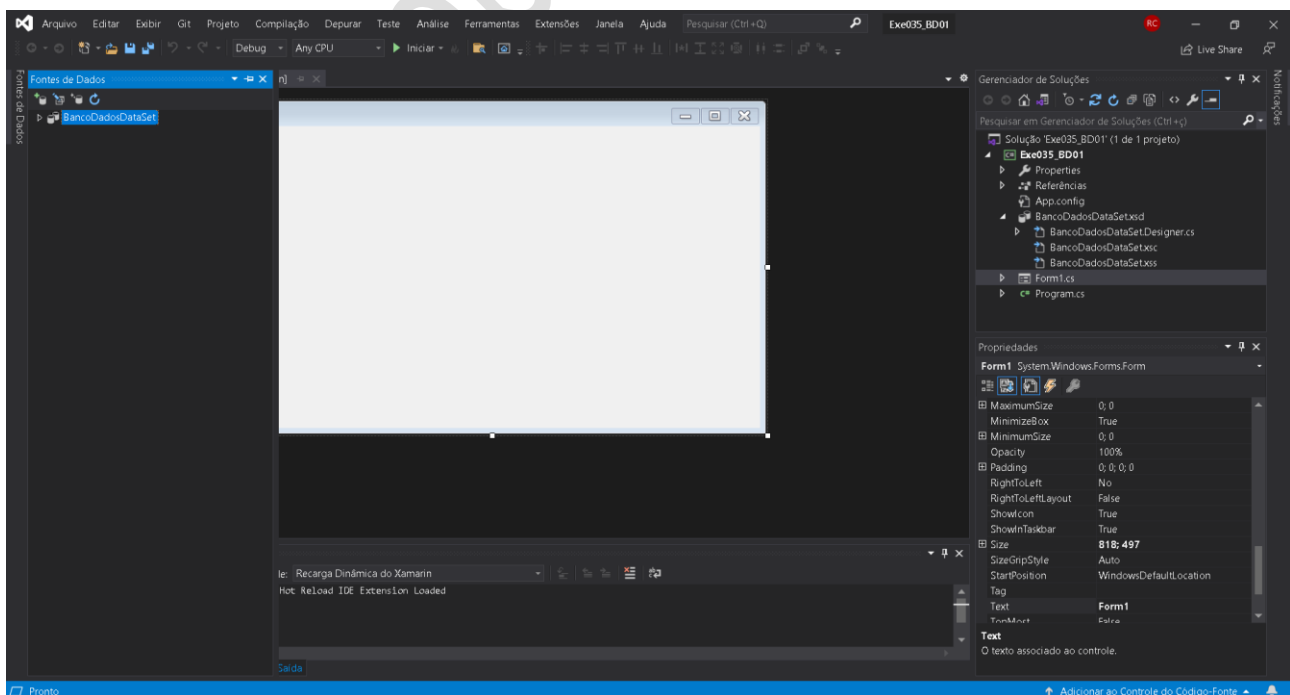
Escolher Objetos do Banco de Dados



IMPORTANTE: Selecione todos os campos (conforme destacado na imagem acima) e observe que o assistente está exibindo uma informação: **BancoDadosDataSet**. Este é o nome que será utilizado na programação para referenciar o banco de dados “desconectado”, na realidade trata-se de uma “cópia”, em memória, dos dados existentes no banco de dados original.

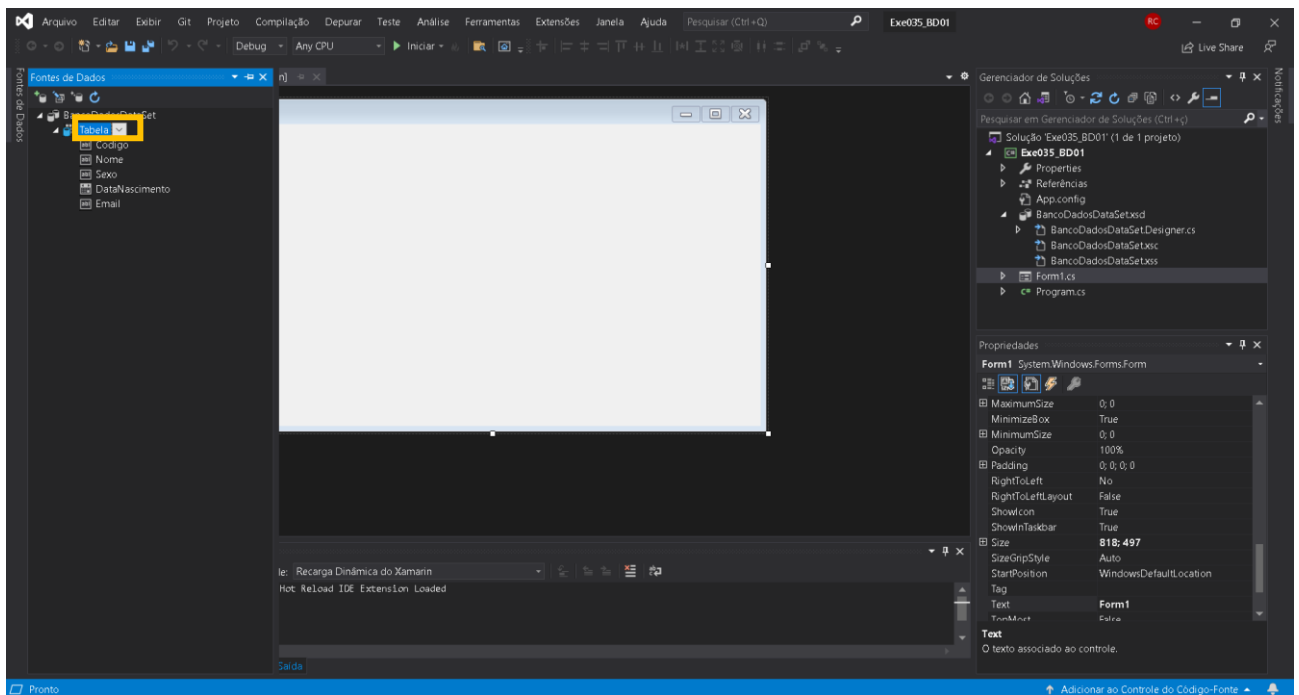
E para finalizar, clique em “CONCLUIR”!!

