



FPT POLYTECHNIC



LẬP TRÌNH C# 3

BÀI 4: ADO.NET NÂNG CAO

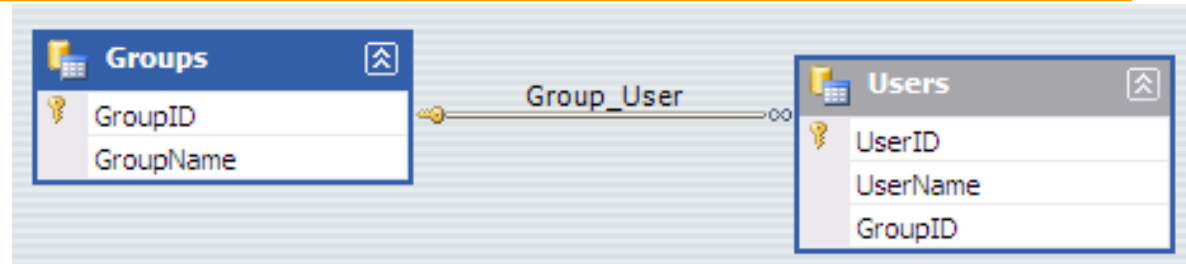
www.poly.edu.vn

- ⊙ DataTable
- ⊙ DataView và ADO.NET



- ❑ DataSet là một đối tượng có thể chứa nhiều DataTable cùng với mối liên hệ giữa chúng (relationship) và kể các ràng buộc (constraint) được lưu hoàn toàn trong bộ nhớ để làm việc offline.
- ❑ Khi bạn thêm các table vào DataSet thì giữa chúng chưa có relation
- ❑ Để tạo ra một relation, bạn sử dụng property **Relations** của DataSet để thêm vào các đối tượng DataRelation

□ Ví dụ có csdl:



□ tạo ra một relation giữa hai table Group, User trong DataSet thông qua cột **GroupID** trong mỗi table với tên relation là **Group_User**

```
DataSet dataSet = LoadData();

DataTable userTable = dataSet.Tables["User"];

DataTable groupTable = dataSet.Tables["Group"];

DataRelation relation=new DataRelation("Group_User",
    groupTable.Columns["GroupID"],
    userTable.Columns["GroupID"]);

dataSet.Relations.Add(relation);
```

- ❑ Phương thức `GetChildRows()`: Sau khi có relation, ta có thể dùng phương thức `instance DataRow.GetChildRows()` để lấy về một mảng các `DataRow` trong bảng con của bảng hiện tại.

```
DataRow[] groupRows = groupTable.Select("GroupID='1'");  
  
DataRow[] memberRows = groupRows[0].GetChildRows("Group_User");  
  
foreach (var row in memberRows)  
    Console.WriteLine(row["UserName"]);
```

- ❖ phương thức `DataTable.Select(string filterExpression)`: trả về mảng `DataRow` có `GroupID = 1`
- ❖ phương thức `GetChildRows(string relationName)`: trả về các phần tử trong mảng `groupRows`

- ❑ Phương thức `GetParentRow()`: Ngược với `GetChildRows()`, phương thức `GetParentRow()` trả về một `DataRow` từ bảng cha của bảng hiện tại dựa vào relation giữa chúng. Ví dụ sau cho thấy `GroupName` của user có `UserID` là "8"

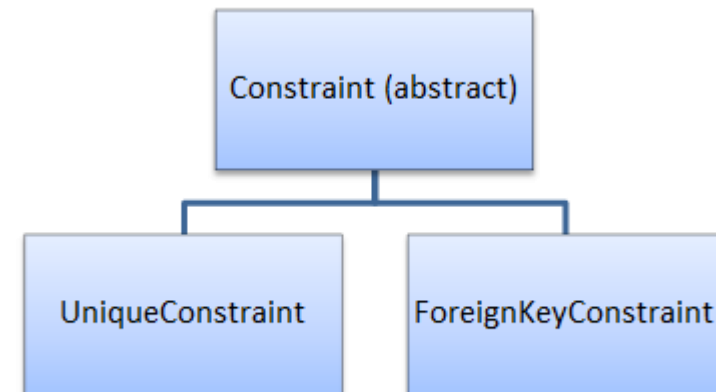
```
DataRow[] childRows = userTable.Select("UserID='8'");  
DataRow parentRow = childRows[0].GetParentRow("Group_User");  
Console.WriteLine(parentRow["GroupName"]);
```

