

**School of Information Technology**

Course : Diploma in Business Informatics

Subject : ITP282 - Enterprise Application Development & Project

AY / Sem : 2018 S2

**Lab 5b**: Retrieving and Updating Data using code

OBJECTIVES:

By the end of this Practical students should be able to:

1. Use ADO.NET classes to retrieve data programmatically using parameters with queries
2. Update data with error handling

**Lab 5b: Retrieving and Updating data using code**

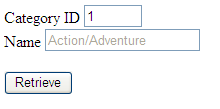
Instead of hard coding values in a SQL statement such as the one shown below, we can make our program more flexible by allowing parameters to be passed in the statement.



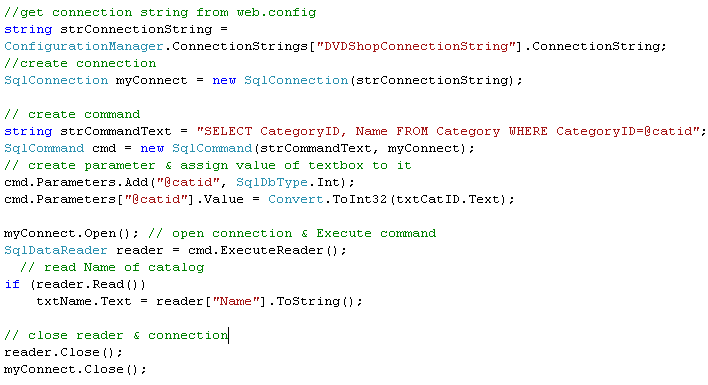
In this example, we do so by replacing the value 3 in above statement with a parameter. In MS SQL parameters must start with an @. The above statement is made more general as shown below, where @catID can be assigned a value in your code.



An example which makes use of parameter in SQL statement is shown below:



A parameter @catid is added to parameter and assigned a value from txtCatID.Text



**Exercise 1: Using Parameterised Query**

* 1. Copy the folder **Lab5b** into your computer. Ex1.aspx and Ex1.aspx.cs is a completed program from **Lab5a *Exercise 4****: Using Data Reader for database* access. Notice that under the App\_Data folder also contains DVDshop.mdf, and web.config also contains DVDShopConnectionString which the C# code uses to retrieve connection information shown below:

<connectionStrings>

<add name="DVDShopConnectionString"

connectionString="Data Source=(LocalDB)\v11.0;AttachDbFilename=|DataDirectory|\DVDShop.mdf;Integrated Security=True;User Instance=False"

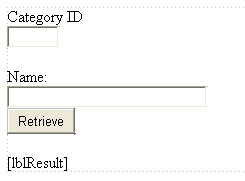
providerName="System.Data.SqlClient"/>

</connectionStrings>

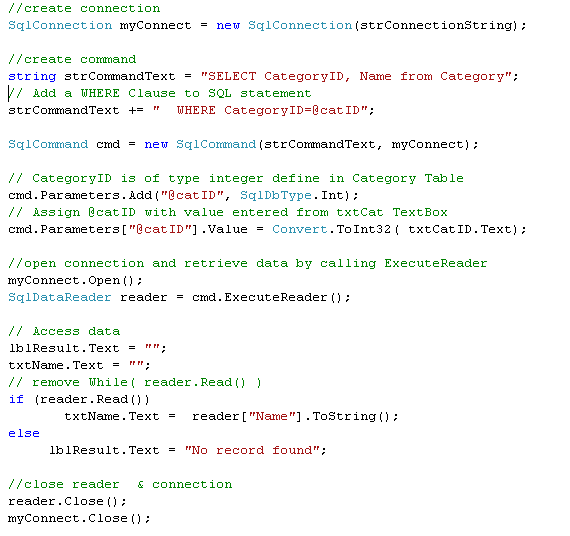
* 1. Modify the form to the one shown below:

ID= txtCatID

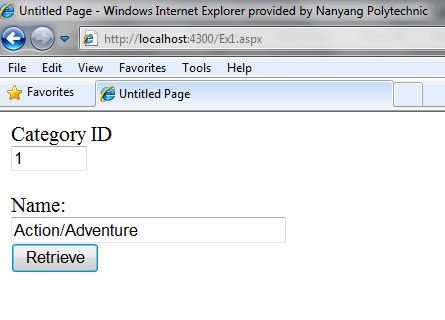
ID= txtName



* 1. In the button click event handler, modify the codes to use parameters with SQL to retrieve those records which matches the CategoryID entered in the Text Box txtCatID. In addition, instead of displaying the results in lblResult, display the results of Name in txtName. The additional codes are highlighted below:

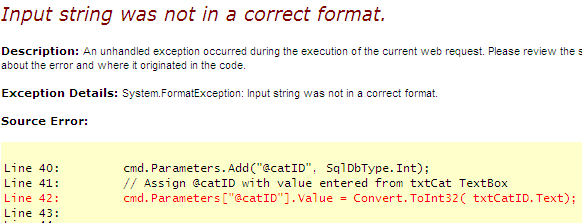


* 1. Build and run your application. Enter the number 1 into txtCatID TextBox and click on the button. What is displayed? Now, enter some non numeric inputs for example, abc into txtCatID. What happens?

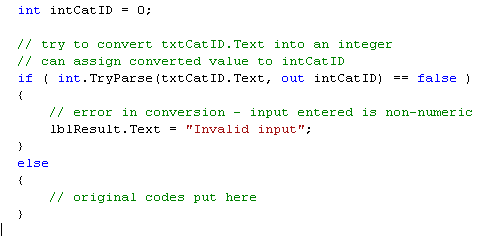


**Exercise 2: Error Handling**

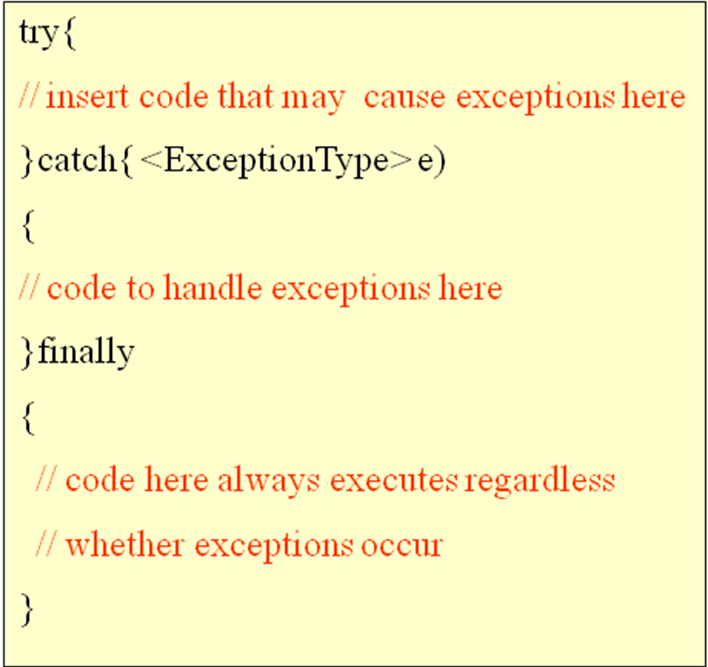
As you have seen, if you enter a non numeric value in the TextBox in above exercise, you will encounter an error:



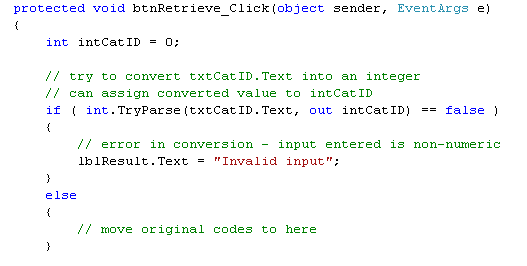
We can check whether the data entered in the Textbox is an integer using int.TryParse:



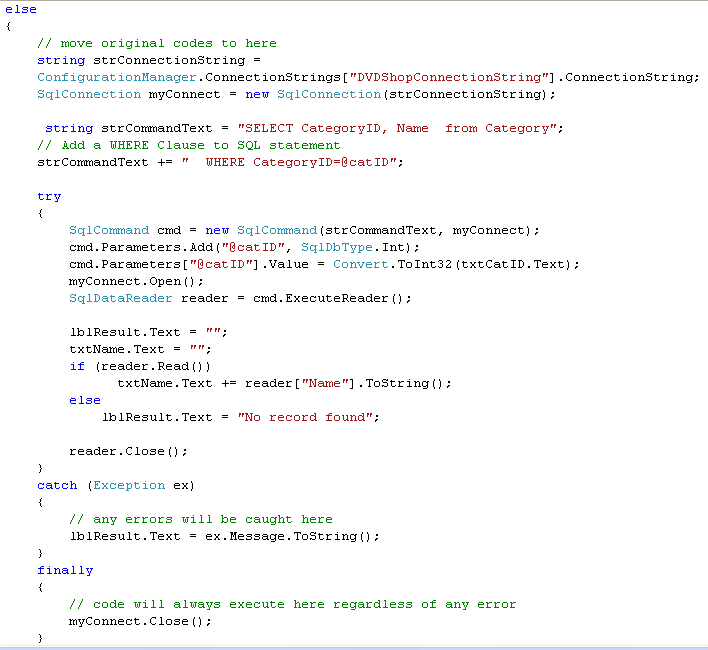
In addition, we also need to use Try-Catch to ‘catch’ any errors during code execution.



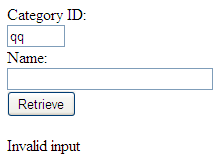
* 1. Modify your codes in button click in Ex1 by adding the following :



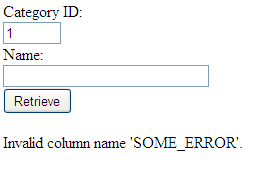
* 1. Move your original codes to within the else { } block just below the comment //move original codes to here
  2. Within the else { } block, modify the original codes to add in try-catch error handling:



* 1. To test out whether your error handling is working, run your application and enter a non-integer value into the Textbox. Int.TryParse would have caught any inputs which are non-integer:

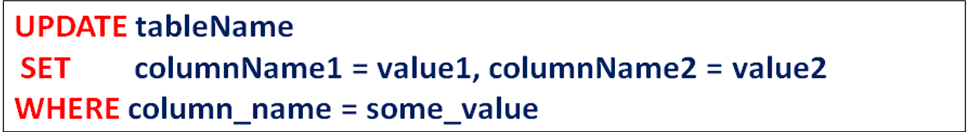


* 1. Modify your SQL statement with an erroneous Statement to “**SELECT CategoryID, Name, SOME\_ERROR FROM Category**”. Run your program. The Try-Catch would have caught the SQL error:

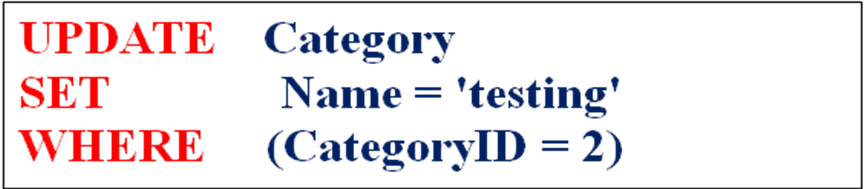


**Exercise 3: Updating Data**

Data is updated to the database using the following SQL UPDATE Statement:



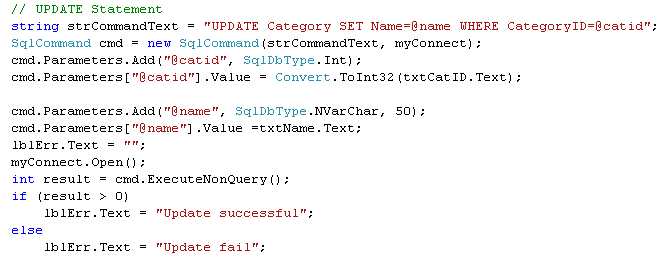
Example:



For UPDATE, DELETE and INSERT Statements, we make use of the method **ExecuteNonQuery** instead of Instead of ExecuteReader as shown below.