A janela “fonte de dados”, agora deverá estar conforme a imagem abaixo:

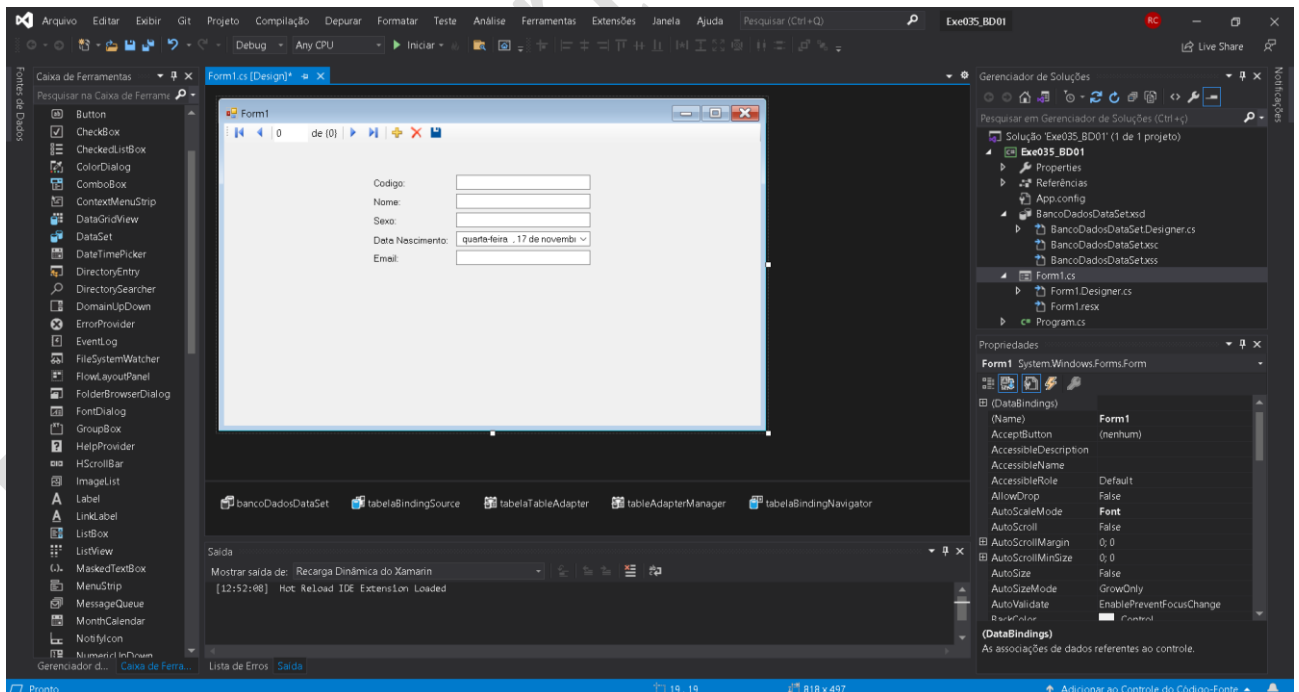


Prof. Roberto de Castro

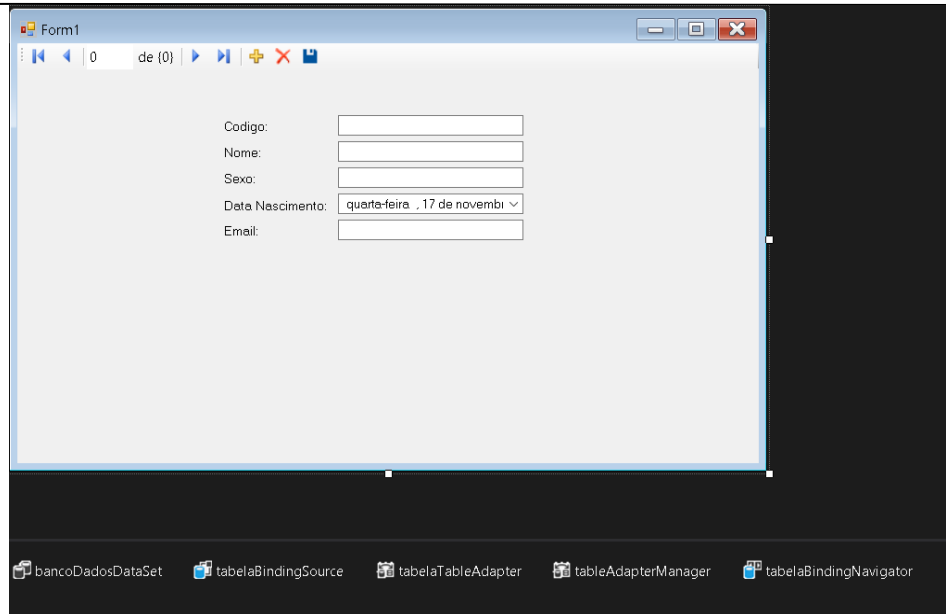
Vamos expandir as opções: BancoDadosDataSet e Tabela (basta clicar na seta ao lado destas opções).



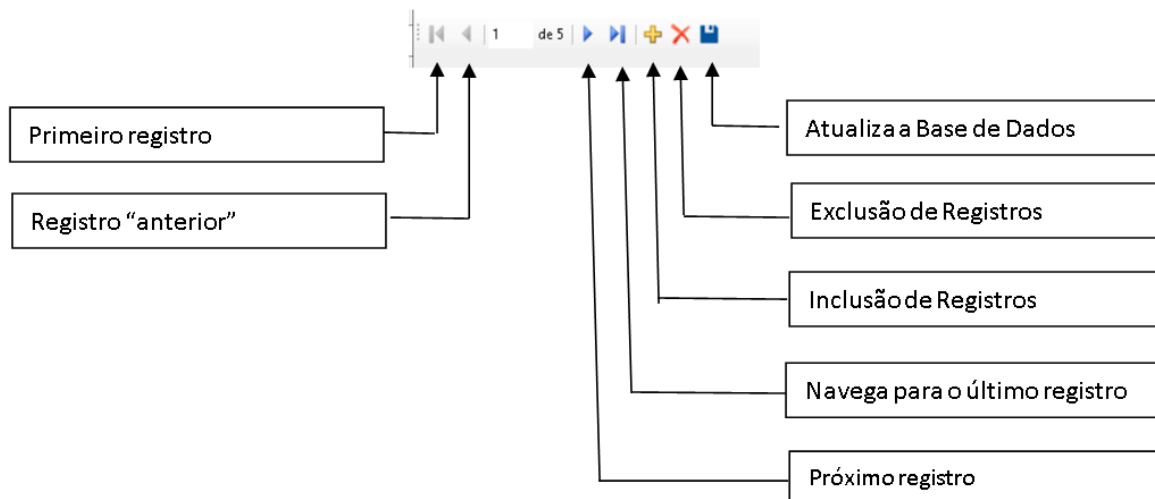
Clique na seta destacada em “amarelo” e selecione a opção “Detalhes”. Clique na caixa com a palavra “Tabela” e, com o “mouse pressionado”, arraste-a para o formulário. O assistente criará toda a interface gráfica e teremos uma aplicação completa que inclui, altera e exclui dados, sem nenhuma escrita de código!



