BÀI THỰC HÀNH SỐ 3 (4 tiết)

THỰC THI TRUY VẤN CÓ DÙNG THAM SỐ

I. Mục tiêu

Bài thực hành này giúp sinh viên tìm hiểu:

- Cách truyền tham số vào một lệnh truy vấn SQL
- Cách thực thi các lệnh SELECT, INSERT, UPDATE, DELETE có dùng tham số.
- Gọi và thực thi các Stored Procedure

Sau bài thực hành này, sinh viên cần nắm rõ những vấn đề sau:

- Sử dụng đối tượng Parameter để truyền tham số vào các lệnh.
- Cách thực thi và nhận kết quả trả về từ các lệnh truy vấn có tham số
- Cách thực thi các thủ tục và nhận dữ liệu trả về.
- Cách xây dựng ứng dụng trên nền Windows Form.

II. Nội dung lý thuyết

Để thực thi một lệnh có chứa tham số, thực hiện các bước sau:

B1. Tạo một đối tượng Command chứa lệnh SQL với các điểm đánh dấu tham số (parameter placeholder). Điểm đánh dấu tham số là tên đại diện cho tham số, chỉ ra vị trí mà một tham số sẽ được cung cấp, bắt đầu bởi dấu @.
B2. Thêm các tham số vào đối tượng Command.
B3. Gán giá trị cho các tham số.
B4. Thực thi lệnh.

1. Đưa các tham số vào đối tượng Command

Để đưa một tham số vào đối tượng Command, ta sử dụng phương thức Add của thuộc tính Parameters của đối tượng Command theo một trong 5 cách sau đây:

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

hoặc

```
public SqlParameter AddWithValue(string parameterName, object value)
```

Trong đó:

- `SqlParameter value`: là một đối tượng thuộc kiểu SqlParameter được khởi tạo trước
- `string parameterName`: là tên của tham số

- `object value`: là giá trị được truyền cho tham số
- `SqlDbType sqlDbType`: là kiểu dữ liệu của tham số
- `int size`: là kích thước tối đa của dữ liệu mà tham số có thể nhận
- `string sourceColumn`: là tên của cột sẽ nhận giá trị từ tham số

Ví dụ:

```
SqlCommand mySqlCommand = mySqlConnection.CreateCommand();

mySqlCommand.CommandText =
"INSERT INTO Customers ( CustomerID, CompanyName, ContactName ) " +
"VALUES ( @CustomerID, @CompanyName, @ContactName )";
```

Đối tượng Command này chứa 3 tham số là `@CustomerID, @CompanyName, @ContactName`. Để truyền tham số vào đối tượng Command, ta viết như sau:

```
mySqlCommand.Parameters.Add("@CustomerID", SqlDbType.NChar, 5);
mySqlCommand.Parameters.Add("@CompanyName", SqlDbType.NVarChar, 40);
mySqlCommand.Parameters.Add("@ContactName", SqlDbType.NVarChar, 30);
```

2. *Truyền giá trị cho các tham số*

Giá trị của mỗi tham số được gán qua thuộc tính ***Value***. Để truy xuất đến một tham số trong Command, ta đặt tên tham số trong cặp dấu móc vuông [ ] ngay sau thuộc tính Parameters của đối tượng Command.

Ví dụ:

```
mySqlCommand.Parameters["@CustomerID"].Value = "J4COM";
mySqlCommand.Parameters["@CompanyName"].Value = "J4 Company";
mySqlCommand.Parameters["@ContactName"].Value = "Jason Price";
```

hoặc cũng có thể gán giá trị ngay khi thêm tham số như sau

```
mySqlCommand.Parameters.Add("@CustomerID",
      SqlDbType.NChar, 5).Value = "J4COM";
```

Để gán giá trị ***null***, ta dùng một trong 2 cách sau:

```
mySqlCommand.Parameters["@ContactName"].IsNullable = true;
```

hoặc

```
mySqlCommand.Parameters["@ContactName"].Value = DBNull.Value;
```

Nếu một tham số có thể nhận giá trị trả về sau khi thực thi truy vấn, ta phải thêm đoạn mã sau:

```
mySqlCommand.Parameters["@Tên_Tham_Số"].Direction =
  ParameterDirection.Output;
```

3. *Thực thi một Stored Procedure*

Có 2 cách để thực thi một lời gọi thủ tục: dùng lệnh T-SQL EXECUTE hoặc thiết lập giá trị thuộc tính CommandType của đối tượng Command là CommandType.StoredProcedure.

