



COMPSCI 280 S1 2011 Enterprise Software Development

ADO.NET
Data Relations



Agenda & Reading

- ▶ **Agenda:**
 - ▶ Relational Databases
 - ▶ Understanding DataRelations
 - ▶ Creating DataRelations
 - ▶ Binding and Filling DataSets
 - ▶ Navigating Related Data Tables
 - ▶ Expression-based Columns
- ▶ **Recommended Reading:**
 - ▶ Microsoft ADO.NET 2.0 step by step, Rebecca M. Riordan
 - ▶ Chapter 6: Modelling a Database by Using DataSets and DataRelations
 - ▶ MSDN Library
 - ▶ <http://msdn.microsoft.com/library/en-us/cpguide/html/cpconaddingrelationshipbetweentwotables.asp>
 - ▶ <http://msdn.microsoft.com/library/en-us/cpguide/html/cpconnavigatingrelationshipbetweentwotables.asp>
 - ▶ <http://msdn.microsoft.com/library/en-us/vbcon/html/vboriDatasetRelations.asp>
- ▶ **Hands-on Lab:**
 - ▶ Lecture25Lab: Using DataRelation to get the corresponding rows from the Orders table

2

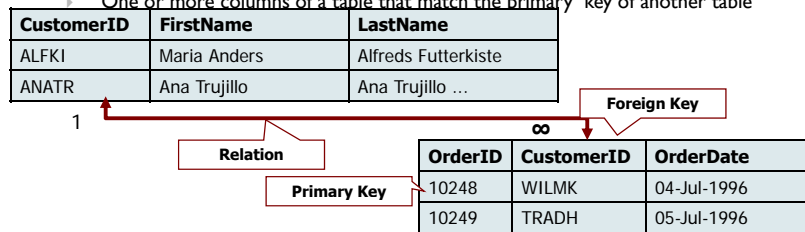
COMPSCI280

Handout25



Relational Databases

- ▶ Tables can be related through primary/foreign key relationships (e.g. A customer may order one or more orders)
 - ▶ **Primary key**
 - ▶ Guarantees the uniqueness of a row
 - ▶ Can be composed of one or more columns
 - ▶ **Foreign key**
 - ▶ Establishes logical relationship between tables
 - ▶ One or more columns of a table that match the primary key of another table



Although a dataset can contain multiple tables and columns as in a database, it does not inherently include a database's ability to relate tables. However, you can create DataRelation objects that establish a relationship between a parent (master) and a child (detail) table based on a common key.

3

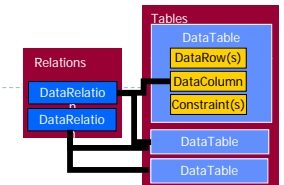
COMPSCI280

Handout25



DataRelations

- ▶ Used to create logical relations between your data
 - ▶ Create relations between two **DataTable** objects
 - ▶ Requires a **DataColumn** object from each **DataTable**
 - ▶ The **Data Type** of both **DataColumns** must be the same
 - ▶ Cannot relate an **Int32 DataColumn** and a **String DataColumn**
- ▶ To create a **DataRelation** at design time
 - ▶ Create a **DataSet**, and 2 **TableAdapters**
 - ▶ northwindDataSet
 - ▶ customersTableAdapter
 - ▶ ordersTableAdapter
 - ▶ Create a **DataRelation** between tables
 - ▶ Edit the xsd file to add a relation between the two tables



- ◆ The **DataRelation** object can make available the records related to a record you are working with. It provides child records if you are in a parent record, and a parent record if you working with a child record.
- ◆ The **DataRelation** object can enforce constraints for referential integrity, such as deleting related child records when you delete a parent record.

4

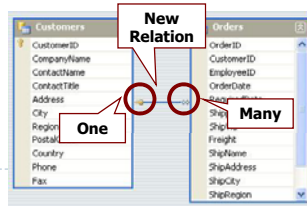
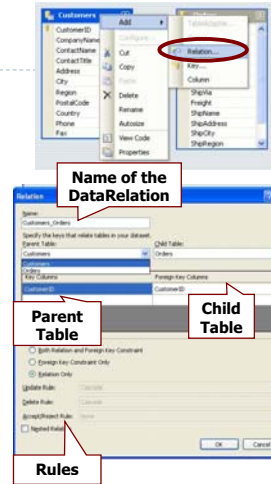
COMPSCI280