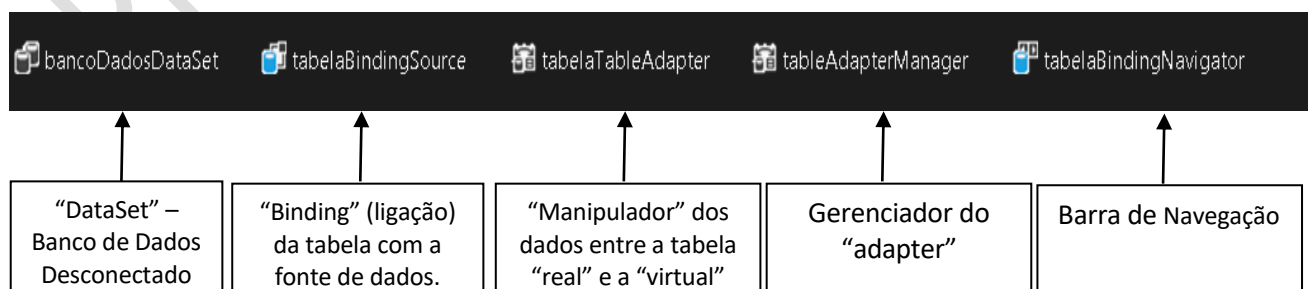
Prof. Roberto de Castro



Na parte “superior” temos a “barra de navegação” que é utilizada, em tempo de execução do projeto, para “navegar” entre os registros existentes, incluir novos dados, excluir e “atualizar” a base de dados.



Na parte “inferior” (bandeja) encontramos os objetos criados:



Prof. Roberto de Castro

Vamos a um detalhamento:

Binding Source → Sua finalidade é fornecer os serviços necessários para vincular os dados da tabela aos controles (caixas de texto...). Toda a interação com os dados (navegação, edição) é feita pelo BindingSource, bem como a ligação entre a fonte de dados e os controles do formulário.

TableAdapter → Responsável pela comunicação entre a aplicação e o Banco de Dados. Quando se cria a fonte de dados por meio da janela "Data Source", um ou mais "TableAdapters" são gerados automaticamente. Ele efetua a conexão com o Banco de Dados, executa um comando e preenche um "Data Table do DataSet". Também atualiza os dados no banco de dados quando a aplicação necessitar. Contém todos os comandos necessários para obter e atualizar os dados de uma tabela do banco de dados, bem como a "string de conexão".

DataSet → É uma instância (cópia desconectada) do banco de dados. Na maioria dos ambientes de SGBD há uma quantidade limitada de conexões com o BD (número de licenças/usuários), essas conexões são "dispendiosas". Ao utilizar o conjunto de dados (DataSet) desconectados, a aplicação libera o acesso para outros usuários/aplicações.

DataTable → Representa uma tabela do banco de dados.

TableAdapterManager → Este componente está presente no Visual Studio desde a versão 2008. Sua função é permitir/gerenciar a atualização dos dados em tabelas relacionadas (realizando a atualização hierárquica: pai / filho). Utiliza o relacionamento da chave primária entre as tabelas para determinar corretamente a ordem na qual deve enviar as atualizações (Inserts, UpDates e Deletes) do DataSet para o banco de dados, sem violar as restrições da chave-primária (integridade referencial) no banco de dados. Esta "integridade referencial" são as regras que controlam o comportamento para inclusão, atualização e exclusão de registros relacionados, prevenindo que os registros de uma tabela pai não sejam deletados enquanto existirem registros na tabela filha relacionada.

Faça um teste... Execute a aplicação!!

Vamos analisar algumas linhas que o assistente incluiu ...

Na carga do formulário (Form_Load) o assistente inseriu a instrução:

```
this.tabelaTableAdapter.Fill(this.bancoDadosDataSet.Tabela);
```

Podemos interpretar esta instrução da seguinte forma: no momento da carga do formulário, realiza-se a conexão com o Banco de Dados e a carga dos dados para o DataSet é realizada. Depois da carga dos dados, a conexão é encerrada. Toda a manipulação dos dados será realizada no DataSet.

OBS: A expressão "this" é opcional, que significa "este".

O assistente inseriu também outra programação, que é executada ao clicar na opção "atualizar", que está na barra de navegação (representado pelo ícone de um "disquete").

```
private void tabelaBindingNavigatorSaveItem_Click(object sender, EventArgs e)
{
    this.Validate();
    this.tabelaBindingSource.EndEdit();
    this.tableAdapterManager.UpdateAll(this.bancoDadosDataSet);
}
```

Podemos interpretar estes comandos como final de edição (manipulação dos dados) e atualização geral do banco de dados.

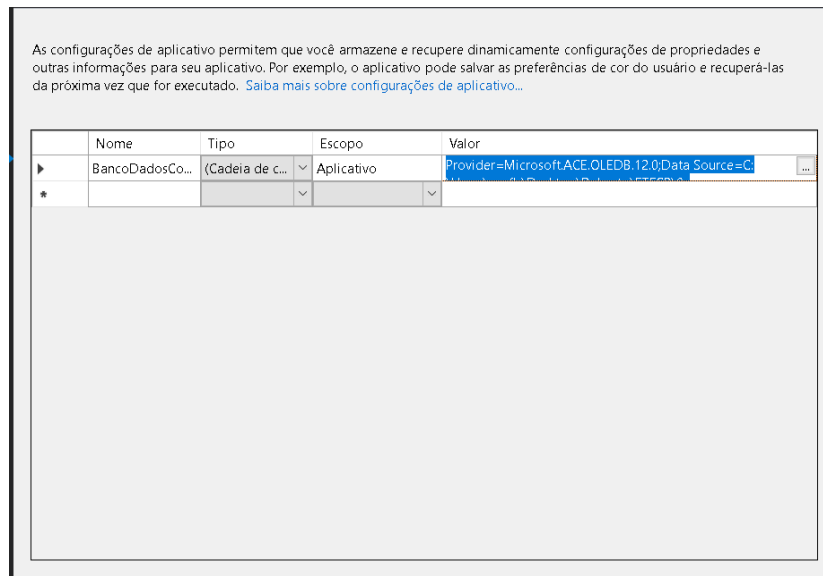
E vamos aproveitar que não fizemos muito esforço para desenvolver este aplicativo, para fazermos algumas "melhorias" e entendermos alguns conceitos.