Cách 1

```
SqlCommand mySqlCommand = mySqlConnection.CreateCommand();

mySqlCommand.CommandText =
  "EXECUTE Tên_Thủ_Tục @Tham_Số_Ra OUTPUT, @Tham_Số_1, " +
  "@Tham_Số_2, @Tham_Số_3, @ Tham_Số_4, ...";

mySqlCommand.ExecuteNonQuery();
// hoặc mySqlCommand.ExecuteScalar();
// hoặc mySqlCommand.ExecuteReader();
```

Với cách này, ta có thể nhận dữ liệu trả về bởi lệnh RETURN trong thủ tục. Ví dụ:

```
mySqlCommand.CommandText =
  "EXECUTE @Tham_Số_Ra = Tên_Thủ_Tục @Tham_Số_1, " +
  "@Tham_Số_2, @Tham_Số_3, @ Tham_Số_4, ...";
```

Cách 2

```
mySqlCommand.CommandText =  "Tên_Thủ_Tục";
mySqlCommand.CommandType = CommandType.StoredProcedure;
```

4. Nhận giá trị trả về từ tham số

a. Lấy dữ liệu trả về từ tham số dùng từ khóa OUTPUT

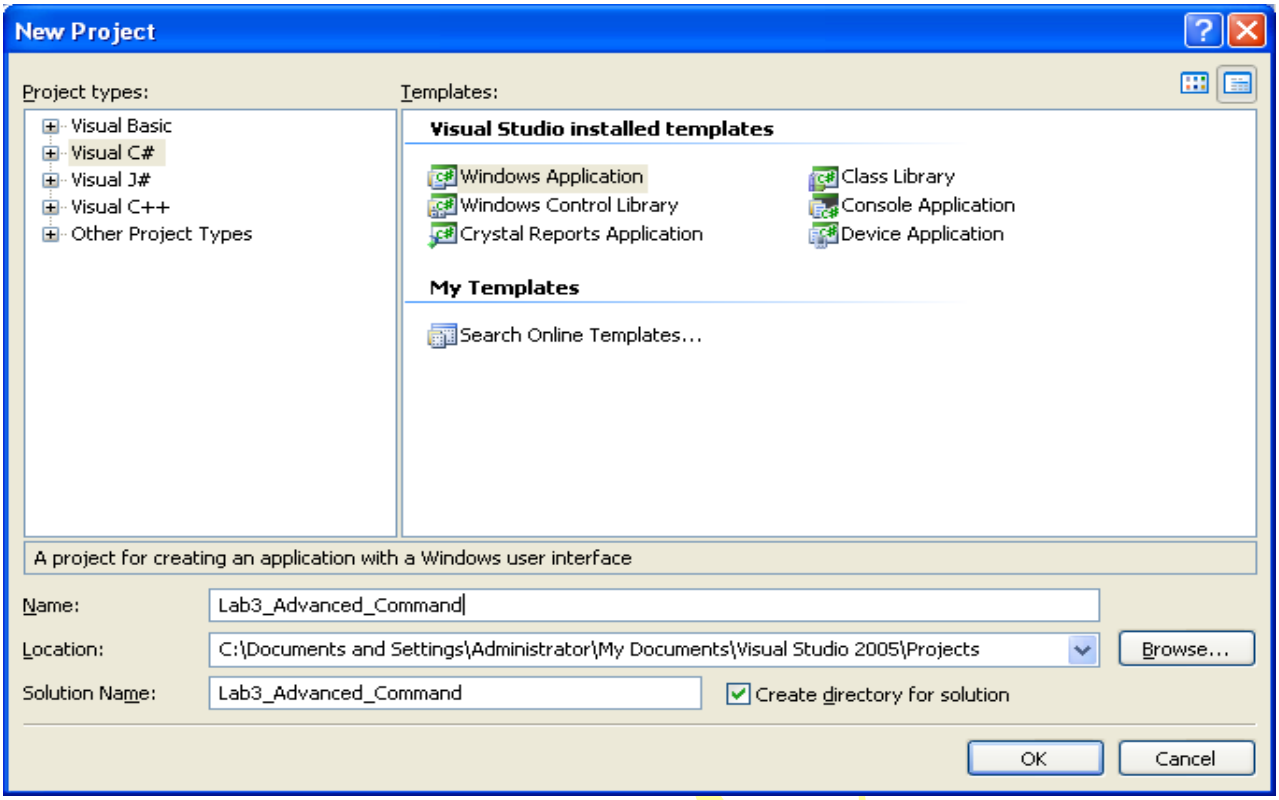
```
mySqlCommand.CommandText =
   "EXECUTE Tên_Thủ_Tục @Tham_Số_Ra OUTPUT, @Tham_Số_1, " +
   "@Tham_Số_2, @Tham_Số_3, @ Tham_Số_4, ...";

mySqlCommand.Parameters.Add("@Tham_Số_Ra", SqlDbType.Int);
mySqlCommand.Parameters["@Tham_Số_Ra "].Direction =
  ParameterDirection.Output;

mySqlCommand.ExecuteNonQuery();
object returnedValue = mySqlCommand.Parameters["@Tham_Số_Ra"].Value;
```

