Laboratório 19 –Acesso a dados

Neste laboratório é apresentada a criação de um repositório de dados utilizando as abordagens banco de dados primeiro (database first) e operações básicas de CRUD

1. Dentro do Visual Studio selecione o menu: ***File | New | Project***. Na lista ***Installed Templates*** (modelos) **Visual C# |WEB**, selecione ***ASP.NET Web Application | Web Forms***

2. Adicione ao projeto o arquivo de dados **MusicStore**. Selecione a pasta **App_Data**(se esta pasta não existir acrescente-a ao projeto) e com o botão direito selecione a opção **Add | Existing Item**, selecione o arquivo **MusicStore.mdf**disponibilizado.

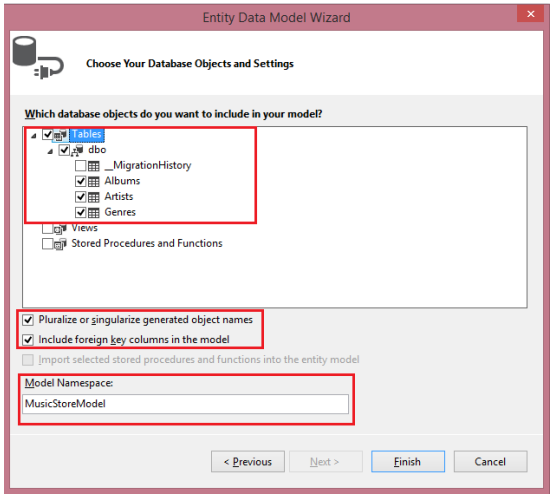
3. Selecione a pasta **App_Data** e dê um duplo clique sobre o arquivo **MusicStore.mdf**, para conectar ao arquivo. Navege pelo **ServerExplorer**, avalie a estrutura do banco de dados e seu conteúdo. Antes de continuar feche a conexão com o banco de dados.

4. Crie um *Data Model* para acesso ao banco de dados. Selecione a pasta **Models** e com o botão direito selecione a opção **Add | New Item**. Na caixa de diálogo **Add New Item**, selecione o template **Data| ADO.NET Entity Data Model**. Altere o nome do modelo para **StoreDB**e clique **Add**.

5. No **Wizard**apresentado selecione a opção **EF Designer from database** e tecle **Next**. Deverá ser sugerido como fonte de dados o banco de dados adicionado ao projeto no passo anterior (MVCMusicStore.mdf). Caso este arquivo não seja apresentado, elecione a opção para criar uma nova conexão (**New Connection | Microsoft SQL Server Database File)**e selecione (**Browse)** o arquivo **MvcMusicStore.mdf**na pasta **App_Data**.

6. Verifique que o nome do modelo seja **MusicStoreEntities**,marque a opção **Save connection settings...** e tecle **Next.**

7. Selecione as tabelas**Albuns, Artists e Genres,** selecione as opções **Pluralize or singularize...** e **Include foreign key...**, conforme figura a seguir, e tecle **Finish**:



8. Veja na pasta **Models | StoreDB.edmx | StoreDB.tt** que foram criadas as classes **Album**, **Genre** e **Artist**, verifique o conteúdo destas classes, bem como a classe de contexto criada (**StoreDB.Context.cs**).

9. Adicione ao projeto um novo formuário ao projeto (**Generos.aspx**) e acrescente um componente do tipo GridView.

10. No evento **Page_Load** da página, instancie o repositório e acrescente inicialize o GridView (note que será necessário adiconar a referência ao namespace):

```
protectedvoidPage_Load(object sender, EventArgs e)
 {
MusicStoreEntities _db = newMusicStoreEntities();

    GridView1.DataSource = _db.Genres.ToList();
GridView1.DataBind();
 }
```

11. Execute e teste.

12. Para ter uma lista mais utilizável, acrescente à página um componente do tipo **ListView** e altere suas propriedades para que fique como no código abaixo (acerte o namespace na propriedade **ItemType** para o nome do seu projeto):

```
<asp:ListViewID="categoryList"
ItemType="Lab19_EF_DBFirst_CRUD.Models.Genre"
runat="server"
SelectMethod="GetCategories">
<ItemTemplate>
<bstyle="font-size: large; font-style: normal">
<ahref="Albuns.aspx?genero=<%#:Item.Name%>">
<%#:Item.Name%>
</a>
</b>
</ItemTemplate>
<ItemSeparatorTemplate>  |</ItemSeparatorTemplate>
      </asp:ListView>
```

13. Após testar, altere o formato de apresentação para uma lista de gêneros:

```
   <ul>
<asp:ListViewID="categoryList"
ItemType="Lab19_EF_DBFirst_CRUD.Models.Genre"
runat="server"
SelectMethod="GetCategories">
<ItemTemplate>
<bstyle="font-size: large; font-style: normal">
<li>
<ahref="Albuns.aspx?genero=<%#:Item.Name%>">
<%#:Item.Name%>
</a>
</li>
</b>
</ItemTemplate>
</asp:ListView>
   </ul>
```

14. Acrescente ao projeto um formulário (**Albuns.aspx**), e repita o processo anterior apresentando as informações utilizando um componente do tipo *GridView*, não esqueça de filtrar pela informação recebida pela *querystring*.

