CENTRO PAULA SOUZA

GOVERNO DE SÃO PAULO

ETEC PROF. PEDRO LEME BRISOLLA SOBRINHO

DESENVOLVIMENTO DE SOFTWARE II

Prof.: Luciana Sanches Carlomagno

AULA 1- CONEXÃO E INSERÇÃO DE DADOS MYSQL

Quem está habituado a programar C# normalmente utiliza SQL Server como banco de dados, uma vez que ela está integrada no Visual Studio e, por isso mesmo, existe grande facilidade em trabalhar com as duas ferramentas.

Por vezes existem projetos em que se torna conveniente (por várias razões) utilizar outro tipo de base de dados. Seja que banco de dados for, a sua integração é sempre diferente do SQL Server.

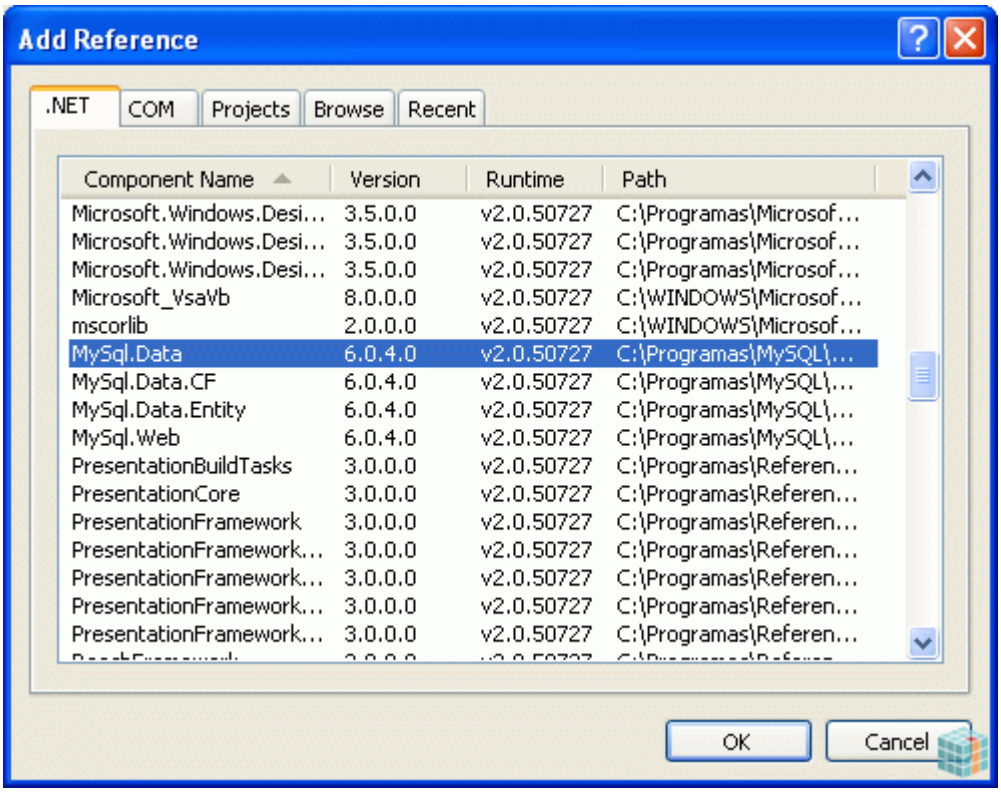
Neste caso vou mostrar como fazer a integração entre o C# e o MySQL.

Supondo que já existe uma instalação do MySQL na máquina, precisamos instalar um intermediário entre o C# e a base de dados. Neste caso necessitamos instalar o **MySQL Connector NET.**

Devemos já ter uma base de dados com uma tabela, em que os campos da tabela são: id, nome, email.

Como exemplo criamos um projeto Windows Forms Application.

