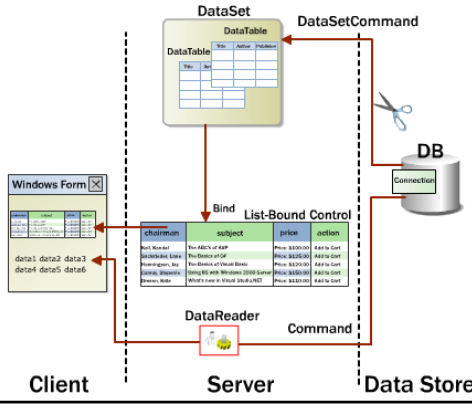
Using Database in Windows Forms Applications

ADO.NET (System.Data) is a set of classes that allow .NET-based applications to read and update information in databases and other data sources.

ADO.NET provides access to different data sources:

- Microsoft SQL Server
- OLE DB-compliant databases
- Microsoft Exchange Server
- XML documents

Non-relational sources



ADO.NET Objects

Data provider serves as a bridge between an application and a data source. It is used to

- Retrieve data from a data source
- Reconcile changes to that data back to the data source

Visual Studio 2003
Microsoft OLE DB Provider for SQL Server (SQL Server)
Microsoft JET 4.0 OLE DB Provider (MS Access)

Visual Studio 2005
.NET Framework Data Provider for SQL Server
.NET Framework Data Provider for OLE DB

Objects	Description
Connection	Represents a connection to the database.
Command	Accesses data directly in the database. Contains a Parameters collection that populates the input and output arguments of SQL statements or stored procedures.
DataReader	Fast, forward-only cursor that loops through a stream of rows.
DataSet	Represents a local copy of data from a data source; has a similar structure to that of the relational database; it exposes a hierarchical object model of tables, rows, columns, constraints and relationships defined for the data set.
DataAdapter	Serves as a bridge between a DataSet and a data source for retrieving and saving data.

Object Connection

Represents a connection to the database.

Object	Database Source	Data Provider
SqlConnection	MS SQL Server v.7	.NET Framework Data Provider for SQL Server
OleDbConnection	OLE DB or Microsoft SQL Server	.NET Framework Data Provider for OLE DB
OdbcConnection	ODBC	.NET Framework Data Provider for ODBC
OracleConnection	Oracle	.NET Framework Data Provider for Oracle

Property
ConnectionString — Determines connection settings to access the particular data source

```
SqlConnection sqlConnection1 = new SqlConnection();
```

(Visual Studio 2003)

```
sqlConnection1.ConnectionString =
        "workstation id=<workstation name>; packet size=4096;"
        + "integrated security=SSPI;"
        + "data source=\"<data_source_name>\";"
        + "persist security info=False;"
        + "initial catalog=<database name>";
```

(Visual Studio 2005)

```
sqlConnection1.ConnectionString =
        "Data Source=<data source name>;"
        + " Initial Catalog=<database name>;"
        + " Integrated Security=True";
```

Object Command

Uses SQL statements or stored procedures to retrieve data and result sets are returned in the form of streams that can be
- read by DataReader or
- pushed into DataSet objects.

Command Object	Database Source
SqlCommand	MS SQL Server v.7
OleDbCommand	OLE DB or Microsoft SQL Server
OdbcCommand	ODBC
OracleCommand	Oracle

public virtual int ExecuteNonQuery();

Method ExecuteNonQuery executes any SQL statement and returns the number of the result rows.

Object DataReader

Reads data from the stream.

DataReader Object	Database Source
SqlDataReader	MS SQL Server v.7
OleDbDataReader	OLE DB or Microsoft SQL Server
OdbcDataReader	ODBC
OracleDataReader	Oracle

public xxxDataReader ExecuteReader ()

The Command.ExecuteReader method returns a DataReader object. It can execute any SELECT statement or stored procedure that contains a SELECT statement. When the application processes a result set with a DataReader the connection is kept busy and when it finishes the DataReader should be closed.

Object DataAdapter

Represents a local copy of data from a data source. Serves as a bridge between a DataSet and a data source.