b. Lấy dữ liệu trả về bởi lệnh RETURN trong thủ tục

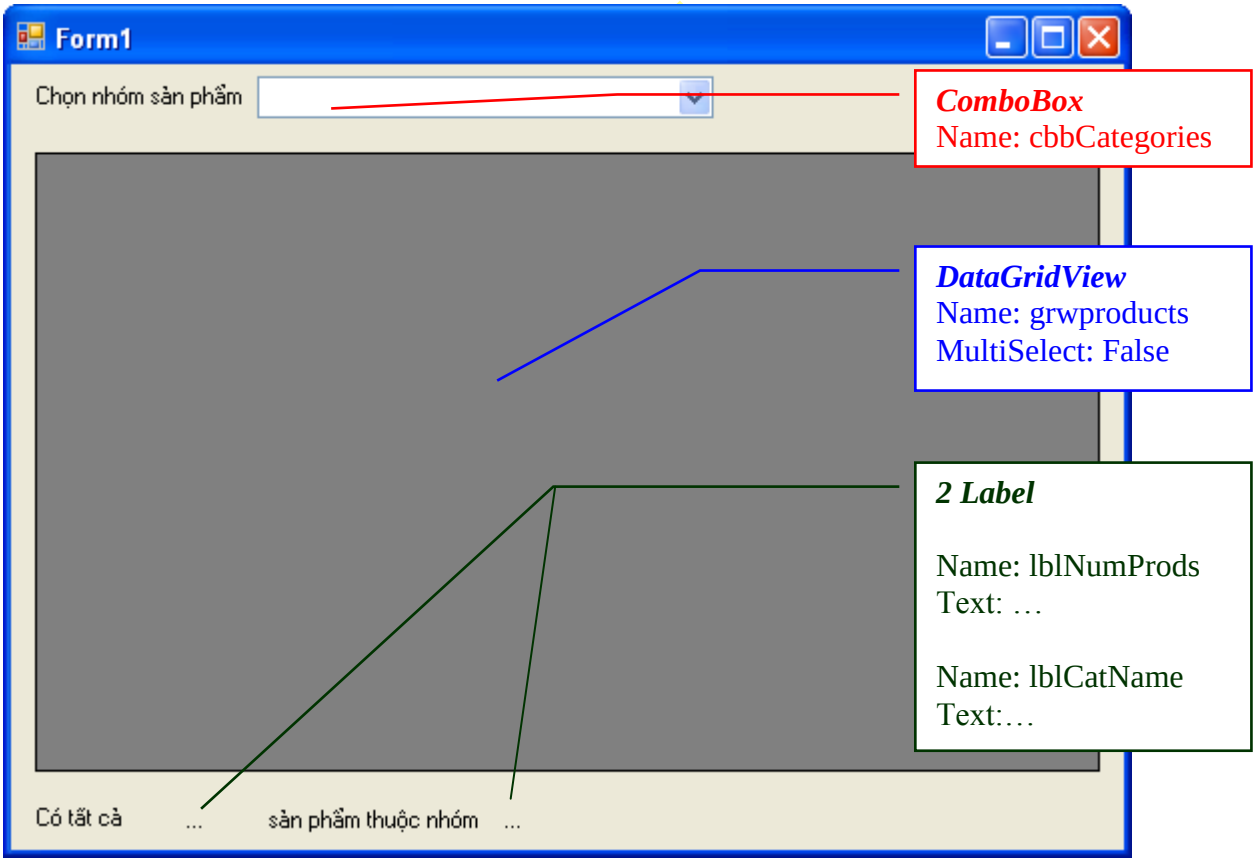
```
mySqlCommand.CommandText =
   "EXECUTE @Tham_Số_Ra = Tên_Thủ_Tục @Tham_Số_1, " +
   "@Tham_Số_2, @Tham_Số_3, @ Tham_Số_4, ...";

mySqlCommand.Parameters.Add("@Tham_Số_Ra", SqlDbType.Int);
mySqlCommand.Parameters["@Tham_Số_Ra "].Direction =
  ParameterDirection.Output;

mySqlCommand.ExecuteNonQuery();
object returnedValue = mySqlCommand.Parameters["@Tham_Số_Ra"].Value;
```