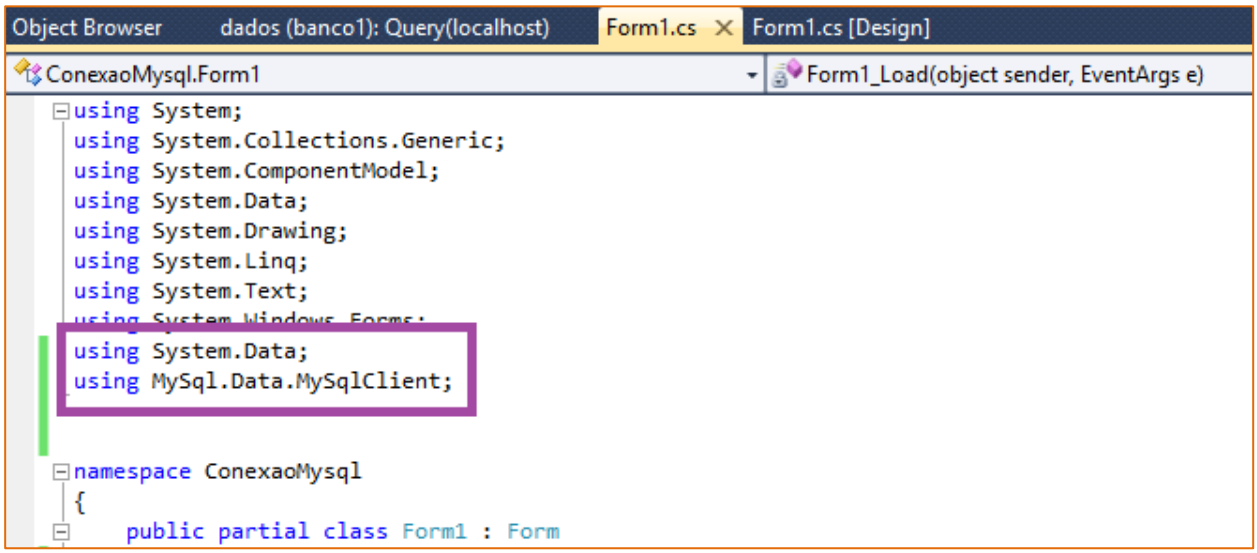
Antes de mais nada, devemos fazer uma referência à classe que vai ligar o C# ao MySQL. Para isso vamos ao painel Solution Explorer, na raiz do projeto, clicamos com o lado direito do mouse e selecionamos **Add Reference.**



Na primeira divisória (.NET) selecionamos a referência MySQL.Data e damos OK.

No formulário efetue a inserção das bibliotecas:

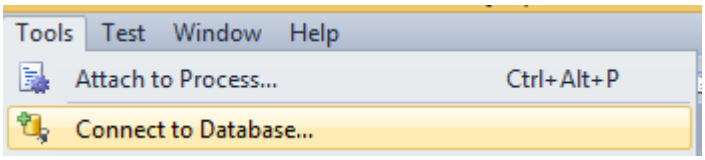
```
using System.Data;
using MySql.Data.MySqlClient;
```



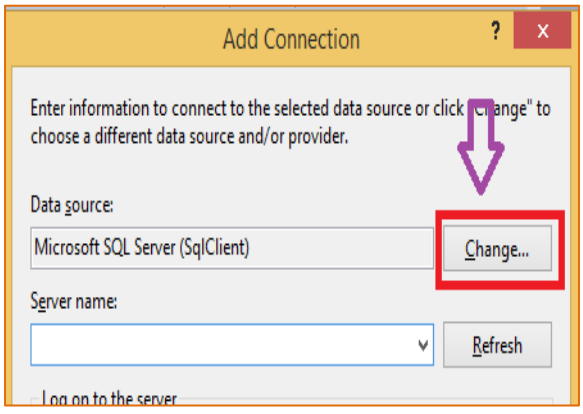
Sem adicionar a referência à **MySQL.Data,** a classe **MySql.Data.MySqlClient** não será reconhecida. Para exemplificar fazemos um formulário de inserção de dados (nome e e-mail) na base de dados.

EFETUAR A CONEXÃO

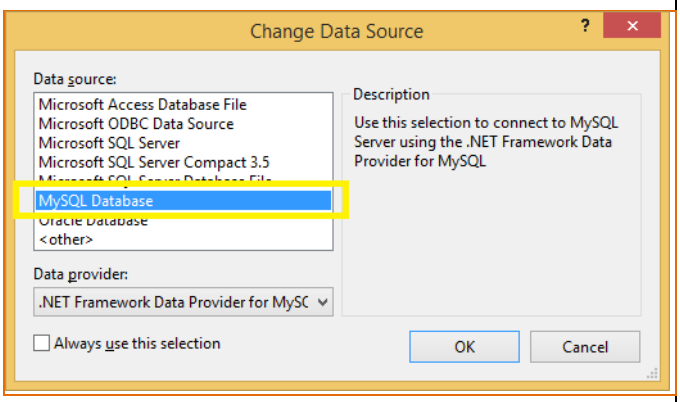
No janela **Server Explorer** (menu View)