DataAdapter Object	Database Source
SqlDataAdapter	MS SQL Server v.7
OleDbDataAdapter	OLE DB or Microsoft SQL Server
OdbcDataAdapter	ODBC
OracleDataAdapter	Oracle

public abstract int Fill (DataSet dataSet);

Method Fill adds/refreshes rows in the DataSet to match those in the data source using the DataSet name, and creates a DataTable with a name "Table"; returns the number of affected rows.

public abstract int Update (DataSet dataSet);

Method Update calls the respective INSERT, UPDATE, or DELETE statements for each inserted, updated, or deleted row in the specified DataSet from a DataTable named "Table"; returns the number of affected rows.

DataAdapter represents a set of database commands and a database connections to fill a DataSet and update the data source.

Properties:

SelectCommand – Gets/sets a SQL statement or stored procedure to select records into the data source; used during Fill to select records from the database for placement in the DataSet.

InsertCommand – Gets/sets a SQL statement or stored procedure to insert new records into the data source; used during Update to insert records into the database that correspond to new rows in the DataSet.

DeleteCommand – Gets/sets a SQL statement or stored procedure to delete records into the data source; used during Update to delete records in the database that correspond to deleted rows in the DataSet.

UpdateCommand – Gets/sets a SQL statement or stored procedure to update records into the data source; used during Update to update records in the database that correspond to modified rows in the DataSet.

InsertCommand, UpdateCommand and DeleteCommand are used to modify the data in the data source.

TableMappings – Gets a collection that provides the master mapping between a source table and a DataTable.

Object DataSet

Represents data in a local cache that functions as a disconnected relational view of the data. The connection is not need to be active for an application to view and manipulate data in a DataSet. This disconnected architecture uses database resources only when reading from, or writing to, the data source.

Properties

Tables	Gets a collection of DataTable objects
Relations	Gets a collection of DataRelation objects

Methods

Clear	Clears the DataSet of any data by removing all rows in all tables.

Object DataTable

Represents the tables in a DataSet.

Properties

Columns	Gets a collection of DataColumn objects
Rows	Gets a collection of DataRow objects
Constraints	Gets a collection of Constraint (UniqueConstraint and ForeignKeyConstraint) objects, representing constraints on DataColumn objects
ChildRelations	Gets a collection of DataRelation objects, representing a relationship to a column in another table in the DataSet (to create links between primary keys and foreign keys in the tables)

Object DataColumn

Represents the columns in a DataTable.

Property

DataType	Determines the kind of data that each column contains.

Object DataRow

Represents the rows in a a DataSet.

Typed DataSet

Derives from the base DataSet class.

Provides type checking at compile time.

Provides faster access to tables and columns in the dataset by name, instead of collection-based methods.

```
studentsDataSet1.Tables("Students");
studentsDataSet1.Students;
```

Untyped dataset – access through collection at run time

Typed dataset – direct access at the compile time

Generated from XML Schema (.xsd) files by using the XSD.exe tool.

Object DbCommandBuilder (Visual Studio 2005)

Automatically generates single-table commands used to reconcile changes made to a DataSet with the associated database.

DbCommandBuilder Object	Database Source
SqlCommandBuilder	MS SQL Server v.7
OleDbCommandBuilder	OLE DB or Microsoft SQL Server
OdbcCommandBuilder	ODBC
OracleCommandBuilder	Oracle

Control DataGrid (Visual Studio 2003)

Displays data in a series of rows and columns. The grid is bound to a data source with a single table or with multiple tables. The grid displays the hierarchical relationships between tables. The control provides a user interface for a dataset, navigation between related tables, and rich formatting and editing capabilities.

Properties:

DataSource	Gets/sets the source (DataTable, DataSet, etc.), containing a list of values used to populate the items within the control.
DataMember	Gets/sets the specific data member in a multimember data source to bind to a data listing control.

Methods:

```
public void SetDataBinding(object dataSource,
                           string dataMember);
```

Sets the DataSource and DataMember properties at run time.

Control DataGridView (Visual Studio 2005)

Displays data in a customizable grid (replaces and adds functionality to the DataGrid control).

Properties:

DataMember Gets/sets the name of the list or table in the data source for which the DataGridView is displaying data.

DataSource Gets/sets the data source that the DataGridView is displaying data for. The BindingSource component is the preferred data source.

Component BindingSource (Visual Studio 2005)

Encapsulates the data source for a form.

Properties:

DataMember Gets/sets the specific list in the data source to which the connector currently binds to.

DataSource Gets/sets the data source that the connector binds to.

