



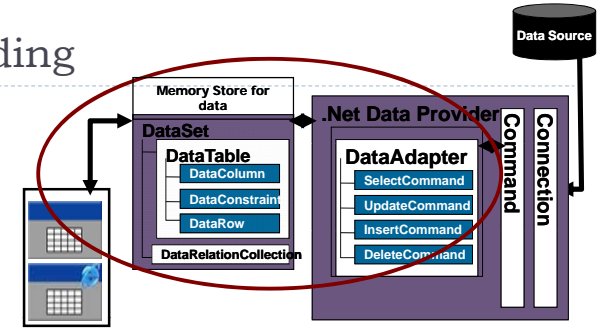
COMPSCI 280 S1 2011 Enterprise Software Development

ADO.NET
TableAdapters & DataSets

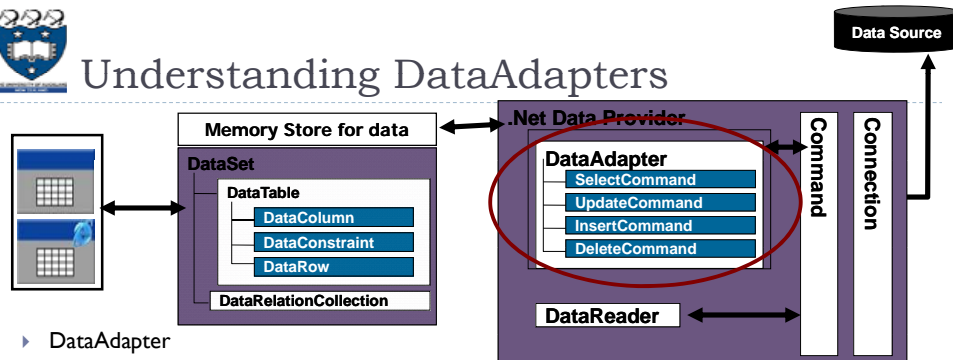


Agenda & Reading

- ▶ **Agenda:**
 - ▶ Understanding DataAdapters
 - ▶ Understanding TableAdapters
 - ▶ Understanding DataSets
 - ▶ Creating TableAdapters
 - ▶ Creating DataSets
 - ▶ Filing DataSets
 - ▶ Binding DataSets
 - ▶ Parameterized Queries
 - ▶ Typed and Untyped DataSets
- ▶ **Recommended Reading:**
 - ▶ Microsoft ADO.NET 2.0 step by step, Rebecca M. Riordan
 - ▶ Chapter 4: Using DataAdapters
- ▶ **Hands-on Lab:**
 - ▶ Lecture21 | Lab: Using DataSets



Understanding DataAdapters



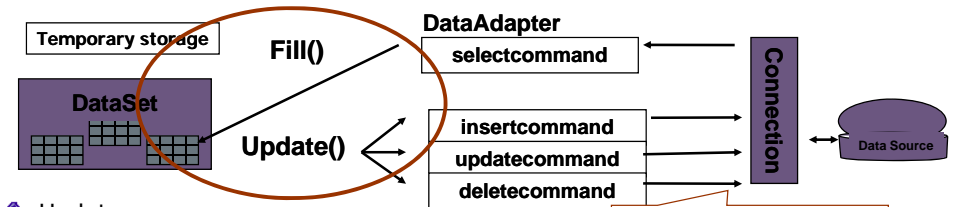
- ▶ **DataAdapter**
 - ▶ is part of a Data Provider
 - ▶ communicate with database in a disconnected environment
 - ▶ manage Data Exchange between the DataSet and the data source
 - ▶ Use a Connection object to communicate with a data source
 - ▶ Use 4 Command objects to read, add, update and delete records in a data source

◆ The .NET Framework Data Provider for OLE DB includes an OleDbDataAdapter object. A DataAdapter is used to retrieve data from a data source to populate tables with a DataSet. The DataAdapter also resolves changes made to the DataSet back to the data source. The DataAdapter uses the Connection object of the .NET framework data provider to connect to a data source and Command objects to retrieve data from and resolve changes to the data source.



Fill/Update

- ▶ **Fill**
 - ▶ Connects to the database
 - ▶ Sends the select command to the database
 - ▶ Gets the result into a DataSet
 - ▶ Disconnects the connection



- ◆ **Update**
 - Reconnects to the database
 - Performs the command to update the data
 - Disconnects the connection

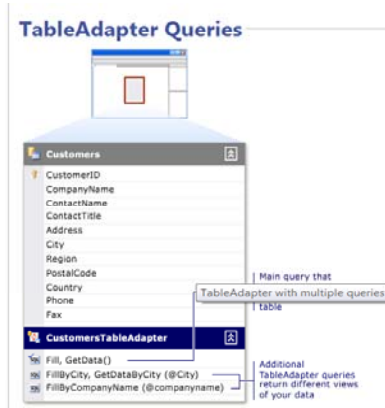
Don't need to do the connecting and disconnecting, it is all done automatically.

