Objectives (1 of 2)

- Retrieve and display data from a SQL Server database on Windows forms.
- Use the ADO.NET data components: TableAdapters, DataSets, and DataConnectors.
- Display database fields in a grid and in details view.
Objectives (2 of 2)

- Sort database data using a method and an SQL query.
- Concatenate data fields using SQL or the **DataSet Designer**.
- Declare and instantiate DataSet and TableAdapter objects in code and write the code to bind data to form controls.

VS 2005 and Database Applications

- Professional VB programmers spend the majority of time on applications involving databases.
- The managed providers included in the .NET Framework:
  - The .NET Framework Data Provider for:
    - SQL Server
    - **OLEDB** (i.e., Access)
    - SQL Server Mobile Edition
    - Oracle
    - ODBC (other database formats)
Universal Data Access (UDA)

- Microsoft’s strategy for accessing data for multiple providers
- Goal is to access any type of data from any application on any type of computer

OLEDB

- Technology designed to implement the UDA concept
- Standardized interface that allows reference to data from any source using the same programming tools
- Using OLEDB, programmers don’t need to be concerned with syntax of a particular data format
ADO.NET 2.0 (1 of 2)

- Microsoft’s latest database object model
- Allows VB programmers to use a standard set of objects to refer to data from any source
- .NET approach uses disconnected datasets with common data representation (data types) from multiple sources

ADO.NET (2 of 2)

- .NET framework is integrated with XML (Extensible Markup Language)
  - XML is an industry standard format for storing and transferring data over multiple platforms.
- Multitier applications that use disconnected datasets provide for
  - **Flexibility** → adapt to changes in data or presentation tier
  - **Scalability** → handle increases in the number of users and servers
Accessing Data in the .NET Framework

- Terminology for working with data
  - *Data sources*
  - Data designers
  - Datasets
  - Binding sources
  - Table adapters
  - And more

- Still used are many of the standard terms for data elements
### Databases

A **data table** consists of rows and columns.
- Each **row** represents the data for one item, person or transaction and is called a **record**.
- Each **column** represents a different element of data and is called a **field**.
- A **primary key field** (or combination of fields) uniquely identifies each record.

<table>
<thead>
<tr>
<th>Authors</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Author ID</td>
<td>Last Name</td>
</tr>
<tr>
<td>172-32,1176</td>
<td>White</td>
</tr>
<tr>
<td>213-46,8915</td>
<td>Green</td>
</tr>
<tr>
<td>290-85,7505</td>
<td>Carson</td>
</tr>
<tr>
<td>267-41,2934</td>
<td>Clay</td>
</tr>
<tr>
<td>274-09,9391</td>
<td>Straight</td>
</tr>
<tr>
<td>341-22,7162</td>
<td>Smith</td>
</tr>
<tr>
<td>409-56,7000</td>
<td>Bennett</td>
</tr>
<tr>
<td>423-17,2213</td>
<td>Dall</td>
</tr>
</tbody>
</table>

### Data Sources

- Data Sources can be created from large-scale databases, local databases, files, or other objects.
- A typed **DataSet** is added when a data source is created from a database file.
  - Each typed DataSet has a schema which describes the fields and their data types.
DataSet Objects

- **DataSet Objects** hold a copy of the data in memory
  - A temporary cache rather than the actual database data
- Regardless of the data source, code handles DataSet objects the same
- Dataset objects hold one or more **DataTable** objects
  - A relationship ties the tables together

The DataSet Object Model

![Diagram of the DataSet Object Model]
TableAdapters

- A TableAdapter object provides communication between a program and a database; sometimes called the “data access layer”
- Connects to the database and handles SQL queries and stored procedures, fills the DataSets, and handles writing changes back in the data source
- Can hold multiple queries
- Data is retrieved by calling a specific query or getting all of the data for the table
- Visual designers in the VS IDE automatically generate Table Adapters when a typed DataSet is added

BindingSource Objects

- Facilitates binding the controls on a Windows form to a data source
- Handles the navigation, sorting, filtering and updating of data
- Keeps all of the form’s controls displaying data from the same record
XML Data

- XML is generated automatically
- Data stored in XML is all text, identified by tags, similar to HTML
- XML files can be edited by any text editor
- Tags identify fields by name
- Specs found for XML at http://www.w3.org/XML