- ❑ DataTable có thể dùng một hoặc nhiều DataColumn để tạo ra một Primary Key
- ❑ Primary Key là định danh phân biệt các DataRow và tránh trùng lặp dữ liệu
- ❑ Dựa vào PrimaryKey, bạn mới có thể dùng phương thức Find() của DataRowCollection.
- ❑ ví dụ tìm và trả về dòng dữ liệu với UserID là "1" trong table User

```
DataColumn[] primaryKeys=new DataColumn[] { table.Columns["UserID"]  
  
table.PrimaryKey = primaryKeys;  
  
DataRow row = table.Rows.Find("1");  
  
Console.WriteLine(row["UserName"]);
```

❑ Constraint là các “luật lệ” mà bạn có thể đặt cho DataColumn nhằm hạn chế và đảm bảo một vài quy tắc nào đó. Có hai loại constraint mà bạn có thể sử dụng:

- ❖ UniqueConstraint: Các giá trị của cột phải là unique (duy nhất).
- ❖ ForeignKeyConstraint: Duy trì liên kết giữa các DataTable trong DataSet
- ❖ Hai lớp này đều thừa kế từ lớp abstract Constraint:



- ❑ Ví dụ tạo constraint cho cột UserID của table User

```
DataTable userTable = dataSet.Tables["User"];  
Constraint constraint=new UniqueConstraint(userTable.Columns["UserID"],true  
userTable.Constraints.Add(constraint);
```

- ❑ Tham số thứ hai của constructor UniqueConstraint chỉ ra cột được sử dụng có phải là primary key không
- ❑ Đặt là true, cột UserID này sẽ trở thành primary key của table này.
- ❑ Có thể đặt thêm UniqueConstraint này cho bất kì cột nào muốn bảo đảm giữ liệu không trùng nhau

- ❑ Constraint này được dùng để tạo relation giữa hai cột thuộc hai table (tạo foreign key cho bảng)
- ❑ ForeignKeyConstraint phải được thêm vào table con vì đây là table chứa foreign key
- ❑ Ví dụ tạo constraint cho cột GroupID của hai table Group và User

```
DataColumn parent = dataSet.Tables["Group"].Columns["GroupID"];
```

```
DataColumn child = dataSet.Tables["User"].Columns["GroupID"];
```

```
ForeignKeyConstraint constraint = new ForeignKeyConstraint("FK_Group_User", parent, child);
```

```
constraint.UpdateRule = Rule.Cascade;
```

```
constraint.DeleteRule = Rule.SetNull;
```

```
dataSet.Tables["User"].Constraints.Add(constraint);
```

THÊM PARAMETER VÀO SQLCOMMAND

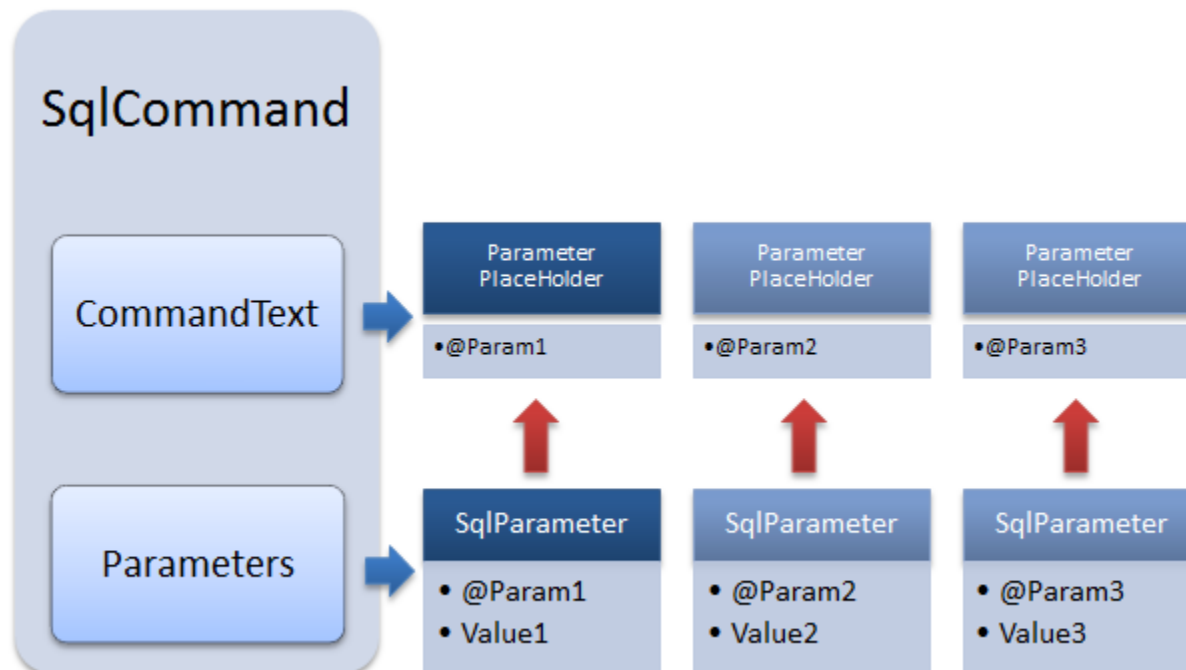
- ❑ Khi bạn làm việc với dữ liệu, bạn sẽ thường xuyên cần lọc kết quả dựa trên một vài điều kiện

```
SqlCommand cmd = new SqlCommand("select * from Customers where city = '" + inputCity + "'");
```