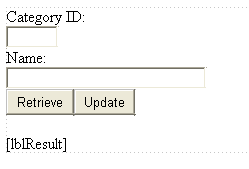
ExecuteNonQuery is used for UPDATE. It returns number of records updated.

Column Name is of data type NVarChar size 50

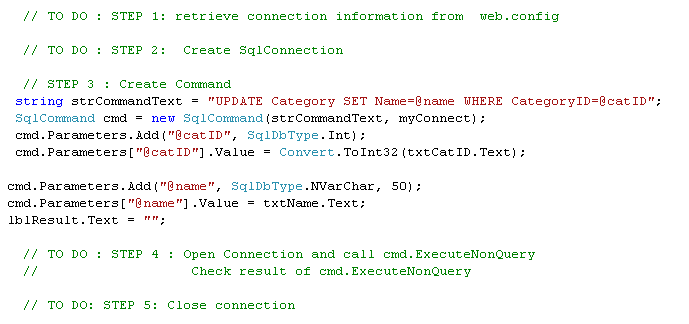


* 1. Continuing from your last exercise, add in an Update button.

ID= btnUpdate



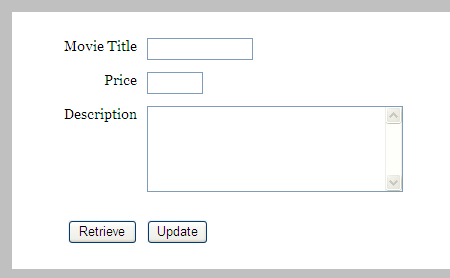
* 1. In btnUpdate Click event handler, type in following codes. Note that those comments TO DO: require you to write additional codes in order for the program to work:



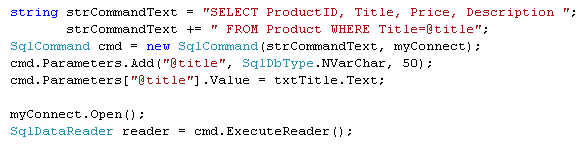
* 1. Build and run your application. Test out to see if the column Name is updated with data entered into txtName.Text.
  2. Add in int.TryParse to make sure an integer is entered into the Text Box for category ID.
  3. Add in try-catch in your SQL related codes to check that errors are caught and handled.

**Exercise 4: Retrieving and Updating Data**

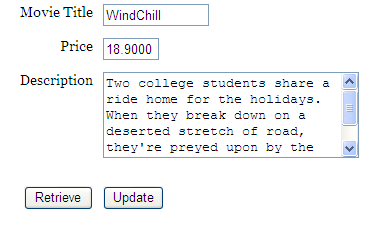
* 1. Open up Ex2.aspx. It should look similar to the following:



* 1. Write codes so that when the user enters a title into txtTitle TextBox, and btnRetrieve is clicked, both the price and description will be retrieved from the Product Table and displayed in the respective Text Boxes. The partial relevant SQL Select statements is shown here:



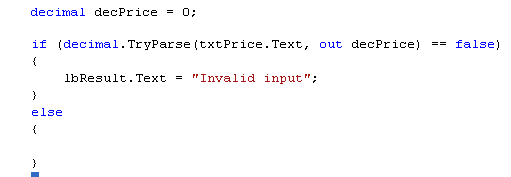
An example is shown:



* 1. . Write codes so that when the btnUpdate button is clicked, the price and description are updated to the database. The partial relevant SQL UPDATE statement is shown here:



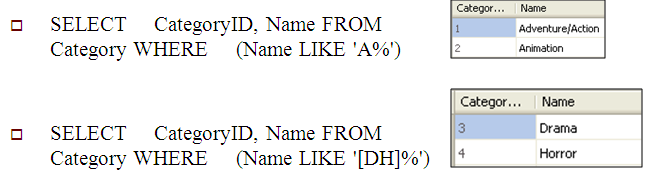
* 1. Improve your code by checking the data type of input entered for the Price is a decimal data type:



* 1. Add in Try Catch error handling when you perform the Updates in your code.
  2. Build and run your application.

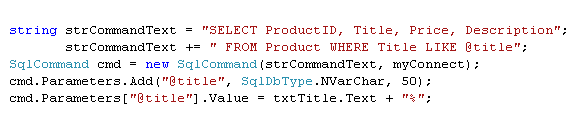
**Exercise 5: Using LIKE clause for matching.**

The LIKE clause allows us to find matching records, for example:



In the first example, any Name starting with A following by any number of characters will be retrieved. In the second example, any Name starting with A or H followed by any number of characters will be retrieved.

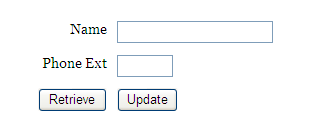
* 1. Continuing from the last exercise, we want to modify the program such that the user need not enter the *full* title of the DVD movie. For example, if he enters Wind, then the nearest matching record WindChill will be retrieved. Modify the SQL SELECT statement as follows:



* 1. Build and run your application. Enter Wind into txtTitle Text Box and click on btnRetrieve. What happens?

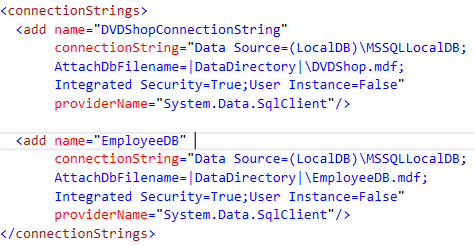
**(Optiona) Exercise 6: Additional Practice**

* 1. Open up Ex3.aspx. In this exercise, you will write a program to perform a telephone number look up application. For example, when partial name Jess is entered (full name is Jessica Ruvalcaba), her phone extension 1820 will be displayed. Note that the database EmployeeDB.mdf is given to you.

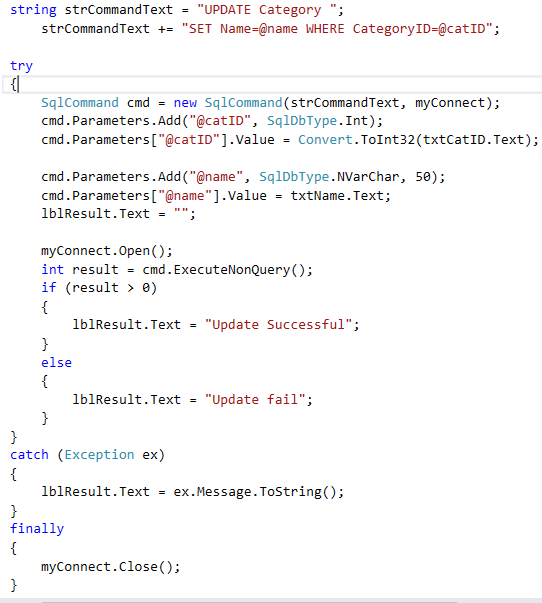
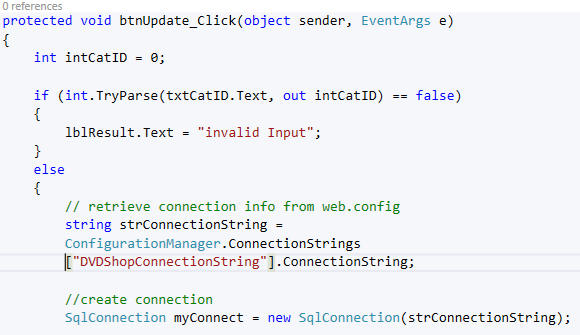
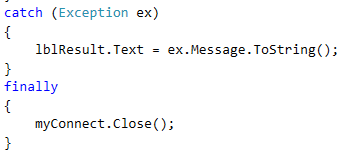
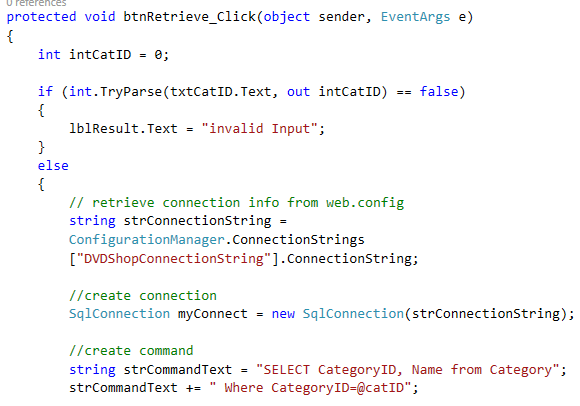