- ◆ The SelectCommand property of the DataAdapter is a Command object that retrieves data from the data source. The InsertCommand, UpdateCommand, and DeleteCommand properties of the DataAdapter are Command objects that manage updates to the data in the data source according to the modifications made to the data in the DataSet.
- ◆ The Fill method of the DataAdapter is used to populate a DataSet with the results of the SelectCommand of the DataAdapter.
- ◆ The Update method of the DataAdapter is called to resolve changes from a DataSet back to the data source.

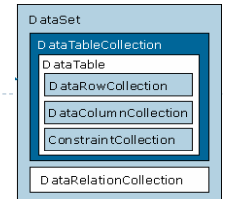
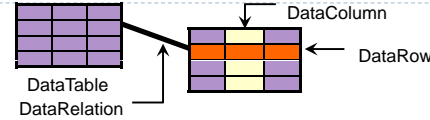


Understanding TableAdapters

- ▶ TableAdapter as a DataAdapter with a built-in connection object
- ▶ It can contain multiple queries.
- ▶ TableAdapters are designer-generated components that improve upon the functionality of DataAdapters.
- ▶ TableAdapters typically contain Fill and Update methods to fetch and update data in a database



Understanding DataSets

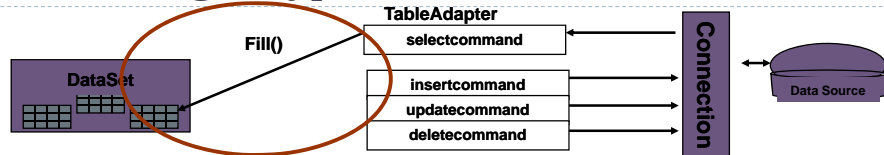


- ▶ DataSet
 - ▶ stores data in a disconnected cache
 - ▶ is an array-like indexing, relational view of Data
 - ▶ is a set of disconnected data.
 - ▶ exposes a hierarchical object model of tables, rows and columns
 - ▶ contains constraints and relationships
 - ▶ typed or untyped

The ADO.NET DataSet is a memory-resident representation of data that provides a consistent relational programming model independent of the data source. The DataSet represents a complete set of data including tables, constraints, and relationships among the tables. Because the DataSet is independent of the data source, a DataSet can include data local to the application, as well as data from multiple data sources. Interaction with existing data sources is controlled through the DataAdapter.



Filling a Typed Dataset



- ▶ Filling a DataSet
 - ▶ Open the Connection to the data source and retrieve the data into a DataSet
- ▶ To fill a DataSet

```
customersTableAdapter.Fill(this.northwindDataSet.Customers);
```

- ▶ To display TableName, ColumnName of the first column and value of the first row and column

```
Console.WriteLine("Table name: " + northwindDataSet.Customers.TableName);
Console.WriteLine("First column: " + northwindDataSet.Customers.CustomerIDColumn.ColumnName);
Console.WriteLine("First value: " + northwindDataSet.Customers.Rows[0][0]);
```

First row

- ▶ The Fill method retrieves rows from the data source using the SELECT statement specified by an associated SelectCommand property. The connection object associated with the SELECT statement must be valid, but it does not need to be open. If the connection is closed before the Fill is called, it is opened to retrieve data, then closed. If the connection is open before Fill is called, it remains open.
- ▶ The Fill operation then adds the rows to destination DataTable objects in the DataSet, creating the DataTable objects if they do not already exist.
- ▶ You can use the Fill method multiple times on the same DataTable. If a primary key exists, incoming rows are merged with matching rows that already exist. If no primary key exists, incoming rows are appended to the DataTable.



Parameterized Queries

- ▶ DataSet with a special type of SQL SELECT (parameterized query)
 - ▶ WHERE clause is incomplete until user enters value for criteria at run time
 - ▶ Uses question mark in place of criteria

SELECT * from Customers

Customers table: Contains 830 rows

```
SELECT Address, City, CompanyName, ContactName, ContactTitle, Country, CustomerID, Fax,
Phone, PostalCode, Region
FROM Customers
WHERE City = ?
```

Customer	Company Name	Contact Name	Contact Title	Address	City
AROU	Around the Horn	Thomas Hardy	Sales Representative	120 Hanover Sq.	London
BSBEV	B's Beverages	Victoria Ashworth	Sales Representative	Fauntleroy Circus	London
CONSH	Consolidated Holdings	Elizabeth Brown	Sales Representative	Berkeley Gardens	London
EASTC	Eastern Connection	Ann Devon	Sales Agent	35 King George	London
NORTS	North/South	Simon Crowther	Sales Associate	South House	London
SEVES	Seven Seas Imports	Hari Kumar	Sales Manager	90 Wadhurst Rd.	London

Customers table: Contains 6 rows only



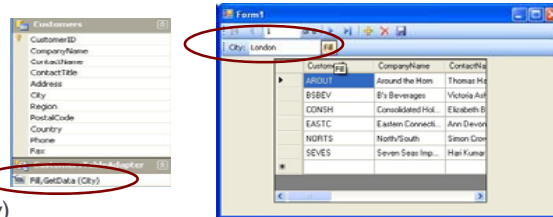
TableAdapter

- To create TableAdapter parameterized Queries
 - Complete the Data Source Configuration Wizard
 - Open the dataset in the Dataset Designer.
 - Right-click the existing TableAdapter and choose Configure
 - Modify the SQL statement by adding a WHERE clause with the desired parameters to the SQL statement. (or by adding "=?" under Filter)
 - Drag the table from the Data Sources Window onto Windows Forms.
 - A ToolStrip control is added to the form that accept any input parameters required by the query.
- (Demo: 21.1_01ParameterizedQuery.wmv)

```
SELECT *
FROM Customers
WHERE (City = ?)
```