- ❑ Bất kì thứ gì trong TextBox sẽ được đặt vào *inputCity* và được thêm vào câu SQL của bạn
- ❑ Cách làm này có thể bị hacker thay thế chuỗi này bằng những thứ gây hại (SQL Injection)
- ❑ Thay vì tạo một chuỗi động, như bạn thấy ở ví dụ trên, hãy sử dụng parameter

THÊM PARAMETER VÀO SQLCOMMAND

- ❑ Bất kì thứ gì được đặt vào một parameter sẽ được coi là một trường dữ liệu, không phải là một phần của câu lệnh SQL, điều này giúp ứng dụng của bạn an toàn hơn.



(Mô hình kết hợp giữa SqlParameter và SqlCommand)

- ❑ Dùng câu truy vấn với parameter bao gồm ba bước sau:
 - ❖ Tạo một SqlCommand từ một câu lệnh có parameter.
 - ❖ Khai báo một đối tượng SqlParameter, gán giá trị thích hợp cho nó
 - ❖ Gán đối tượng SqlParameter vào property Parameters của đối tượng SqlCommand

THÊM PARAMETER VÀO SQLCOMMAND

```
static void Main()
{
    // conn and reader declared outside try
    // block for visibility in finally block
    SqlConnection conn = null;
    SqlDataReader reader = null;

    string inputCity = "London";

    try
    {
        // instantiate and open connection
        conn = new
            SqlConnection("Server=(local);DataBase=Northwind;Inte
            conn.Open();

        // don't ever do this!
        SqlCommand cmd = new SqlCommand(
            "select * from Customers where city = '" + inputCity

        // 1. declare command object with parameter
        SqlCommand cmd = new SqlCommand(
            "select * from Customers where city = @City", conn);

        // 2. define parameters used in command object
        SqlParameter param = new SqlParameter();
        param.ParameterName = "@City";
        param.Value = inputCity;

        // 3. add new parameter to command object
        cmd.Parameters.Add(param);

        // get data stream
        reader = cmd.ExecuteReader();

        // write each record
        while(reader.Read())
        {
            Console.WriteLine("{0}, {1}",
                reader["CompanyName"],
                reader["ContactName"]);
        }
    }
    finally
    {
        // close reader
        if (reader != null)
        {
            reader.Close();
        }

        // close connection
        if (conn != null)
        {
            conn.Close();
        }
    }
}
```



DEMO

-Hiện thực các ví dụ





LẬP TRÌNH C# 3

BÀI 4: ADO.NET NÂNG CAO(P2)