* 1. In addition, when the Update button is clicked, the new extension entered into the Text Box will be updated to database.
  2. Use try-catch to make your code more robust.

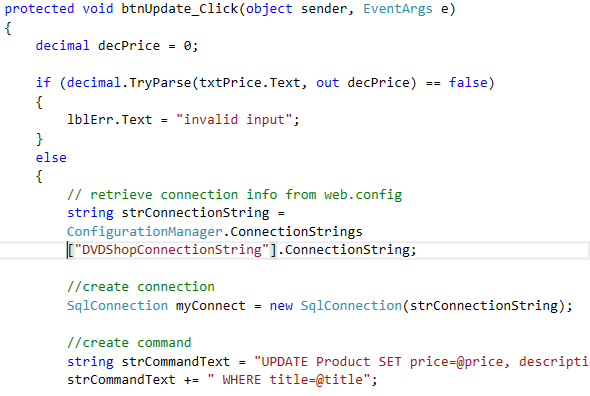
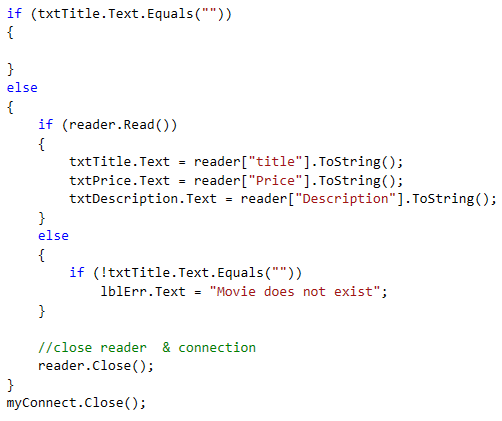
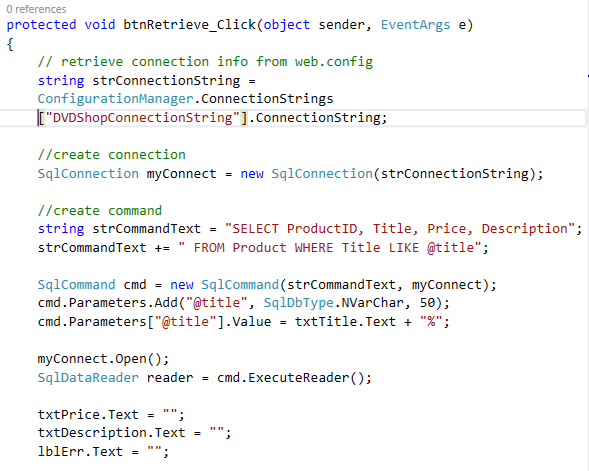
**========== End ==========**



**EX1.aspx.cs**



**EX2.aspx.cs**



**EX3.aspx.cs**