Control BindingNavigator (Visual Studio 2005)

Represents the navigation and manipulation user interface (UI) for controls on a form that are bound to data. Creates a standardized means for users to search and change data on a Windows Form. BindingNavigator is used with the BindingSource component to enable users to move through data records on a form and interact with the records.

Properties:

BindingSource Gets/sets the BindingSource component that is the source of data.

Control TableAdapter (Visual Studio 2005)

Provides communication between the application and a database by executing SQL statements and stored procedures against a database.

Creates with the Dataset Designer inside of strongly typed datasets:

- During creation of a new dataset with the Data Source Configuration Wizard
- In existing datasets with the TableAdapter Configuration Wizard
- By dragging database objects from Server Explorer onto the Dataset Designer.

The methods TableAdapter.Insert, TableAdapter.Update, and TableAdapter.Delete can be called directly to manipulate data in the database.

Control ToolStrip (Visual Studio 2005)

Provides a container for Windows toolbar objects.

ToolStrip is the container for:

- ToolStripButton
- ToolStripComboBox
- ToolStripSplitButton
- ToolStripLabel
- ToolStripSeparator
- ToolStripDropDownButton
- ToolStripProgressBar
- ToolStripTextBox

Populating a Dataset – Using a DataAdapter to Fill the DataSet (VisualStudio 2003)

```
// Declarations
SqlConnection sqlConnection1;
SqlDataAdapter sqlDataAdapter1;

SqlCommand sqlSelectCommand1;
SqlCommand sqlInsertCommand1;
SqlCommand sqlDeleteCommand1;
SqlCommand sqlUpdateCommand1;

DataSet dataSet11;

DataGrid dataGrid1;
```

```
// Initializing the connection
sqlConnection1 = new SqlConnection();
sqlConnection1.ConnectionString = "workstation id=<workstation name>;
  packet size=4096;
  integrated security=SSPI;
  data source=\"<data source name>\";
  persist security info=False;
  initial catalog=<data base name>;
```

```
// Initializing the data adapter
sqlDataAdapter1 = new SqlDataAdapter();
sqlDataAdapter1.SelectCommand  = sqlSelectCommand1;
sqlDataAdapter1.InsertCommand  = sqlInsertCommand1;
sqlDataAdapter1.DeleteCommand  =sqlDeleteCommand1;
sqlDataAdapter1.UpdateCommand =sqlUpdateCommand1;
sqlDataAdapter1.TableMappings.AddRange
  (new System.Data.Common.DataTableMapping[]
    { new System.Data.Common.DataTableMapping
       ("<table name from the data source>",
        "<table name from the DataSet>",
        new System.Data.Common.DataColumnMapping[] {
          new System.Data.Common.DataColumnMapping
            ("<column name from the data source>",
             "<column name from the DataSet>"),
          …
          }
       )
    }
  );
```

S2

```
sqlSelectCommand1.CommandText = <text of SQL SELECT statement>;
sqlSelectCommand1.Connection = sqlConnection1;
sqlInsertCommand1.CommandText = <text of SQL INSERT statement>;
sqlInsertCommand1.Connection = sqlConnection1;
sqlDeleteCommand1.CommandText = <text of SQL DELETE statement>;
sqlDeleteCommand1.Connection = sqlConnection1;
sqlUpdateCommand1.CommandText = <text of SQL UPDATE statement>;
sqlUpdateCommand1.Connection = sqlConnection1;

// SELECT statement to query the database
sqlDataAdapter1.SelectCommand.CommandText =
<text of SQL SELECT statement>;

// Using Fill method to populate a table in a dataset
dataSet11.Clear();
sqlDataAdapter1.Fill (dataSet11.Tables["<table name>"]);

// Displaying the result dataset
dataGrid1.SetDataBinding (dataSet11, "<table name>");
```

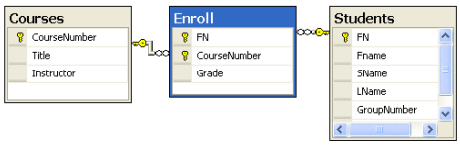
Populating a Dataset – Using a TableAdapter to Fill the DataSet (Visual Studio 2005)

```
System.ComponentModel.Container components =
          new System.ComponentModel.Container ();

DBNameDataSet dataSet = new DBNameDataSet ();

BindingSource tableNameBindingSource = new BindingSource (components);
tableNameBindingSource.DataMember = "TableName";
tableNameBindingSource.DataSource = dataSet;

DBNameDataSetTableAdapters.TableNameTableAdapter
          tableNameTableAdapter = new
                    DBNameDataSetTableAdapters.TableNameTableAdapter();
 tableNameTableAdapter.ClearBeforeFill = true;

BindingNavigator tableNameBindingNavigator =
          new BindingNavigator (components);
tableNameBindingNavigator.BindingSource = tableNameBindingSource;
```