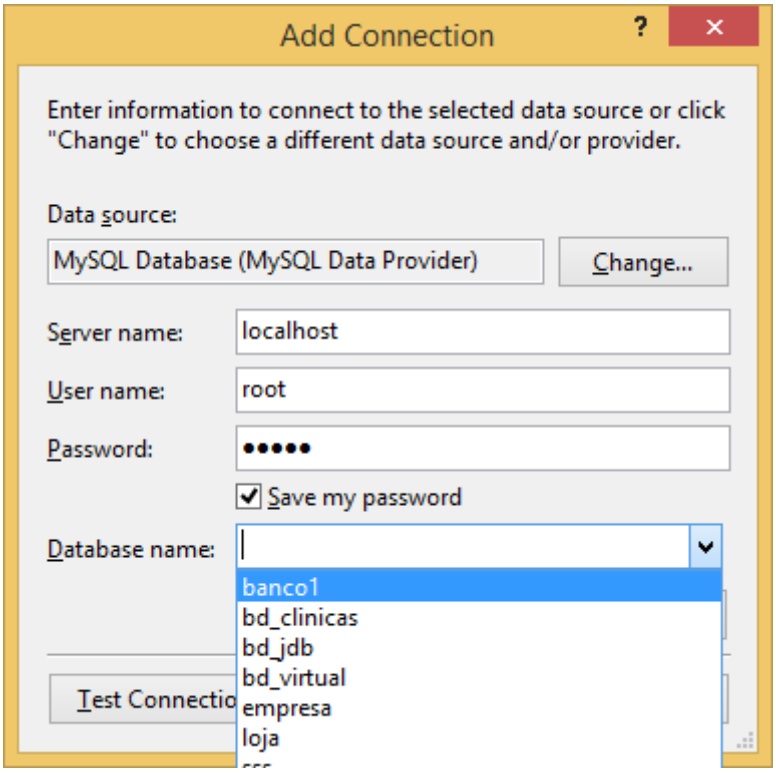


Será apresentado a janela de conexão, clique em Change para mudar o tipo de conexão:



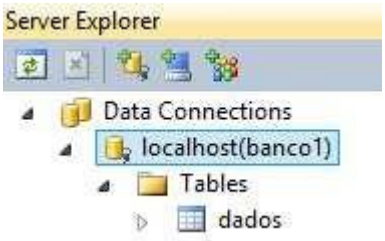
Caso o conector esteja instalado será exibido a opção **MYSQL DATABASE.**





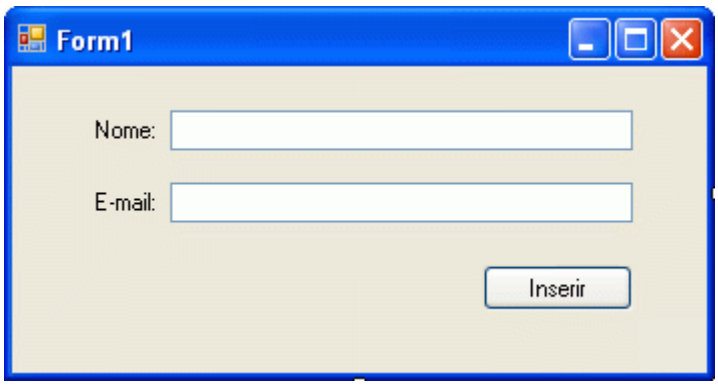
Nesta tela configure as opções do **servidor, usuário e senha**. Em seguida selecione a base de dados e clique em **OK.**

Dentro do seu projeto em Server Explorer será apresentado:



DESENVOLVIMENTO A INTERFACE

Construa o formulário de acordo com a estrutura da tabela e um botão que terá a ação de inserir os dados na base de dados.



Vamos então definir, em primeiro lugar, o dataset e a string de conexão à base de dados.

No evento click do botão inserir ação:

INSERÇÃO DE DADOS

Instruçoes Try ...Catch

Usando o Try, Catch

Basicamente o comando try recebe comandos que causam erro e o comando catch recebe um possível tratamento para o erro gerado.