15. Execute e teste. A apresentação não deve ter ficado grande coisa, mas deve ter funcionado.

16. Para apresentar os albuns utilizando um *ListView*, crie na pasta Models uma nova classe (**AlbumViewModel.cs**), com o conteúdo abaixo:

```
publicclassAlbumViewModel
    {
publicintalbumId { get; set; }
publicstring Nome { get; set; }
publicstringGenero { get; set; }
publicstringArtista { get; set; }
publicdecimalPreco { get; set; }
     publicstringUrl { get; set; }

    }
```

17. Acrescente à página *Albuns.aspx*um *ListView* configurado como abaixo:

```
<asp:ListViewID="albumList"runat="server"
DataKeyNames="albumId"GroupItemCount="1"
ItemType="Lab19_EF_DBFirst_CRUD.Models.AlbumViewModel"SelectMeth
od="getAlbuns">
<EmptyDataTemplate>
<table>
<tr>
<td>Nehum álbum encontrado.</td>
</tr>
</table>
</EmptyDataTemplate>
<EmptyItemTemplate>
<td/>
</EmptyItemTemplate>

<ItemTemplate>
<tdrunat="server">
<table>
<tdstyle="width:100px">
<ahref="Remover.aspx?id=<%#:Item.albumId%>">remover</a>
</td>
<tdstyle="width:200px">
<%#:Item.Titulo%>
</td>
<tdstyle="width:100px">
<%#:String.Format("{0:c}", Item.Preco)%>
</td>
<tdstyle="width:100px">
<%#:Item.Genero%>
</td>
<tdstyle="width:200px">
<%#:Item.Artista%>
</td>
```

```
</table>
</p>
</td>
</ItemTemplate>

</asp:ListView>
```

18. Adicione um novo formulário ao projeto de nome **Remover.aspx.** Acrescente um controle do tipo label (com id **lblTitulo**) e um botão (com id **btnConfirma**) no formulário. No evento Page_Load da página use o código:

```
protectedvoidPage_Load(object sender, EventArgs e)
        {
int id;
stringparam =  Request.QueryString["id"];

if (!Int32.TryParse(param, out  id))
{
                lblTitulo.Text = "Solicitação inválida";
                btnConfirma.Visible = false;
}
else
            {
MusicStoreEntities _db = newMusicStoreEntities();
string query = (from a in _db.Albums
wherea.AlbumId == id
selecta.Title).FirstOrDefault();

if (query != null)
                {
ViewState["id"] = id;
lblTitulo.Text = "Confirma remoção do álbum \"" + query + "\"?";
}
else
                {
                    lblTitulo.Text = "Solicitação inválida";
btnConfirma.Visible = false;
                }
            }

        }
```

19. E no evento click do botão o código para remoção:

```
protectedvoidbtnConfirma_Click(object sender, EventArgs e)
            {
int? id = (int)ViewState["id"];
if (id != null) {
MusicStoreEntities _db = newMusicStoreEntities();

var query =  from album in _db.Albums
wherealbum.AlbumId == id
select album;

if (query.Count() > 0)
```

```
                {
                    _db.Albums.Remove(query.First());
                    _db.SaveChanges();
Response.Redirect("Default.aspx");
                }

            }
```

20. Compile e teste.

21. Acrescente um novo formulário para mostrar os detalhes de um álbum (**Detalhes.aspx**). Para mostrar os detalhes de um álbum arraste componenetes do tipo *Label* e apresente os valores conforme exemplos anteriores.

22. Execute e teste.

23. Para a página de detalhes, uma alternativa é a utilização do componente *FormView*. Arraste-o para o formulário e configure-o da seguinte forma:

```
<asp:FormViewID="albumDetail"runat="server"
ItemType="Lab19_EF_DBFirst_CRUD.Models.AlbumViewModel"
SelectMethod="GetAlbum"
RenderOuterTable="false">
<ItemTemplate>
<div>
<h1><%#:Item.Titulo%></h1>
</div>
<br/>
<table>
<tr>
<td>
<imgsrc="<%#:Item.Url%>"style="border:solid;
height:300px"alt="<%#:Item.Titulo%>"/>
</td>
<td> </td>
<tdstyle="vertical-align: top; text-align:left;">
<b>Genero:</b><%#:Item.Genero %>
<br/>
<span><b>Preço:</b> <%#:String.Format("{0:c}", Item.Preco) %></span>
<br/>
<span><b>Artista:</b> <%#:Item.Artista%></span>
<br/>
</td>
</tr>
</table>
</ItemTemplate>

</asp:FormView>
```

24. Antes de testar, crie uma nova pasta **Images**, dentro da pasta **Content**, e adicione à esta o arquivo **placeholder.png** disponibilizado. Execute e teste.

25. Adicione um novo formulário para adiconar novos albuns (**Criar.aspx**). Utilize componentes do tipo DropDownList para a entrada do gênero e do nome do artista.

26. Execute e teste.

27. Altere o formulário que lista os álbuns para contér um botão que permita adiconar novos álbuns e também um link que permita ver os detalhes do álbum. Execute e teste.

28. Crie um novo formulário que permita editar um álbum existente.