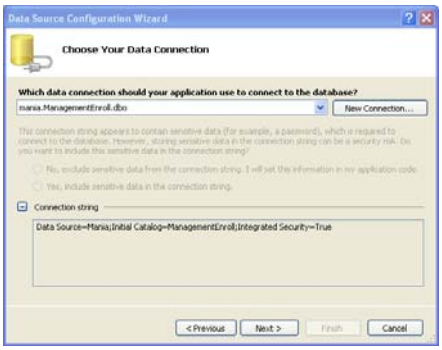
```
DataGridView grid = new DataGridView();
grid.DataSource = tableNameBindingSource;

// Using Fill method to populate a table in a dataset
tableNameTableAdapter.Fill (dataSet.TableName);
```

S2 е на таблиц
SB; 28.5.2004 г.

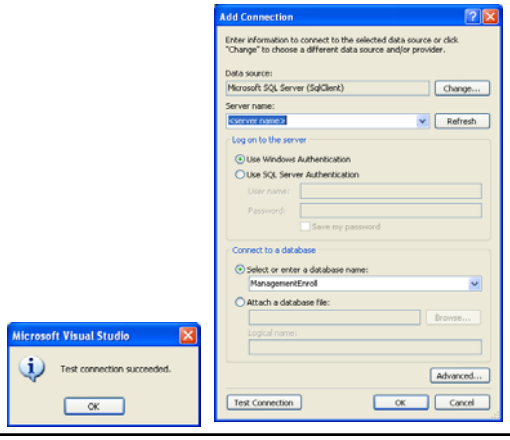
Example: Performs a simple query to the ManagementEnroll database (MS SQL Server database source) that retrieves the entire Courses table and displays the data.



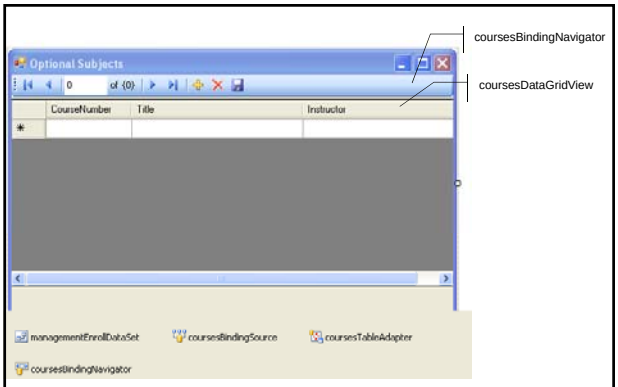
1. Create a new Windows Application project.
2. Connect the application to the ManagementEnroll database
   - Data ⇒ Add New Data Source ... ⇒
     Data Source Configuration Wizard
   - <L> Database ⇒ Next ⇒
     Choose Your Data Connection



- ⇒ New Connection ...
  Add Connection

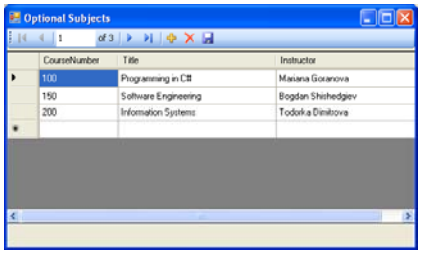


- Clear the check box to save the Connection string in the application.
- ⇒ Next
  Choose Your Database Objects
- Choose Courses from Tables
  ⇒ Finish
- Data ⇒ Show Data Sources
  Data Sources
- Drag the table Courses



A DataSet, BindingSource, TableAdapter, BindingNavigator appear in the component tray. A DataGridView control and a tool strip for navigating records appear on the form.

