Otázka 12 - Komunikace s databázovým systémem

### **Úvod**

*Otázka je zpracována a popisována na programovacím jazyce C#. Tento jazyk v základu umí pracovat pouze s MSSQL.*

*Z tohoto, důvodu když tvoříme klienta pro jiné databázové systémy (např. MySQL a Oracle), musíme použít knihovny, které je potřeba stahovat přímo ve Visual Studiu. Oboje však funguje na stejném principu a dá se říci, že když umíte udělat klient na jednu databázi, s trochou googlení, umíte udělat klient na všechny databáze. Tomu, kdo si tuto otázku vybere u maturity bych doporučoval mluvit o MySQL, protože právě zde se dá ztratit spoustu času mluvením o stahování potřebné knihovny.*

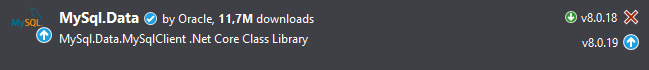
*-Matěj Najdrle*

**

### **Připojení pro MySQL**

Jak jsem již výše uvedl, je potřeba stáhnout si knihovnu která nám bude podporovat připojení k naší MySQL databázi. To lze udělat následovně:

1. Pravý tlačítkem myši klikneme na projekt ve Visual studiu (Vpravo v solution exploreru) a zvolíme Manage NuGet Packages.
2. Zobrazí se v podstatě “Obchod” s knihovnami a do search baru v sekci Browse napíšeme MySQL a měla by se zobrazit knihovna MySQL.Data (Pokud ne, tak to nějak dopište celé.)
3. Jednou kliknete a dáte install.



Když máme staženou knihovnu, nic nám nebrání v tom připojit se skrze C# k MySQL databázi, je ovšem třeba udělat referenci na právě staženou knihovnu vepsáním tohoto kódu nad namespace projektu:



Pokud pracujeme ve Windows Forms, můžeme například po stisknutí tlačítka poslat request k připojení k databázi. Samozřejmě vše závisí na tom, o jak rozvinutou databázi se jedná a zda se přihlašujeme pod specifickým uživatelem. Pro naše účely si ukážeme jak se po stisknutí tlačítka přihlásit pod rootem.Nejprve ze všeho si deklarujeme proměnnou typu MySqlConnection a pojmenujeme si jí napřiklad conn.

public static MySqlConnection conn;

string connString;

connString = "datasource = localhost; port = 3306; username = " + usernameBox.Text + "; password =" + passwordBox.Text + "; database = informacni\_system";

conn = new MySqlConnection(connString);

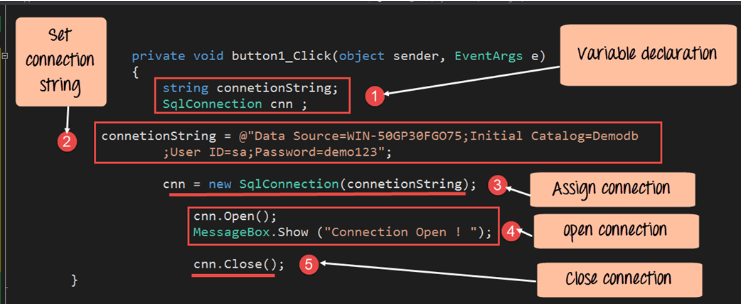
conn.Open();

Poté do metody, která po stisknutí tlačítka provede nějakou akci (ta se vygeneruje po dvojitém kliknutí na tlačítko ve formuláři.) Vytvoříme connection string (connString) a dosadíme do něj podle +obrázku, kde datasource je adresa DB serveru, port je logicky jeho port, username a password jsou přihlašovací údaje od účtu pod kterým se přihlašujeme a database je název databáze.

Tento string pak vložíme jako argument při inicializaci connu a otevřeme připojení.

### **Připojení pro MSSQL**

Celý proces je velice podobný jako u MySQL. Zde však používáme datový typ SqlConnection a jiný connection string kde Data Source představuje opět adresu serveru kde se databáze nachází, Initial Catalog je název databáze a User ID a Password jsou přihlašovací údaje uživatele který se k databázi přihlašuje. Zde jsem si k lepší představě vypůjčil obrázek z: <https://www.guru99.com/c-sharp-access-database.html>



### **Manipulace s databází**

Máme opět spoustu možností. Nejjednodušší bude pravděpodobně si vytvořit proceduru přímo v databázi a tu skrze klienta volat přes stisknutí příslušného tlačítka. Máme také možnost přímo od uživatele přijímat SQL příkazy a ty z klienta poslat do databáze, kde se aplikují. Opět záleží na rozsáhlosti a funkčnosti databáze.

1. **Volání procedur z databáze**

Tento postup se hodí pro opakované akce. Například přidávání nového uživatele.

Psát pokaždé nový SQL příkaz pro přidání nového uživatele by bylo náročné jak časově, tak znalostně např. pro běžného bankovního pracovníka.