DataSet Schema

- Describes the fields, data types, and constraints, such as required fields
- ADO.NET validates the data against the schema and checks for constraint violations
- ADO.NET can treat the XML data as objects
- Data handling in XML and ADO.NET executes faster than in earlier forms of ADO
Visual Studio IDE and Data Sources Window

- IDE provides tools to assist in developing database applications
- Use the **Data Sources window** to create data-bound controls on a form
- Used to add and view the DataSets that are used in a project

Local Database Files

- VS 2005 offers an option to make a copy of the database file inside the project if project is based on a database file
  - This is a duplicate of the original file
- Can be used for development and testing
- Can move the project from one folder or computer to another and the database file is still available in the project
- Can test the application without altering the original data file
Creating a Windows Database Application in a Grid

The DataSet Designer

- After you add a new data source, a file with the extension .xsd is added to the files in the Solution Explorer
- Double-click to open the DataSet Designer
- Used to view and modify the definition of a DataSet
Preview Data

- In the DataSet Designer, right-click the table name and select Preview Data.
- Click the Preview button in the Preview Data dialog box.

Displaying Data in Individual Fields

- In the Data Sources window, select the table name.
- Drop down the list of choices.
- Select Details, rather than the default DataGridView.
Selecting the Control Type for Details View

- Choose the type of control for each database field or change defaults
- Make the selection before dragging the table to the form
- Select a database field, drop down the list for an individual field and make a selection

Populating Combo Boxes with Data

- Set the **DataSource property**
  - Connects to the DataSet
- Set the **DisplayMember property**
  - Selects the field to display in the list
- Automatically fill a list by binding it to a field in a DataSet
- Set the combo box data bindings to not update the DataSet when the combo box selection changes
Adding an Expression to Concatenate Fields

- Used when you need information from a database in a format other than the way it is stored, such as a calculated expression
- Double-click the DataSet’s filename in the Solution Explorer –OR-
- Use the Data Sources window and select **Edit DataSet with Designer**

  - SQL Server uses a plus sign for concatenation rather than the ampersand used by VB and single quotes rather than double quotes for literals
    
    \[ \text{lname} + ', ' + \text{fname} \]

Sorting Data for the ListBox

- Sorting must be done on the data either using the binding source or in the SQL
Modifying the SQL Select Statement

- When a DataSet is created from a database the resulting table data are produced by a query written in SQL.
- Each table in the DataSet has an associated TableAdapter component with a Fill method.
- Fill method retrieves the data from the data source:

  ```csharp
  this.EmployeeTableAdapter.Fill(Me.PubsDataSet.employee)
  ```

- View and modify the SQL statement directly or by using the Query Builder (recommended practice).

Query Builder

- Creates the SQL statement to match the selection criteria that is entered, can be used to test a created SQL statement or add an expression field.
Eliminating Unnecessary SQL Queries

- When a TableAdapter is generated, VS designer automatically generates the SQL SELECT statement for retrieving data.
- The designer also generates SQL INSERT, DELETE, and UPDATE statements.
  - For programs that only display data, the extra statements serve no purpose and should be eliminated.
  - To delete extra code for updates, display the DataSet in the designer and click on the TableAdapter name.
  - Select either Configure from the Data menu or right-click on the Table Adapter and select Configure from the context menu to display the TableAdapter Configuration Wizard.

TableAdapter Configuration Wizard

- Click the Advanced Options button to display the dialog box.
Multiple Tier Projects

- Separate database access from the user interface
- Create a data component as a separate tier

Coding the Form’s Database Objects

- Code must be written to instantiate the data-tier component, retrieve the DataSet, and bind the form’s controls

```csharp
// Create an instance of the data-tier component.
PubsEmployeeData employeeData = New PubsEmployeeData;

// Retrieve the dataset from the data tier.
DataSet aDataSet = new DataSet();
aDataSet = employeeData.getEmployeeDataset();
```
Binding Data Fields to Form Controls

- DataSet is created at run time
- Bind the fields from the DataSet to form controls after retrieving the DataSet
- Best way to bind form controls is to use the BindingSource object

- Declare and instantiate a new BindingSource object in code and set its DataSource and Data Member properties
- If using a stored procedure specify the name of the stored procedure in place of the table name