3. Run the application




```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;

namespace DatabaseManagementEnroll
{
   public partial class Form1 : Form
   {
      public Form1()
      {
         InitializeComponent();
      }
```

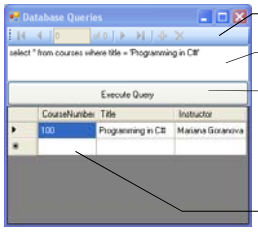
```
      private void coursesBindingNavigatorSaveItem_Click(object sender,
                                                         EventArgs e)
      {
         this.Validate();
         this.coursesBindingSource.EndEdit();
         this.coursesTableAdapter.Update
                                  (this.managementEnrollDataSet.Courses);
      }

      private void Form1_Load(object sender, EventArgs e)
      {
          this.coursesTableAdapter.Fill(this.managementEnrollDataSet.Courses);
      }
   }
}
```

```
namespace DatabaseManagementEnroll
{
   partial class Form1
   {
      private System.ComponentModel.IContainer components = null;

      protected override void Dispose(bool disposing) {   …        }

      #region Windows Form Designer generated code

      private void InitializeComponent()
      {   this.components = new System.ComponentModel.Container();
         this.managementEnrollDataSet =
                  new DatabaseManagementEnroll.ManagementEnrollDataSet();
         this.coursesBindingSource =
                  new System.Windows.Forms.BindingSource(this.components);
         this.coursesTableAdapter =
               new DatabaseManagementEnroll.
               ManagementEnrollDataSetTableAdapters.CoursesTableAdapter();
         this.coursesBindingNavigator =
               new System.Windows.Forms.BindingNavigator(this.components);
```

```
         this.coursesDataGridView =
                  new System.Windows.Forms.DataGridView();

         this.coursesBindingSource.DataMember = "Courses";
         this.coursesBindingSource.DataSource =
                  this.managementEnrollDataSet;

         this.coursesTableAdapter.ClearBeforeFill = true;

         this.coursesBindingNavigator.BindingSource =
                  this.coursesBindingSource;

         this.coursesDataGridView.DataSource = this.coursesBindingSource;

         this.Controls.Add(this.coursesDataGridView);
         this.Controls.Add(this.coursesBindingNavigator);
         this.Name = "Form1";
         this.Text = "Optional Subjects";
         this.Load += new System.EventHandler(this.Form1_Load);
      }
      #endregion
```

```
      private ManagementEnrollDataSet managementEnrollDataSet;
      private System.Windows.Forms.BindingSource coursesBindingSource;
      private DatabaseManagementEnroll.
         ManagementEnrollDataSetTableAdapters.CoursesTableAdapter
         coursesTableAdapter;
      private System.Windows.Forms.BindingNavigator
         coursesBindingNavigator;
      private System.Windows.Forms.DataGridView coursesDataGridView;
   }
}
```

Example: Executes the SQL SELECT statements on the ManagementEnroll database and displays the results in the DataGridView control.



bindingSource1 bindingNavigator1

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using System.Data.SqlClient;

namespace DatabaseQueries
{
   public partial class Form1 : Form
   {
      public Form1()
      {   InitializeComponent();   }

      private void submitButton_Click(object sender, EventArgs e)
      {
         dataGridView1.DataSource = bindingSource1;
         GetData (queryTextBox.Text);
      }
```

```
   private void GetData(string selectCommand)
   {  try
      {
         // Declare an instance of SqlDataAdapter with a select command
         // and a connection string
         string connectionString = "Data Source=Mania;" +
                  "Initial Catalog=ManagementEnroll;" +
                  "Integrated Security=True";
         SqlDataAdapter adapter =
                  new SqlDataAdapter(selectCommand, connectionString);
         // Declare an instance of SqlCommandBuilder to generate
         // automatically single-table commands used with the
         // SqlDataAdapter
         SqlCommandBuilder commandBuilder =
                  new SqlCommandBuilder(adapter);
         // Declare a DataTable
         DataTable table = new DataTable();
         // Set the locale information used to compare strings within the table
         // to neutral culture
         table.Locale = System.Globalization.CultureInfo.InvariantCulture;
```

```
         // Use the Fill method of the adapter to populate table
         adapter.Fill (table);
         // Set the DataSource property of bindingSource1 to table
         bindingSource1.DataSource = table;
         // Set the AutoResizeColumns property of dataGridView1
         dataGridView1.AutoResizeColumns (
         DataGridViewAutoSizeColumnsMode.AllCellsExceptHeader);
      }
      catch (SqlException e)
      {  MessageBox.Show(e.Message);   }
   }
  }
}
```

Passing Parameters to SELECT Statement – Parameters Collection of a Command Object