Handout25



Creating DataRelations

- Click Add | New Relation
- The EditRelation Dialog box
 - Allows you to create a relation
 - Name** text box
 - defines the relation name
 - Parent element** and **Child element** list box
 - define the two tables that will be related
 - Fields** section contains common fields from master and detail tables
 - Update rule, Delete rule, and Accept/Reject rule** controls what happens when the key of a master record is updated or the master record is deleted
 - Demo: 25.1 Using TableAdapters.wmv



Example: RelationDemo

Creating DataRelations (con't)

- To create a DataRelation at run time
 - Create a connection, 2 DataAdapters and 1 DataSet
 - Fill the parent and child tables in order
 - Relate and create two DataColumn objects and point them to the particular columns in existing Data Tables
 - Create a new DataRelation
 - Use DataColumn objects as parameters when creating the DataRelation
 - Add a DataRelation to the Relations collection of the DataSet

```

OrdersTableAdapter = new NorthwindDataSetTableAdapters.OrdersTableAdapter();
Order_DetailsTableAdapter = new NorthwindDataSetTableAdapters.Order_DetailsTableAdapter();

NorthwindDataSet1 = new NorthwindDataSet();

OrdersTableAdapter.Fill(NorthwindDataSet1.Orders);
Order_DetailsTableAdapter.Fill(NorthwindDataSet1.Order_Details);

DataRelation relation = new DataRelation("OrdersOrderDetails",
    NorthwindDataSet1.Orders.OrderIDColumn, NorthwindDataSet1.Order_Details.OrderIDColumn);
NorthwindDataSet1.Relations.Add(relation);

```

The above code example creates a DataRelation using two DataTable objects in a dataset. Each DataTable contains a column named CustomerID, which serves as a link between the two DataTable objects. The example adds a single DataRelation to the Relations collection of the dataset. The first argument specifies the name of the DataRelation being created. The second argument sets the parent DataColumn and the third argument sets the child DataColumn.



Filling DataSets

- You must fill tables in a specific order
 - Fill the parent table first, then
 - Fill the child table
- Example 1:
 - Parent: Orders; Child: Order Details

```

OrdersTableAdapter.Fill(NorthwindDataSet1.Orders);
Order_DetailsTableAdapter.Fill(NorthwindDataSet1.Order_Details);

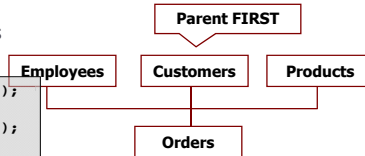
```

- Example 2:
 - Parent tables: Customers, Products & Employees
 - Child: Orders

```

customersTableAdapter.Fill(NorthwindDataSet1.Customers);
ProductsTableAdapter.Fill(NorthwindDataSet1.Products);
EmployeesTableAdapter.Fill(NorthwindDataSet1.Employees);
OrdersTableAdapter.Fill(NorthwindDataSet1.Orders);

```



- Example 3

- A - B - C

```

ATableAdapter.Fill(Ds1, "A");
BTableAdapter.Fill(Ds1, "B");
CTableAdapter.Fill(Ds1, "C");

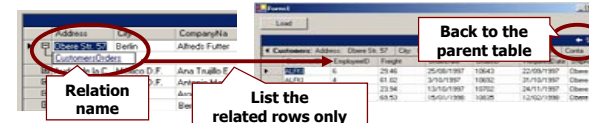
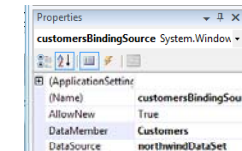
```



Example: OrdersDemo

Binding

- Using a DataGrid Control
 - Used by presentation objects (for example, a DataGrid) to allow easier navigation (for example, "drill down" capability from parent rows to child rows)
 - Create a BindingSource component (parent)
 - DataSource = northwindDataSet
 - DataMember = Customers
 - Create a DataGrid Control and
 - DataSource = CustomersBindingSource



Binding

CustomerID	CompanyName	ContactName	ContactTitle	Address	City
ALFKJ	Alfreds Futterkiste	Maria Anders	Sales Representative	Obere Str. 57	Berlin
ANATR	Ana Traklo Empa	Ana Traklo		Avda. de la Constitución 2312	México D.F.
ANTON	Antonio Moreno				

List the related rows only

Using two DataGridView controls

- Select a row from the parent DataGridView control, the details will be displayed in another DataGridView control