- Fill the DataSet
- (Demo: 21.1_02RunQuery.wmv)

```
customersTableAdapter.Fill(northwindDataSet.Customers, CityToolStripTextBox.Text);
```



Multiple queries

- TableAdapters can contain multiple queries to fill their associated data tables.
- To add a query to a TableAdapter in the Dataset Designer
 - Open a dataset in the Dataset Designer.
 - Right-click the desired TableAdapter; and select Add Query.
 - Complete the TableAdapter Query Configuration Wizard.
 - (Demo: 21.2MultipleQueries.wmv)
- To fill the dataset
 - By All customers

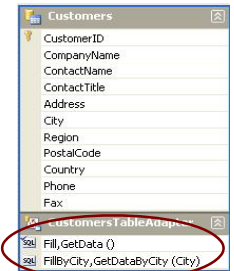
```
CustomersTableAdapter.Fill(northwindDataSet.Customers);
```

- By a parameterized query

```
CustomersTableAdapter.FillByCity(northwindDataSet.Customers, "London");
```

- TableAdapter:

- The number of records = 6 (<> 6, if the ClearBeforeFill property is set to false)



Typed & Untyped DataSets

Typed DataSets

- Use info from an XML schema file to generate a new class
- Derived from DataSet
- Inherit from the base class and assumes the functionality of the base class.
- Advantages
 - Supported by IntelliSense
 - Type checking at compile time
 - Slightly faster than Untyped

```
NorthwindDataSet northwindDataSet = new NorthwindDataSet ();
```

```
northwindDataSet.Customers.CustomerIDColumn
```

```
public partial class northwindDataSet : global::System.Data.DataSet {
    private CustomersDataTable tableCustomers;
```

```
DataSet ds1 = new DataSet();
```

Untyped DataSets

- No xml schema file initially
- Not Inherit from the dataset class
- Contain table(s), columns, rows, etc. but only exposed as collections.
- Advantage:
 - Creating a dataset dynamically

```
ds1.Tables["Customers"].Columns["CustomerID"]
```

- A typed DataSet is a class that derives from a DataSet. As such, it inherits all the methods, events, and properties of a DataSet. Additionally, a typed DataSet provides strongly typed methods, events and properties. This means you can access tables and columns by name, instead of using collection-based methods. Additionally, it provides access to values as the correct type at compile time. Therefore, type mismatch errors are caught when the code is compiled rather than at run time.
- An untyped DataSet is not an instance of a class generated using an XML Schema file. As a consequence, an untyped DataSet has no inherent structure. You create the tables, columns, constraints and relation yourself at design time, at run time in code, or by allowing a DataAdapter's mappings to do so when you use the data adapter to fill the dataset.



Creating Untyped DataSets

Example:DataSetDemo

To create an untyped dataset

Declare objects

- Connection
- DataAdapter
- DataSet
- BindingSource

```
private OleDbConnection cnNorthWind;
private OleDbDataAdapter daCustomers;
private DataSet dsCustomers;
private BindingSource bsCustomers;
```

Create Connection, DataAdapter & DataSet

```
string strConnection = @"File Name=H:\compsci280\conn.udl";
cnNorthWind = new OleDbConnection(strConnection);
String sql = "select * from Customers";
daCustomers = new OleDbDataAdapter(sql, cnNorthWind);
dsCustomers = new DataSet();
```

1 command – only select fields from Customers
Can't update/insert/delete

Untyped DataSet

Fill the DataSet

```
daCustomers.Fill(dsCustomers, "Customers"); //fill first
```

Create a BindingSource

```
bsCustomers = new BindingSource(dsCustomers, "Customers");
```

Associate the Data Binding

```
dgCustomers.DataSource = bsCustomers;
```

- For an untyped DataSet, you must call the Fill method to populate the DataSet first and then bound the DataSet to the DataGridView control.



How to run our examples?

▶ How to run our examples?

- ▶ Database is in "H:\compsci280" folder, OR
 - ▶ Run the project
 - ▶ Click Fill.
- ▶ Database is **NOT** in "H:\compsci280" folder
 - ▶ Method 1: at run time
 - Run the project
 - Click **Reset** to reset the ConnectionString at run time
 - Choose a UDL file to set the ConnectionString

```
customersTableAdapter.Connection.ConnectionString = "File Name=conn.udl";
```

▶ Method 2: at design time

- Select the Project Property
- Select the settings tab
- Change the ConnectionString of the connection object
- Rebuild and run the project ...

Name	Type	Scope	Value
NorthwindC...	(Conne...	Application	Provider=Microsoft.Jet.OLEDB.4.0;Data Source=H:\compsci280\Northwind.mdb