```
// SELECT statement – SQL Client
sqlDataAdapter1.SelectCommand.CommandText=
"SELECT * FROM TABLE WHERE (Key=@Key)";
// Set the value to be updated – Value property
sqlDataAdapter1.SelectCommand.Parameters["@Key"].Value =
                                                  <value>;
sqlDataAdapter1.Fill(dataSet11.Tables["<table name>"]);


// SELECT statement – OLE or Odbc
oleDbDataAdapter1.SelectCommand.CommandText=
"SELECT * FROM TABLE WHERE (Key=?)";
```

Update Data to a Data Source – Method Update

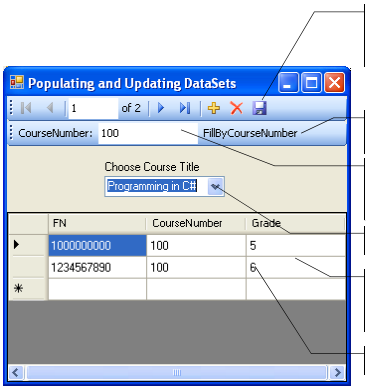
InsertCommand, UpdateCommand and DeleteCommand properties of the DataAdapter identify the changes occurring in the INSERT, UPDATE and DELETE statements. The Parameters collection passes the information to the Command object to identify the column, data type, size, and data.

```
public SqlParameter Add (string parameterName,
        SqlDbType sqlDbType, int size, string sourceColumn);
```

The method adds a parameter of SqlParameter type to the collection with a name parameterName, data type sqlDbType, column size and a column name in the data source sourceColumn.

```
// INSERT statement – SQL Client
sqlDataAdapter1.InsertCommand.CommandText=
"INSERT INTO <table name> (<attribute1>, <attribute2>)
VALUES(@<attribute1>, @<attribute2>)";

// Adding parameters with the Add method and setting their values
sqlDataAdapter1.InsertCommand.Parameters.Add
("@<attribute1>", SqlDbType.<type>, <size>).Value = <value>;
sqlDataAdapter1.InsertCommand.Parameters.Add
("@<attribute2>", SqlDbType.<type>, <size>)= <value>;