Řekneme tedy, že máme např. v databázi proceduru, která nám vytvoří nový profil a máme formulář v C# ze kterého chceme zadané informace uložit do databáze. Procedura má vstupní hodnoty, které uloží a tak lze pouze sesumírovat veškeré argumenty do jeho stringu a ten následně dosadit do stringu, který je SQL příkaz, jenž se přes datový typ MySqlCommand/SqlCommand (záleží na použité databázi) pošle do query databáze.

private void sendRequest()

{

string createString = "'" + firstNameBox.Text + "','" + lastNameBox.Text + "','" + usernameBox.Text + "','" + tmpPasswordBox.Text + "','" + emailBox.Text + "','" + phoneBox.Text + "','" + motherPhoneBox.Text + "','" + fatherPhoneBox.Text + "','" + classBox.Text + "','"+officeBox.Text + "','" + indentificationBox.Text+"'";

string commandString = "call create\_profile("+createString +");";

MySqlCommand create = new MySqlCommand(commandString, LoginForm.conn);

create.ExecuteNonQuery();

}

Pokud bychom chtěli selectovat přes proceduru, fungovalo by to na podobném principu s tím rozdílem, že je potřeba vypsat výsledek. Pokud bychom chtěli například najít profil podle příjmení, můžeme například použít stejný formulář jako pro vytváření, ale upravíme ho tak, že budeme mít textbox a tlačítko, což nám poslouží jako searchbar, a po kliknutí na tlačítko se do formuláře zapíše vše o uživateli, kterého najdeme podle zadaného příjmení. Vzhledem k tomu že select vrací celou tabulku, je potřeba si výstup upravit a přerozdělit dílčí výsledky do příslušných textboxů.

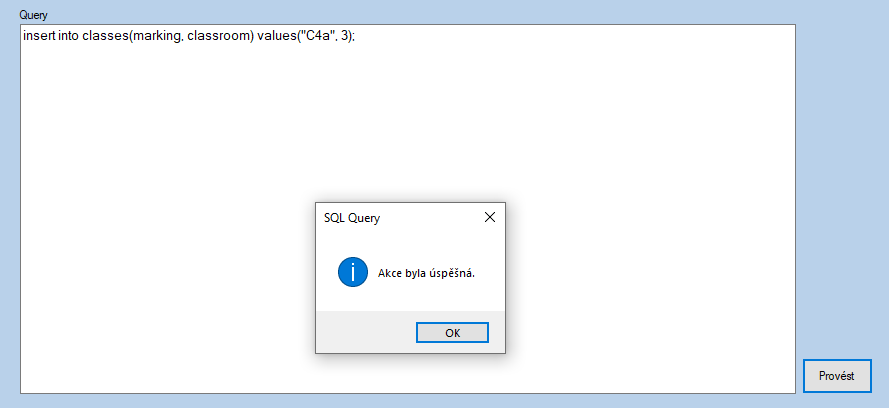
1. **Posílání SQL příkazů do databáze**

Toto se hodí zejména při jednorázových a zejména neobvyklých akcích a hlavně pro testovací či administrátorské účely přímo z klienta. Obecně záleží na tom, jakou akci chceme provést.

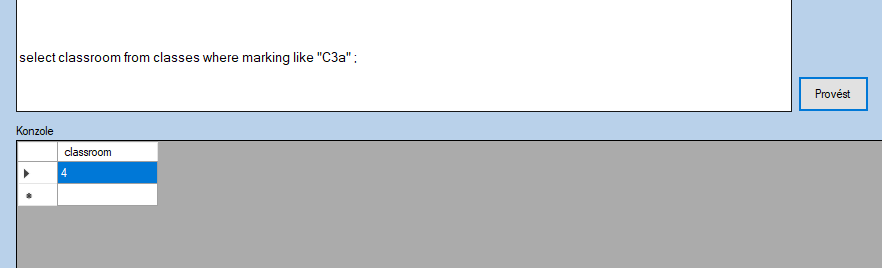
Akce jako insert nebo delete, které nevracejí žádný výsledek kromě toho, zda byla akce úspěšná, či nikoliv, jdou vyřešit poměrně snadno, horší je to však u selectu a

zejména u konzolových aplikací.

Ať už pracujeme v konzolové či okenní aplikaci, je potřeba si udělat nějakou nabídku, formulář, nazvěme si to jak chceme, kam uživatel bude moct psát SQL příkazy a odeslat je do databáze k applynutí.



Ještě před odesláním však na klientské úrovni potřebujeme odlišit, zda se jedná o select nebo ne. Pokud ne, v try bloku se odešle příkaz a pokud je úspešný, nějakým způsobem o tom informujeme uživatele, pokud ne, v catchi toto bude také nějak ošetřeno. Mohli bychom samozřejmě mluvit o ošetření špatného inputu do databáze například přes Regex, ale to je pro vysvětlovací účely bezpředmětné. Viz metoda OtherCommands()



Problém nastává, když budeme chtít po databázi nějaký výsledek. Je potřeba ho nějak zapsat a protože výsledek selectu je tabulka je potřeba v konzolové aplikaci tuto tabulku upravit na string, popřípadě stringy, aby mohl být výsledek vypsán. Pokud používáme okenní aplikaci, lze vypsat tabulku do DataGridView. Viz metoda Select().

try

{

if(queryLabel.Text.ToLower().Contains("select"))

{

Select();

}

else

{

OtherCommands();

}

}

catch

{

MessageBox.Show("Špatná syntaxe.", "SQL Query", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

private void Select()

{

MySqlCommand select = new MySqlCommand(queryLabel.Text, LoginForm.conn);

MySqlDataAdapter da = new MySqlDataAdapter(select);

DataTable result = new DataTable();

da.Fill(result);

consoleGrid.DataSource = result;

}

private void OtherCommands()

{

MySqlCommand command = new MySqlCommand(queryLabel.Text, LoginForm.conn);

command.ExecuteNonQuery();

TextBox info = new TextBox();

MessageBox.Show("Akce byla úspěšná.", "SQL Query", MessageBoxButtons.OK, MessageBoxIcon.Information);