1. Incluir um "botão" para finalizar o projeto.

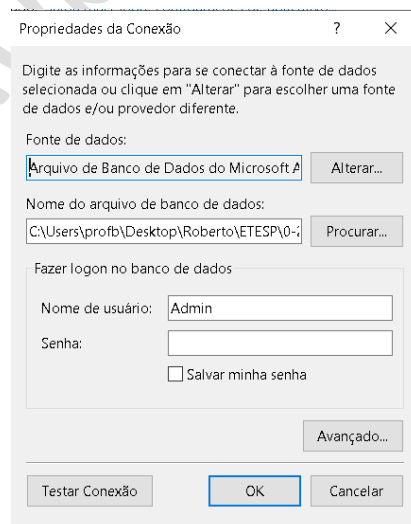
Prof. Roberto de Castro

2. Alterar a formatação da caixa de texto da “data de nascimento” (que na realidade não é uma caixa de texto, mas sim um “DateTimePicker”). Selecione o controle e altere a propriedade “format” para “short”.
3. Alterar as propriedades “MaxLength” das caixas de texto: Nome, Sexo e E-mail (para aceitarem somente o número de caracteres definidos na tabela: nome = 50, sexo = 1, email = 255).
4. Propriedade do formulário: StartPosition (Center Screen) e MaximizeBox (False).

ATENÇÃO: Caso você copie o aplicativo para outro local (para outro computador), haverá a necessidade de copiar também o banco de dados e neste caso, será necessário, no novo local, fazer a alteração do endereço do banco de dados. Para realizar esta operação clique em Projeto → Propriedades (Properties) → Configurações (Settings). Aí está a nossa “Connection String”:



Se alguma alteração for necessária, basta clicar nos “...” que aparecem ao lado da caixa “valor” e efetuar a alteração:

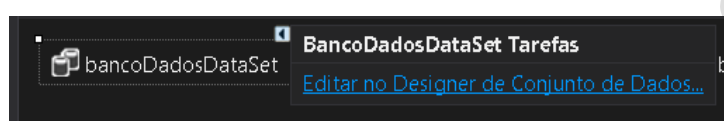


Prof. Roberto de Castro

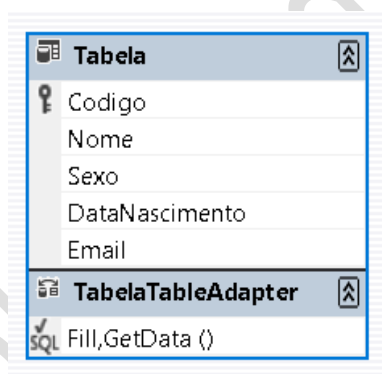
Vamos “testar” novamente o projeto. Execute e “navegue” entre os registros utilizando a “barra de navegação”. Faça três inclusões de dados (basta ir clicando no sinal de +, antes de cada inclusão). A cada inclusão, navegue pelos registros para verificar que eles foram incluídos. Altere alguns nomes. Exclua alguns registros. Não clique ainda no botão de salvar dados (“disquete”). Minimize o projeto e “abra” o banco de dados pelo Access... Você perceberá que todas as alterações, inclusões e exclusões que você realizou até o momento AINDA não estão “refletidas” no banco de dados real. O que está acontecendo?

Este aplicativo que desenvolvemos utiliza o conceito de “acesso desconectado”. O acesso desconectado “copia” os dados da base de dados original, para um “contêiner” (DataSet) que funciona como se fosse um banco de dados em cache (memória). Toda a atualização será realizada quando clicar na barra de navegação, na opção “Salvar Dados” (disquete).

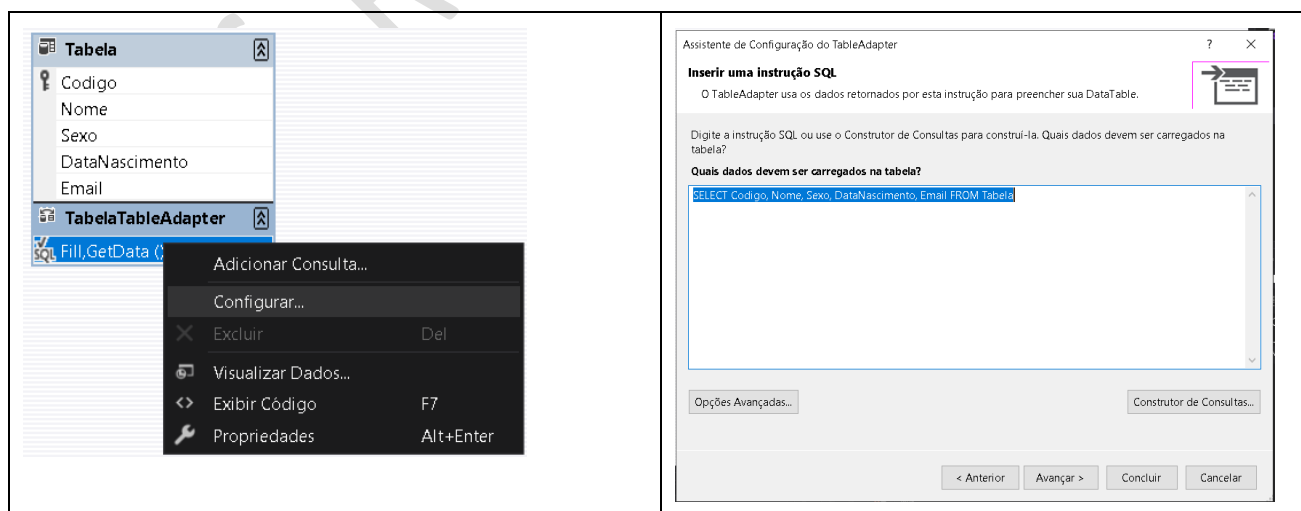
Agora vamos “conhecer” com um pouco mais de detalhes o “bancoDadosDataSet”. Dê um clique para selecioná-lo. Clique no link que será exibido em azul “Editar no Designer de Conjunto de Dados”.