// Updating the data source
sqlDataAdapter1.Update(dataSet1, "<table name>");
```

Example: Populate a dataset and update a database with changes made to the dataset. Fill the dataset with the Enroll table of the ManagementEnroll database. Create a new dataset and fill with the Courses table. Write all of the code required to populate the dataset for the Courses table. Use the code generated by the TableAdapter Query Configuration Wizard for the Enroll table. Update the database with any changes to the Enroll table.



Click the Save button to update the data into the database.

Click the button to execute the query.

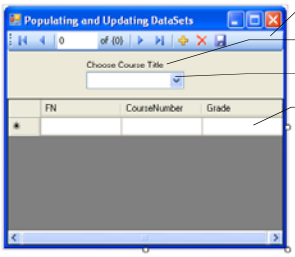
The course number changes according to the selected course title.

Select a course title.

DataGridView displays the corresponding data from the Enroll table.

The user changes data.

1. Create a new Windows Application project.
2. Connect the application to the ManagementEnroll database using Data Source Configuration Wizard, choose the Enroll table as data base object and drag Enroll.



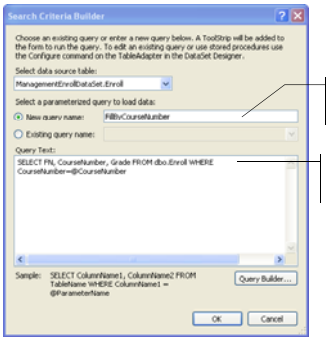
enrollBindingNavigator

courseTitleLabel

courseTitleComboBox

enrollDataGridView

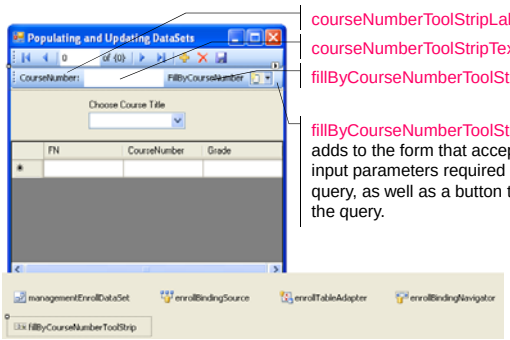
3. Create TableAdapter query
   - <R> enrollDataAdapter ⇒ Add Query … ⇒ Search Criteria Builder



Query appears as a method FillByCourseNumber

SELECT statement with a parameter

⇒ OK



courseNumberToolStripLabel

courseNumberToolStripTextBox

fillByCourseNumberToolStripButton

fillByCourseNumberToolStrip adds to the form that accepts any input parameters required by the query, as well as a button to run the query.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using System.Data.SqlClient;

namespace PopulatingAndUptadingDataSets
{
    public partial class Form1 : Form
    {
        private string courseNumber;

        public Form1()
        {
            InitializeComponent();
        }
```

```
        // Declare an instance of SqlCommandBuilder to generate
        // automatically single-table commands used with the SqlDataAdapter
        SqlCommandBuilder commandBuilder =
                new SqlCommandBuilder(adapter);
        // Declare a DataTable and initialize with the string Course
        DataTable table = new DataTable("Courses");
        // Set the locale information used to compare strings within the table
        // to neutral culture
        table.Locale = System.Globalization.CultureInfo.InvariantCulture;
        // Use the Fill method of the adapter to populate table
        adapter.Fill (table);
        // Create a new dataset
        DataSet dataSet = new DataSet();
        // Add table to the Tables collection of the dataset
        dataSet.Tables.Add (table);
```

```
    private void UpdateTable()
    {
        // Clear the text in the courseNumberToolStripTextBox
        courseNumberToolStripTextBox.Text = "";
        // Update the course number according the selected course title
        courseNumber = courseTitleComboBox.SelectedValue.ToString();
    }

    private void comboBox_SelectedIndexChanged(object sender,
                                               EventArgs e)
    {
        // Update the course number according the selected course title
        UpdateTable();
        // Set the text of the courseNumberToolStripTextBox to the course
        //  number
        courseNumberToolStripTextBox.Text = courseNumber;
    }
```

```
    private void enrollBindingNavigatorSaveItem_Click(object sender,
                                                      EventArgs e)
    {
        this.Validate();
        this.enrollBindingSource.EndEdit();
        this.enrollTableAdapter.Update(this.managementEnrollDataSet.Enroll);
    }

    private void Form1_Load(object sender, EventArgs e)
    {
        this.enrollTableAdapter.Fill(this.managementEnrollDataSet.Enroll);
        try
        {
            // Declare an instance of SqlDataAdapter with a select command and
            // a connection string
            string connectionString = "Data Source=Mania;" +
                    "Initial Catalog=ManagementEnroll;Integrated Security=True";
            string selectCommand =
                    "SELECT CourseNumber, Title FROM Courses";
            SqlDataAdapter adapter =
                    new SqlDataAdapter (selectCommand, connectionString);
```

```
            // Bind courseTitleComboBox control to the Courses table
            // Set the DataSource property of courseTitleComboBox to the
            // Courses table in the dataSet
            courseTitleComboBox.DataSource = dataSet.Tables["Courses"];
            // Set the DisplayMember property of courseTitleComboBox to the
            // Title column
            courseTitleComboBox.DisplayMember = "Title";
            // Set the ValueMember property of courseTitleComboBox to the
            // CourseNumber column
            courseTitleComboBox.ValueMember = "CourseNumber";
            // Update the course number according the selected course title
            UpdateTable();