Try - Tentar

Catch - Captura

Em outras palavras:

O programa vai "TENTAR" fazer determinadas linhas de comando (ou apenas uma). Se ele não conseguir, ele vai "CAPTURAR" outro bloco de comando e deixar aquele que ele "TENTOU" fazer pra trás. Assim o seu programa não dá erro quando, por exemplo, você deixa uma TextBox (lugar pra digitar algo) pra permitir apenas números, e acidentalmente você aperta uma letra; assim, em vez do sistema dar erro, ele vai mandar uma mensagem dizendo que só é permitido digitar números. Vamos a prática:

```
try
{
    Comando que ele "tentará" fazer 1;
    Comando que ele "tentará" fazer 2;
    Comando que ele "tentará" fazer "";
}
```

```
catch
{
     Comando que ele "capturará" se ele "tentar" e falhar 1;
     Comando que ele "capturará" se ele "tentar" e falhar 2;
     Comando que ele "capturará" se ele "tentar" e falhar "";
    return; O programa irá executar o bloco "Catch" e fará o programa parar totalmente pra uma possível correção nos dados, para fazer com que o bloco "Try" não dê erro.
}
```

Vale lembrar que:

Qualquer comando que der erro no bloco "Try (tentar)" será o suficiente para que o bloco todo seja cancelado e o programa siga no bloco "catch(capturar)". Não importa a ordem do bloco "Try", onde der erro TUDO será cancelado e o programa continuará no bloco "Catch".- Os metodos **Habilita e Desabilita** foram criados para limpar e habilitar/desabilitar os campos da caixa de texto atraves da propriedade Enabled;

- Quando o formulário é carregado (evento Load) os campos são desabilitados.

- Quando o usuário clicar no **botão Novo**, automaticamente os campos estarão sem conteudo e habilitados para receber um novo cadastro.

- No metodo verifica_campos verifica se existem alguma caixa de text sem valor, pois na definição da estrutura da nossa tabela, não é permitido que nenhum campo fique sem valor (nulos nao permitidos).

- Dentro do **botão Gravar** coloquei um try/catch para capturar algum erro que possa acontecer. Comecei instanciando o MySqlConnection, passando a ele a string de conexão, depois instanciei o MySqlCommand, atribuindo a ele o MySqlConnection. Depois, usei o método CommandText, do MySqlCommand, para fazer a instrução SQL, só que ao invés de passar diretamente os valores dentro dele, eu passei apenas os parâmetros, como uma forma de segurança dos dados. Assim minha instrução SQL fica parametrizada, pelo uso do arroba (@) + o nome da coluna da tabela.

Logo após, passei os valores por meio do método AddWithValue, pertencente ao método Parameters, do MySqlCommand. Nele, que espera dois parâmetros, que são os parâmetros declarados no INSERT e os valores em si, foram passados os valores das colunas.

Após isso, abro minha conexão por meio do método Open, uso o ExecuteNonQuery, que é perfeito para inserção no banco, já que a mesma não nos retorna dados(a usamos também para fazer Update e Delete) e fecho a conexão.

Apenas para fins didáticos uma breve explicação dos 4 tipos de execuções que tenho no SqlCommand:

Repare que pela imagem a string de conexão (SqlConnection) possue um sinal de +, isto foi feito para

```
1 referencia
private void btnCadastrar_Click(object sender, EventArgs e)
{
    try
    {
        MySqlConnection conn = new MySqlConnection("server=localhost;database=db_hotel;uid=root;pwd=''");
        MySqlCommand comando = new MySqlCommand();
        comando.Connection = conn;
        comando.CommandText = ("insert into usuario(login,senha) values (@login,@senha)");
        comando.Parameters.AddWithValue("@login", txtLogin.Text);
        comando.Parameters.AddWithValue("@senha", txtSenha.Text);
        conn.Open();
        comando.ExecuteNonQuery();
        MessageBox.Show("usuário inserido com sucesso");
    }
    catch (MySqlException ex)
    {

        MessageBox.Show("Erro " + ex.Message);


    }
}
```

```
MySqlConnection conn = new MySqlConnection("server=localhost;database=db_hotel;uid=root;pwd=''");
```

A linha acima, que efetua a conexão é o primeiro passo para efetuar as operações com os dados, por isso precisa ser realizada corretamente, para não ocorrer erros podemos selecionar a mesma na caixa de propriedades, na opção Connection String.