Será exibido um “diagrama” ilustrando a estrutura de nosso “DataSet”. O diagrama está dividido em duas partes. Na parte superior serão exibidos os campos da tabela e na parte inferior os comandos “SQL” de manipulação deste “DataSet”. Observe a imagem:



Pressione, com o “botão direito” do mouse, sobre a linha “Fill, GetData()”. Selecione a opção “Configurar”.



Prof. Roberto de Castro

ATENÇÃO: Apenas visualize, sem alterar nenhuma programação. Temos uma instrução “SQL” que seleciona todos os campos e registros de nossa tabela. É justamente esta instrução que será executada na “carga do formulário” (Form_Load). Este comando “Select...” foi criado e salvo com o nome de “Fill” (preencher), mas poderia ter sido dado qualquer outro nome...

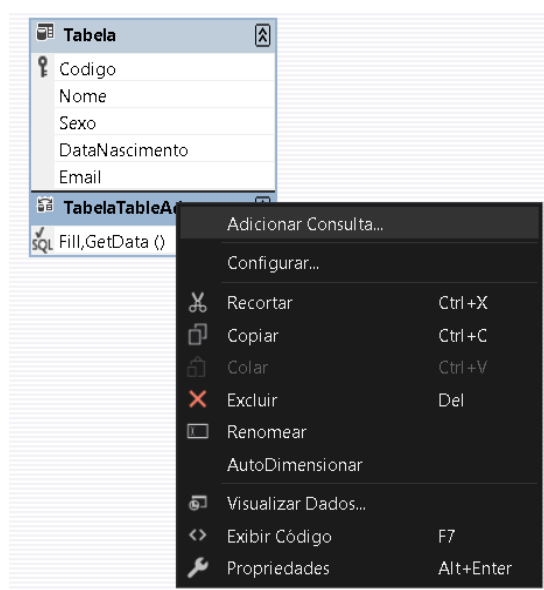
Veja novamente o comando que está no “Form_Load”:

```
tabelaTableAdapter.Fill(this.bancoDadosDataSet.Tabela);
```

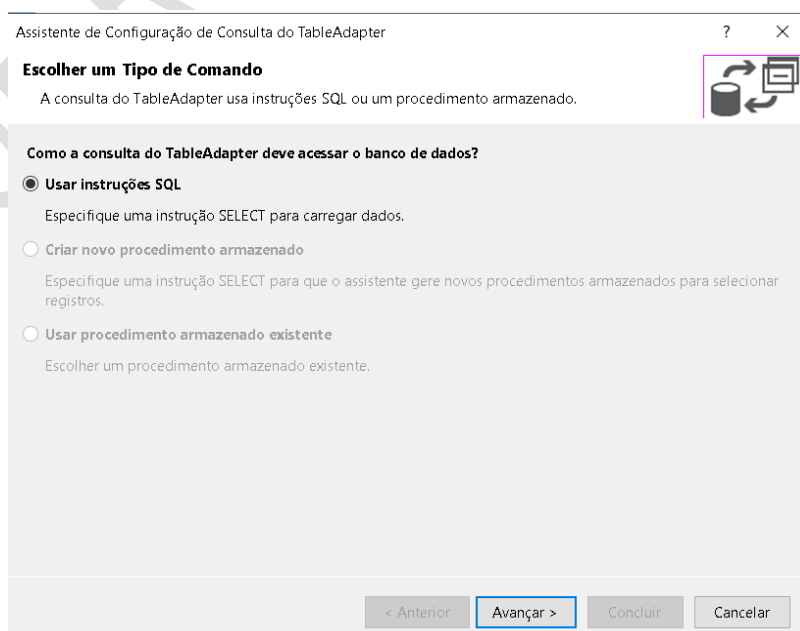
O destaque em vermelho “Fill” é o nome da instrução SQL que está gravada no tabelaTableAdapter.

Vamos criar dois novos comandos SQL que irão “contar” o número de pessoas do Sexo masculino e o número de pessoas do sexo feminino, gravados na tabela.

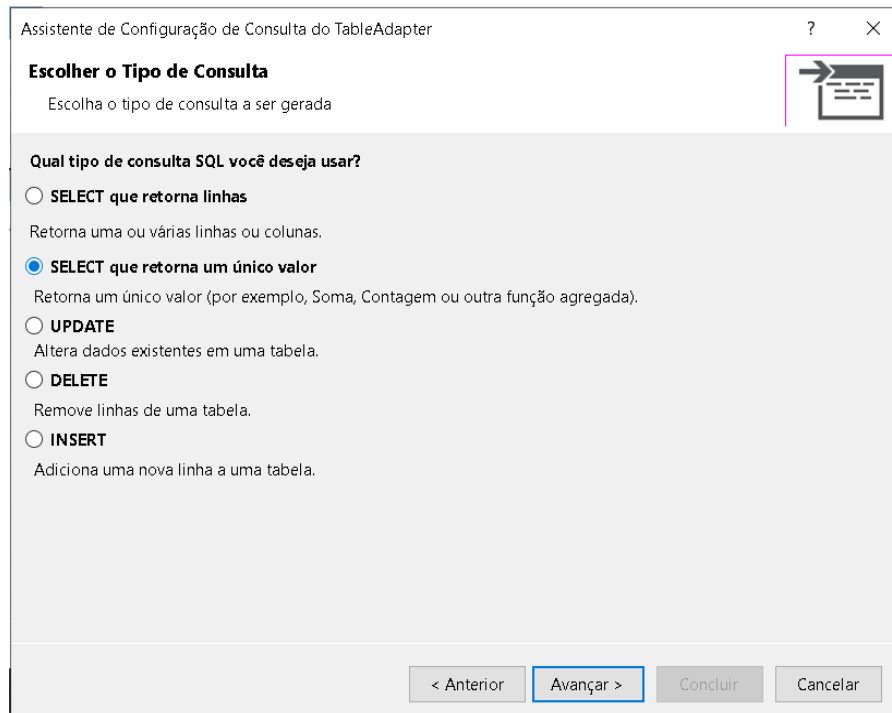
Pressione o botão direito do mouse sobre a palavra “TabelaTableAdapter” e selecione a opção “Adicionar Consulta”:



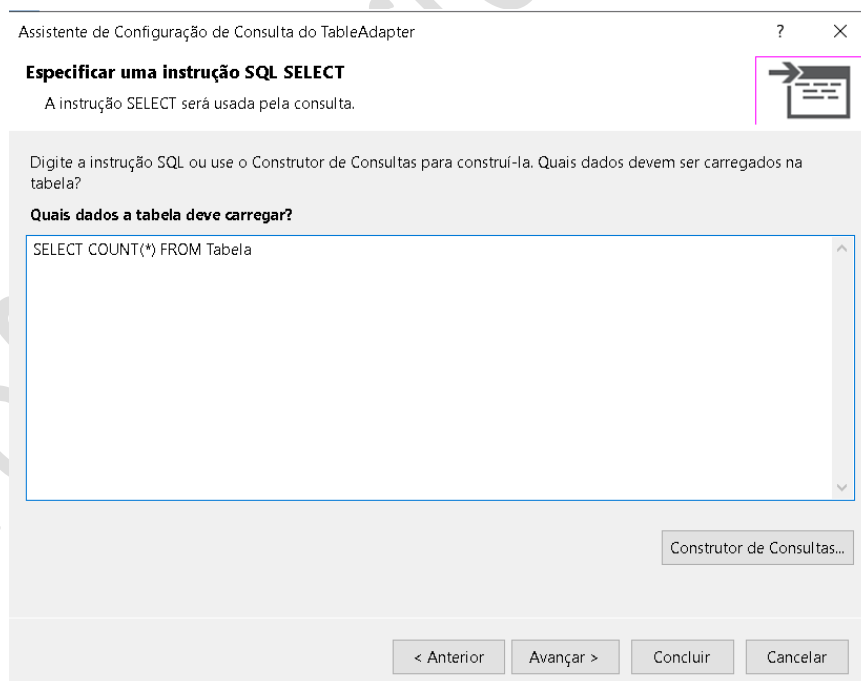
Na primeira tela exibida, apenas clique em “Avançar >”



Na próxima etapa, selecione a opção “Select que retorna um único valor” e clique em “Avançar >”



Observe, na imagem abaixo, que o assistente já traz uma sintaxe de SQL para “contar” TODOS os registros de nossa tabela.

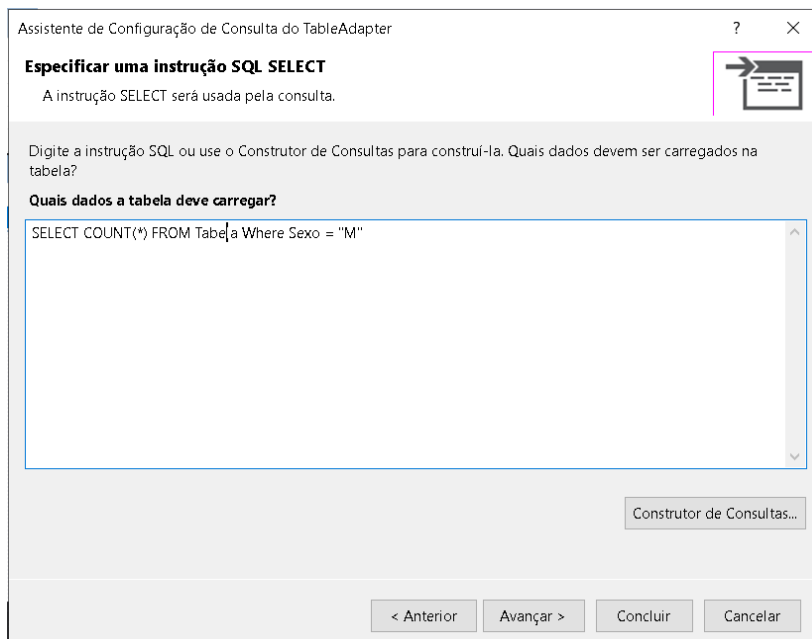


No nosso caso específico, precisaremos incluir uma condição: contar somente as pessoas do sexo = masculino.

O assistente disponibiliza a opção “Construtor de Consultas” (um excelente aliado para o aprendizado de SQL), mas para o momento não será necessário. Apenas complete o comando SQL para ficar conforme o modelo lado: `SELECT COUNT(*) FROM Tabela Where Sexo = “M”`.

Prof. Roberto de Castro

IMPORTANTE: Existe aqui a possibilidades de ocorrer erro, principalmente no caso de ocorrer divergência na grafia do nome do campo Sexo e no nome da Tabela. Digite com muita atenção!!!



Assistente de Configuração de Consulta do TableAdapter

Especificar uma instrução SQL SELECT

A instrução SELECT será usada pela consulta.

Digite a instrução SQL ou use o Construtor de Consultas para construí-la. Quais dados devem ser carregados na tabela?

Quais dados a tabela deve carregar?

SELECT COUNT(*) FROM Tabela Where Sexo = "M"

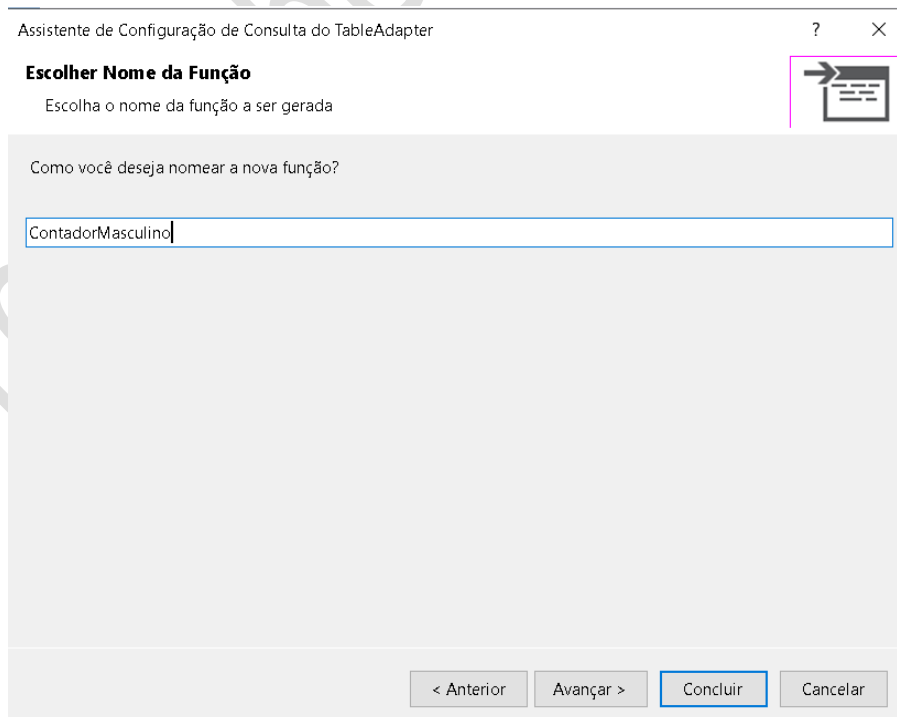
Construtor de Consultas...