            // Add an event handler for the SelectedIndexChanged event of the
            // courseTitleComboBox
            courseTitleComboBox.SelectedIndexChanged += new
                    System.EventHandler(comboBox_SelectedIndexChanged);
        }
        catch (SqlException exp)
        {   MessageBox.Show(exp.Message);   }
    }
```

```
    private void fillByCourseNumberToolStripButton_Click(object sender,
                                                         EventArgs e)
    {
        try
        {
            this.enrollTableAdapter.FillByCourseNumber
                    (this.managementEnrollDataSet.Enroll,
                     ((int)(System.Convert.ChangeType
                       (courseNumberToolStripTextBox.Text, typeof(int)))));
        }
        catch (System.Exception ex)
        {
            System.Windows.Forms.MessageBox.Show(ex.Message);
        }

    }
```

```
namespace PopulatingAndUptadingDataSets
{
   partial class Form1
   {
      private System.ComponentModel.IContainer components = null;

      protected override void Dispose(bool disposing) { …   }

      #region Windows Form Designer generated code

      private void InitializeComponent()
      {
         this.enrollBindingNavigator =
          new System.Windows.Forms.BindingNavigator(this.components);

         this.enrollBindingNavigatorSaveItem =
          new System.Windows.Forms.ToolStripButton();

         this.enrollDataGridView = new System.Windows.Forms.DataGridView();
         this.courseTitleLabel = new System.Windows.Forms.Label();
         this.courseTitleComboBox = new System.Windows.Forms.ComboBox();
```

```
         this.enrollBindingSource =
          new System.Windows.Forms.BindingSource(this.components);
         this.managementEnrollDataSet =
          new PopulatingAndUptadingDataSets.ManagementEnrollDataSet();
         this.enrollTableAdapter =
          new PopulatingAndUptadingDataSets.
          ManagementEnrollDataSetTableAdapters.EnrollTableAdapter();
         this.fillByCourseNumberToolStrip =
          new System.Windows.Forms.ToolStrip();
         this.courseNumberToolStripLabel =
          new System.Windows.Forms.ToolStripLabel();
         this.courseNumberToolStripTextBox =
          new System.Windows.Forms.ToolStripTextBox();
         this.fillByCourseNumberToolStripButton =
          new System.Windows.Forms.ToolStripButton();

         this.enrollBindingNavigator.BindingSource = this.enrollBindingSource;

         this.enrollBindingNavigatorSaveItem.Click +=
          new System.EventHandler(this.enrollBindingNavigatorSaveItem_Click);
```

```
         this.enrollDataGridView.DataSource = this.enrollBindingSource;
         this.enrollDataGridView.Dock =
                  System.Windows.Forms.DockStyle.Bottom;

          this.courseTitleLabel.Text = "Choose Course Title";

         this.enrollBindingSource.DataMember = "Enroll";
         this.enrollBindingSource.DataSource = this.managementEnrollDataSet;

         this.managementEnrollDataSet.DataSetName =
                  "ManagementEnrollDataSet";

         this.enrollTableAdapter.ClearBeforeFill = true;

         this.fillByCourseNumberToolStrip.Text =
                  "fillByCourseNumberToolStrip";

         this.courseNumberToolStripLabel.Text = "CourseNumber:";

         this.courseNumberToolStripTextBox.Name =
                  "courseNumberToolStripTextBox";
```

```
         this.fillByCourseNumberToolStripButton.Text = "FillByCourseNumber";
         this.fillByCourseNumberToolStripButton.Click +=  new
          System.EventHandler(this.fillByCourseNumberToolStripButton_Click);

         this.Controls.Add(this.fillByCourseNumberToolStrip);
         this.Controls.Add(this.courseTitleComboBox);
         this.Controls.Add(this.courseTitleLabel);
         this.Controls.Add(this.enrollDataGridView);
         this.Controls.Add(this.enrollBindingNavigator);
         this.Name = "Form1";
         this.Text = "Populating and Updating DataSets";
         this.Load += new System.EventHandler(this.Form1_Load);
      }
      #endregion

      private ManagementEnrollDataSet managementEnrollDataSet;
      private System.Windows.Forms.BindingSource enrollBindingSource;
      private PopulatingAndUptadingDataSets.
          ManagementEnrollDataSetTableAdapters.EnrollTableAdapter
          enrollTableAdapter;
      private System.Windows.Forms.BindingNavigator enrollBindingNavigator;
      …
```

```
      private System.Windows.Forms.ToolStripButton
          enrollBindingNavigatorSaveItem;
      private System.Windows.Forms.DataGridView enrollDataGridView;
      private System.Windows.Forms.Label courseTitleLabel;
      private System.Windows.Forms.ComboBox courseTitleComboBox;

      private System.Windows.Forms.ToolStrip fillByCourseNumberToolStrip;
      private System.Windows.Forms.ToolStripLabel
          courseNumberToolStripLabel;
      private System.Windows.Forms.ToolStripTextBox
          courseNumberToolStripTextBox;
      private System.Windows.Forms.ToolStripButton
          fillByCourseNumberToolStripButton;
   }
}
```