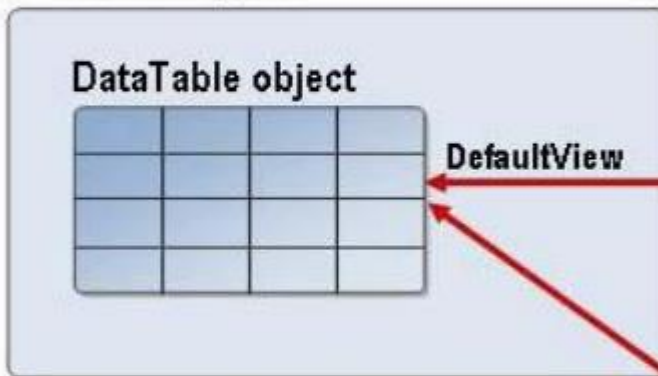
- ❑ Nếu DataTable được dùng lưu trữ dữ liệu thì DataView được dùng hiển thị dữ liệu
- ❑ DataView cho phép lọc và sắp xếp dữ liệu của DataTable
- ❑ Một DataTable có thể có nhiều DataView
- ❑ Một DataTable luôn có một Default View

❑ Mô hình Dataview

A DataView provides a sorted or filtered view of data in a DataTable object

DataSet object

DataTable object



DefaultView

DataView object

Default sort / filter
criteria

DataView objects

Additional views

Alternative sort / filter
criteria

□ Khai báo và khởi tạo

‘Khai báo và khởi tạo DataView

```
DataView dv = new DataView(bang);
```

‘Tham chiếu đến DataView mặc định

```
DataView dv = bang.DefaultView;
```

□ Sử dụng DataView

- ❖ Để sắp xếp dữ liệu thuộc tính Sort
- ❖ Để lọc dữ liệu theo điều kiện thuộc tính RowFilter
- ❖ Để lọc dữ liệu theo trạng thái dòng thuộc tính RowStateFilter
- ❖ Để tìm kiếm phương thức Find và FindRows

❑ Ví dụ tạo và hiển thị dữ liệu lên Dataview

```
private void btnDisplay_Click(object sender, EventArgs e)
{
    string ConString = @"Data Source=.\SQLEXPRESS;Initial Catalog=Northwind;Integrated Security=True";
    string Query = "SELECT * FROM Items";

    SqlDataAdapter adapter = new SqlDataAdapter(Query, ConString);
    DataSet set = new DataSet();

    adapter.Fill(set, "Items");
    DataView dv = set.Tables["Items"].DefaultView;
    dataGridView1.DataSource = dv;
}
```

The screenshot shows a Windows Form titled 'Form1'. It contains a DataGridView with the following data:

	ID	Name	Price	Date
▶	1	LED Screen	\$120	27-01-2017
	2	New Printer	\$20	25-05-2017
	3	4GB DDR3 RAM	\$50	26-05-2017
*				

Below the DataGridView is a scroll bar. To the right of the DataGridView is a panel containing five buttons: 'Display DataView' (highlighted with a green border), 'Sort DataView', 'Add New Row', 'Edit Row', and 'Delete Row'.

❑ Vi dụ Sorting and Filtering DataView

```
private void btnSort_Click(object sender, EventArgs e)
{
    string ConString = @"Data Source=.\SQLEXPRESS;Initial Catalog=Northwind;Integrated Security=True";
    string Query = "SELECT * FROM Items";

    SqlDataAdapter adapter = new SqlDataAdapter(Query, ConString);
    DataSet set = new DataSet();

    adapter.Fill(set, "Items");
    DataView dv = set.Tables["Items"].DefaultView;
    dv.Sort = "Name ASC";
    dataGridView1.DataSource = dv;
}
```

The screenshot shows a Windows Form titled 'Form1' with a DataView control. The DataView displays a table with the following data:

ID	Name	Price	Date
3	4GB DDR3 RAM	\$50	26-05-2017
1	LED Screen	\$120	27-01-2017
2	New Printer	\$20	25-05-2017

The 'Sort DataView' button is highlighted with a green rectangle. Other buttons visible include 'Display DataView', 'Add New Row', 'Edit Row', and 'Delete Row'.

❑ Vi dụ thêm Row vào DataView

```
private void btnAdd_Click(object sender, EventArgs e)
{
    string ConString = @"Data Source=.\SQLEXPRESS;Initial Catalog=Northwind;Integrated Security=True";
    string Query = "SELECT * FROM Items";

    SqlDataAdapter adapter = new SqlDataAdapter(Query, ConString);
    DataSet set = new DataSet();

    adapter.Fill(set, "Items");
    DataView dv = set.Tables["Items"].DefaultView;
    dv.AllowNew = true;
    DataRowView newRow = dv.AddNew();
    newRow.BeginEdit();
    newRow["Name"] = "Router";
    newRow["Price"] = "$130";
    newRow["Date"] = "26 August 2016";
    newRow.EndEdit();
    dataGridView1.DataSource = dv;
}
```