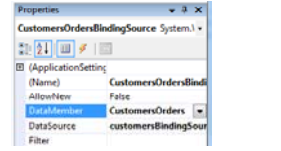
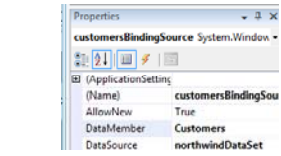
- Create a BindingSource component (parent)

- Create another BindingSource component: (child)
 - CustomersOrdersBindingSource
 - Set the DataSource property to the BindingSource component
 - Set the DataMember property to the relation (CustomersOrders)

Note: It must be the name of relation

- Create a DataGridView control
 - DataSource = CustomersBindingSource

- Create another DataGridView control
 - DataSource = CustomersOrdersBindingSource



Navigating Related DataTables

To get all related child rows

```
DataRowView drv = (DataRowView) OrdersBindingSource.Current;
```

Use the GetChildRows method to retrieve a collection of related rows

- Pass a DataRelation name as the parameter and return an array

```
DataRow[] drChild = drv.Row.GetChildRows("OrdersOrderDetails");
foreach (DataRow dr in drChild)
    txtLog.AppendText(dr["UnitPrice"] + " * " + dr["Quantity"] + Environment.NewLine);
```

The Relation

To get a related parent row

Use the BindingSource to get the current row from the CHILD table

```
DataRowView drv = (DataRowView) OrderDetailsBindingSource.Current;
```

use the GetParentRow method to retrieve the parent row

- Pass a DataRelation name as the parameter
- Return a single DataRow

```
DataRow drParent = drv.Row.GetParentRow("OrdersOrderDetails");
Console.WriteLine(drParent["OrderID"] + Environment.NewLine);
```

- You can use the DataRelation object to get related records. The process is indirect, in that you do not join tables. Instead, you call the GetChildRows method of a data row in the parent table, passing to it the DataRelation object that defines the parent/child relationship. The method returns an array of related child records.
- Similarly, you can get the parent row of a given child record by calling the GetParentRow method of a data row in the child table. In that case, the method does not return an array, but a single data row.

Expression-based Columns

To create an expression-based Column using aggregate function and relations

- Create a new DataColumn
 - Name, Type, Expression: contains either the "Child" or "Parent" keyword
- Add the column to the DataTable

```
DataColumn dcCount = new DataColumn("Count", Type.GetType("System.Int32"),
    "Count(Child.OrderID)");
NorthwindDataSet1.Orders.Columns.Add(dcCount);

DataColumn dcID = new DataColumn("CustomerID", Type.GetType("System.String"),
    "Parent.CustomerID");
NorthwindDataSet1.Order_Details.Columns.Add(dcID);
...
```

Child

Add to the parent table

Parent

Add to the Child table

- Using the Child keyword in a column expression lets you calculate an aggregate function on the children of a parent row. For the Child keyword to work, it is necessary for a relationship to exist between the involved columns.
- The effect of the column definition above is that the SumOffreight column added to the Customers table contains the total of all the freight for a given customer.
- Note: You can reset the Expression property by assigning it a null value or empty string. If a default value is set on the expression column, all previously filled rows are assigned the default value after the Expression property is reset.

More Examples: with Relation

To Parent table:

```
DataColumn dcTotal = new DataColumn("Total", Type.GetType("System.Decimal"),
    "Sum(Child(OrdersOrderDetails).Price)");
NorthwindDataSet1.Orders.Columns.Add(dcTotal);

DataColumn dcAvg = new DataColumn("Avergae", Type.GetType("System.Decimal"),
    "Avg(Child.Price)");
NorthwindDataSet1.Orders.Columns.Add(dcAvg);
```

Name of the relation

To Child table:

```
DataColumn dcPrice = new DataColumn("Price", Type.GetType("System.Decimal"),
    "UnitPrice * Quantity");
NorthwindDataSet1.Order_Details.Columns.Add(dcPrice);
```

Note: The order:

- Fill Dataset -> Add Expression-based DataColumn -> DataBinding (better performance in filling Dataset)
- Add Expression-based DataColumn -> DataBinding -> Fill Dataset (Slow in filling Dataset)
- DataBinding -> Add Expression-based DataColumn -> Fill Dataset (Can't see the all newly added data columns)

- The Child function accepts the name of the DataRelation to get to the child rowset. This argument is optional and only necessary when there is more than one DataRelation indicating a child relation from the source DataTable. So, if DataTable only has a single child DataTable, then the syntax can be simplified to exclude the name of the DataRelation since it is the only DataRelation.