< Anterior Avançar > Concluir Cancelar

Dica sobre o "Select Count": O "Select Count(*)" é utilizado para contar todos os registros em uma tabela, não importando se um campo é nulo ou não. "Select Count(nomeCampo)" é utilizado para contar registros com o campo especificado não nulo.

Clique em avançar.

Na próxima tela o assistente solicita a digitação do nome para o comando digitado. Vamos digitar "ContadorMasculino" e clicar em "Avançar >"



Assistente de Configuração de Consulta do TableAdapter

Escolher Nome da Função

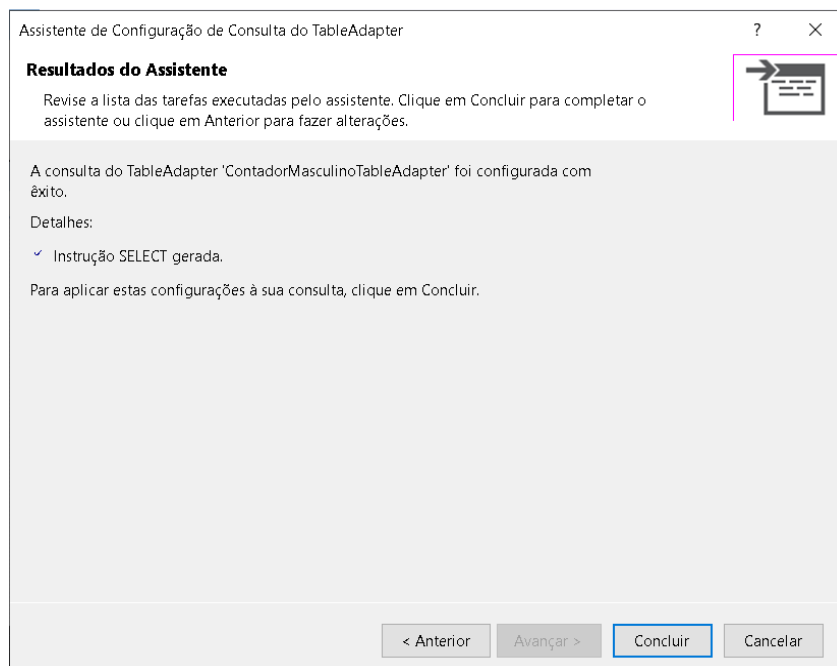
Escolha o nome da função a ser gerada

Como você deseja nomear a nova função?

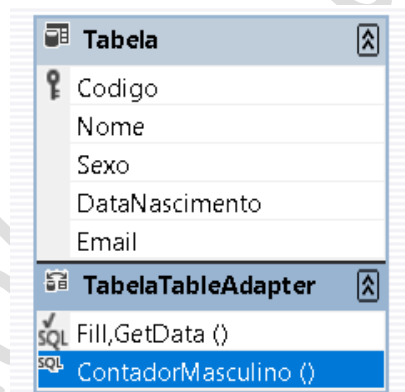
ContadorMasculino

< Anterior Avançar > Concluir Cancelar

O assistente exibe a mensagem de que a "consulta" foi criada...



Clique em concluir. Observe que o “TabelaTableAdapter” agora possui um novo “comando”: ContadorMasculino.



Agora vamos utilizar esta “consulta” que acabamos de criar...

No formulário de nosso projeto, inclua o botão “Exibir Total”:

Acesso a Banco de Dados

Codigo:

Nome:

Sexo:

Data Nascimento: 19/11/2021

Email:

Exibir Total SAIR

Inclua a programação do botão “Exibir Total”:

```
private void BtnTotal_Click(object sender, EventArgs e)
{
    MessageBox.Show("total de Masculino=" + tabelaTableAdapter.ContadorMasculino().ToString());
    // ou, para o caso de desejar a informação em um campo para fazer calculos...
    //int totalM = (int)tabelaTableAdapter.ContadorMasculino();
    //MessageBox.Show("total de Masculino=" + totalM.ToString());
}
```

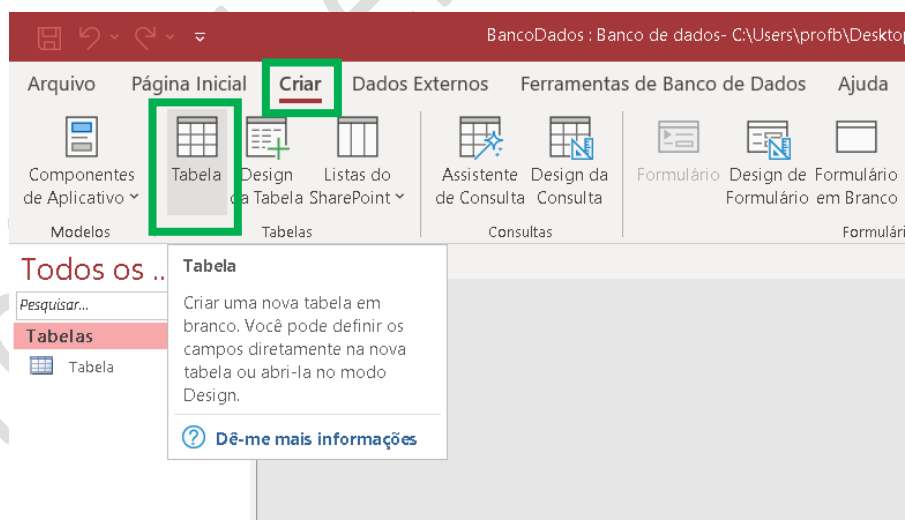
Para treinar: Inclua uma nova consulta SQL no Adapter, para contar os registros com o sexo="F".

Boa Sorte!!

Exercício proposto Exe036 BD02: Com base nos conceitos apresentados e na estrutura da tabela “Funcionarios”, desenvolva o projeto para manipulação de dados e a implementação de uma “consulta” para exibir a somatória dos salários.