ID	Name	Price	Date
1	LED Screen	\$120	27-01-2017
2	New Printer	\$20	25-05-2017
3	4GB DDR3 RAM	\$50	26-05-2017
	Router	\$130	26-08-2016

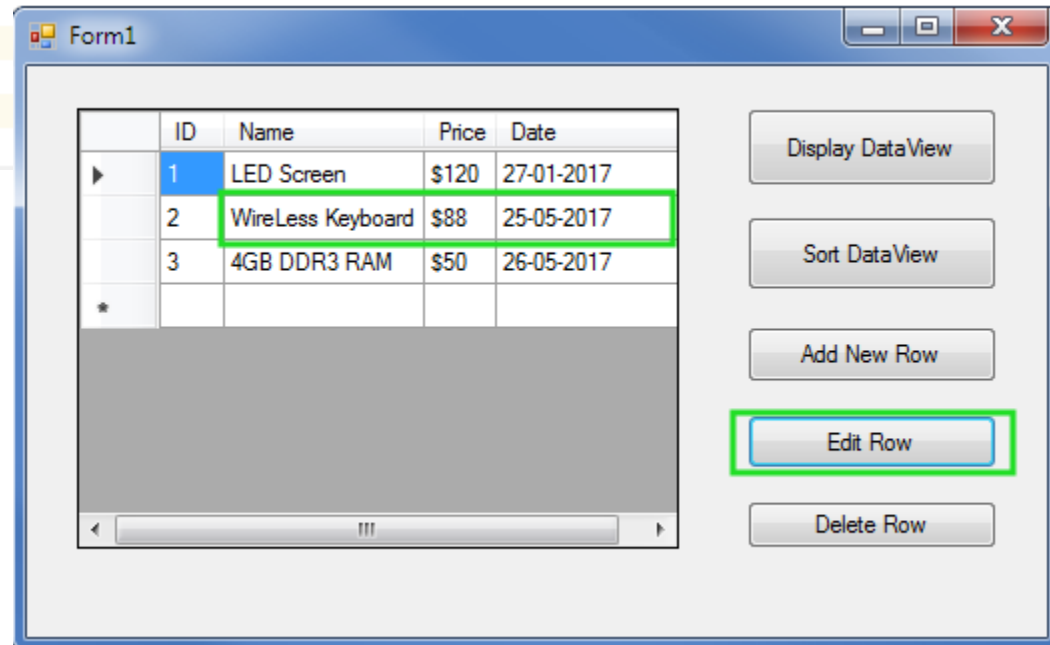
❑ Vi dụ Editing or Updating Row in DataView

```
private void btnEdit_Click(object sender, EventArgs e)
{
    string ConString = @"Data Source=.\SQLEXPRESS;Initial Catalog=Northwind;Integrated Security=True";
    string Query = "SELECT * FROM Items";

    SqlDataAdapter adapter = new SqlDataAdapter(Query, ConString);
    DataSet set = new DataSet();

    adapter.Fill(set, "Items");
    DataView dv = set.Tables["Items"].DefaultView;
    dv.AllowEdit = true;
    dv[1].BeginEdit();
    dv[1]["Name"] = "WireLess Keyboard";
    dv[1]["Price"] = "$88";
    dv[1].EndEdit();

    dataGridView1.DataSource = dv;
}
```



	ID	Name	Price	Date
▶	1	LED Screen	\$120	27-01-2017
	2	WireLess Keyboard	\$88	25-05-2017
	3	4GB DDR3 RAM	\$50	26-05-2017
*				

Buttons on the right:

- Display DataView
- Sort DataView
- Add New Row
- Edit Row** (highlighted in green)
- Delete Row