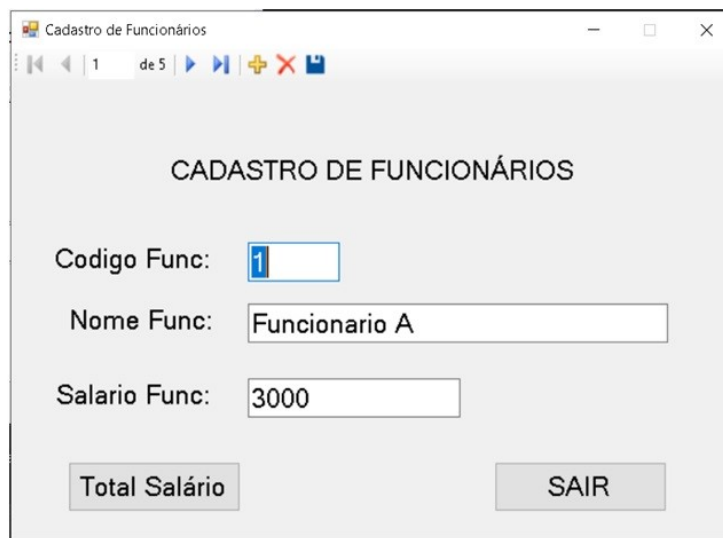
Campo	Tipo	Observação
CodigoFunc	Número	Campo Chave
NomeFunc	Texto Curto	Tamanho 50; Requerido
SalarioFunc	Moeda	Requerido / Regra de validação: > 1000 e < 20000

OBS: Você poderá criar esta tabela dentro do banco de dados já existente, ok?? Abra o Banco de Dados, clique no menu “Criar” e selecione “Tabela”. O nome da tabela será “Funcionarios”.



Após criar a tabela, cadastre 5 (cinco) funcionários...

No “Visual Studio”, modelo do formulário:

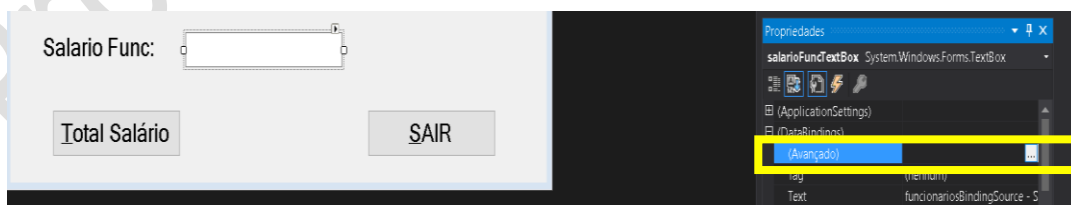


Vamos configurar o “TextBox” do “Salario Func” para exibir “valores monetários”:

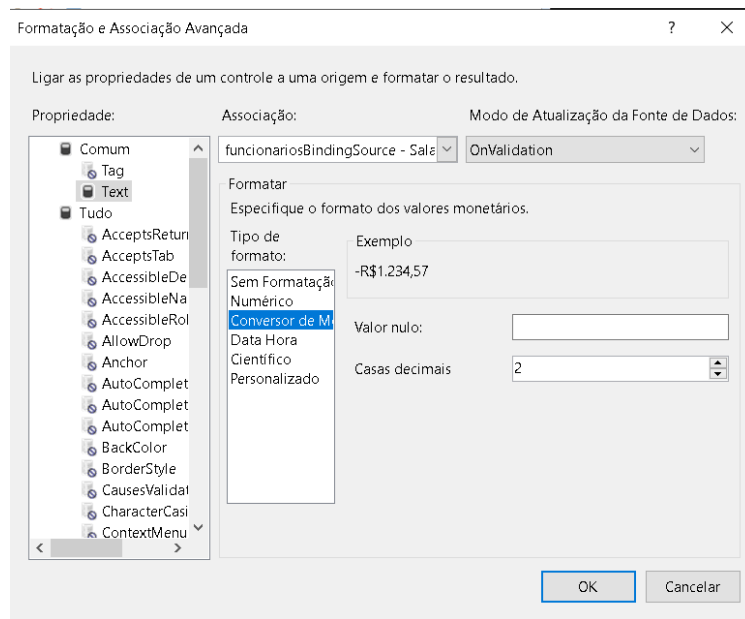


Etapas:

- 1 Selecione o TextBox “SalarioFunc”;
- 2 Na janela “Propriedades (properties)”, clique na propriedade “DataBinding → Advanced” (clique nos “...”)

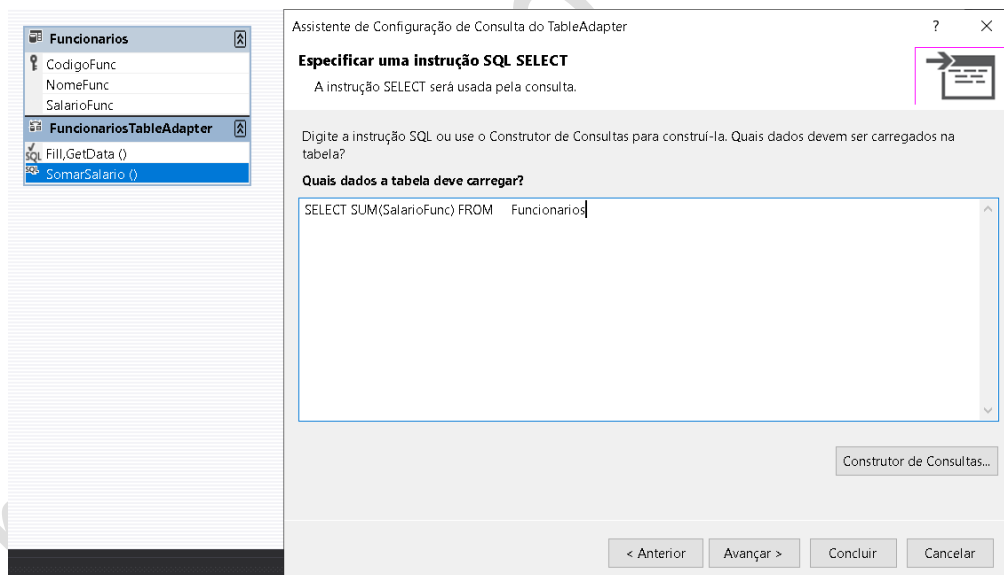


Selecione o formato “Conversor de Moeda” e clique em OK:



Dicas:

1 Instrução SQL para “Somar salário”: SELECT SUM(SalarioFunc) from Funcionarios



2 Programação do formulário, botão “Total Salário”:

```
private void BtnTotal_Click(object sender, EventArgs e)
{
    decimal totalSalario = (decimal)funcionariosTableAdapter.SomarSalario();
    MessageBox.Show("Total de Salário: " + totalSalario.ToString("C2"));
}
```