❑ Ví dụ Deleting Row from DataView

```
private void btnDelete_Click(object sender, EventArgs e)
{
    string ConString = @"Data Source=.\SQLEXPRESS;Initial Catalog=Northwind;Integrated Security=True";
    string Query = "SELECT * FROM Items";

    SqlDataAdapter adapter = new SqlDataAdapter(Query, ConString);
    DataSet set = new DataSet();

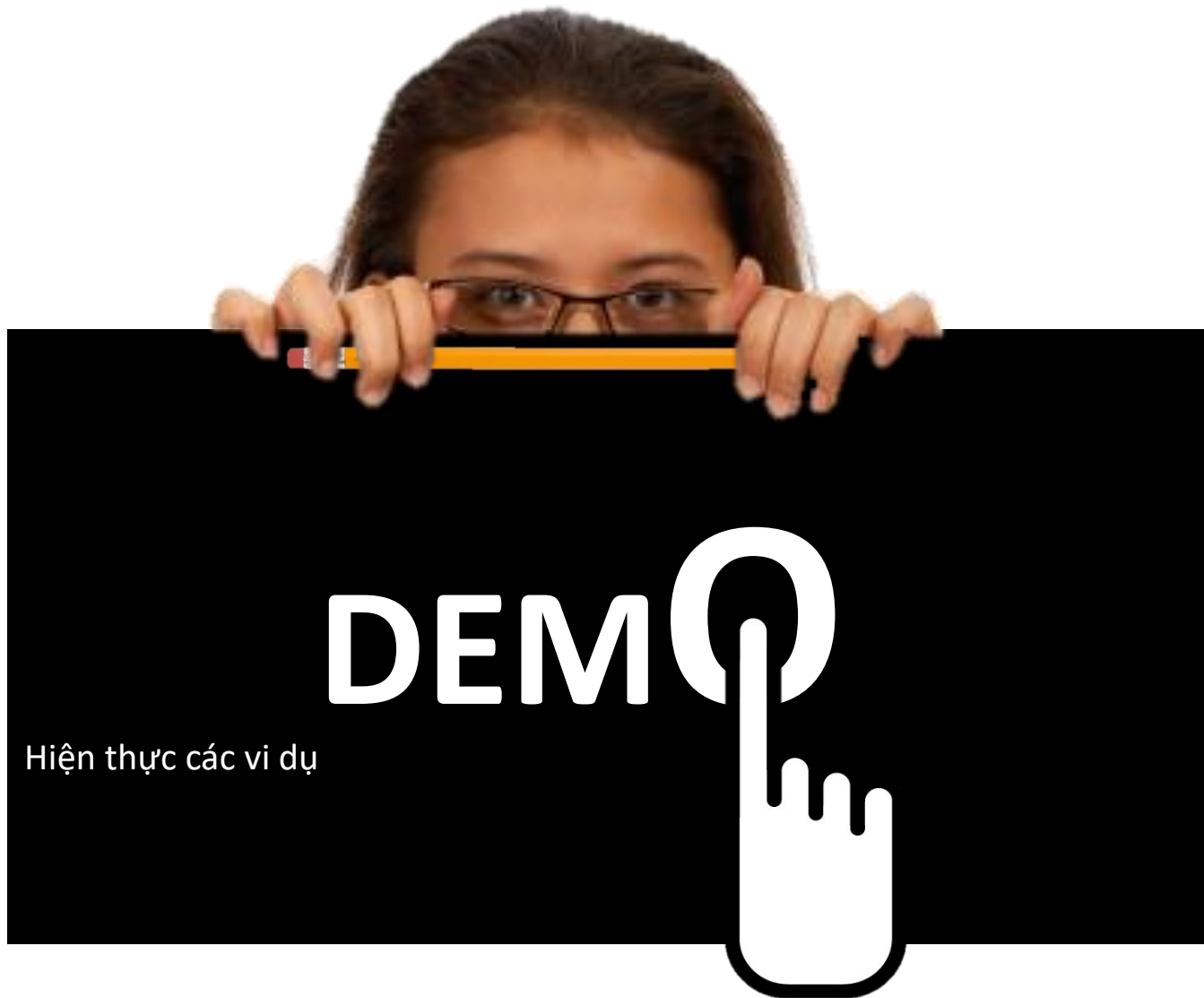
    adapter.Fill(set, "Items");
    DataView dv = set.Tables["Items"].DefaultView;
    dv.AllowDelete = true;
    dv.Table.Rows[2].Delete();

    dataGridView1.DataSource = dv;
}
```

Form1

	ID	Name	Price	Date
▶	1	LED Screen	\$120	27-01-2017
	2	New Printer	\$20	25-05-2017
*				

Buttons: Display DataView, Sort DataView, Add New Row, Edit Row, Delete Row



Tổng kết bài học

◎ DataTable

◎ DataView và ADO.NET





KẾT THÚC