III. Hướng dẫn thực hành

Tạo một dự án Windows Application mới, đặt tên là Lab3_Advanced_Command



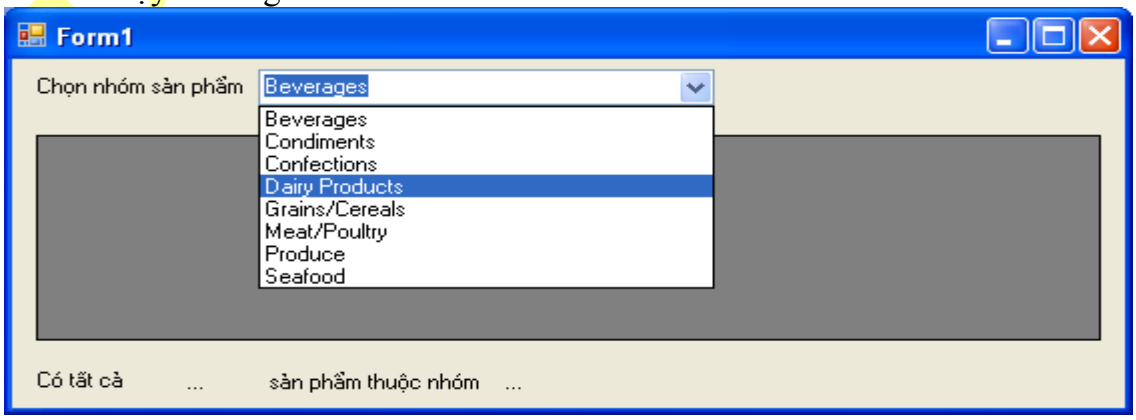
Nhấp đôi chuột vào Form1.cs và thiết kế Form có dạng như sau:



Nhấp đôi chuột lên Form1 để tạo phương thức xử lý sự kiện Form1_Load. Tạo hàm để tải danh sách nhóm sản phẩm lên ComboBox và bổ sung đoạn mã sau để xử lý sự kiện Form1_Load.

```
private void LoadCategories()
{
    string connectionString =
        "server=localhost;database=Northwind;uid=sa;pwd=sa;";
    SqlConnection conn = new SqlConnection(connectionString);

    SqlCommand cmd = conn.CreateCommand();
    cmd.CommandText =
        "SELECT CategoryID, CategoryName FROM Categories";

    SqlDataAdapter adapter = new SqlDataAdapter(cmd);
    DataTable dt = new DataTable();

    // mở kết nối csdl
    conn.Open();

    // Lấy dữ liệu từ csdl đưa vào DataTable
    adapter.Fill(dt);

    // đóng kết nói và giải phóng tài nguyên
    conn.Close();
    conn.Dispose();

    // Đưa dữ liệu vào ComboBox
    cbbCategories.DataSource = dt;

    // Hiển thị tên nhóm sản phẩm
    cbbCategories.DisplayMember = "CategoryName";

    // Nhưng khi lấy giá trị thì lấy ID của nhóm
    cbbCategories.ValueMember = "CategoryID";
}

private void Form1_Load(object sender, EventArgs e)
{
    this.LoadCategories();
}
```

Nhấn F5 để chạy chương trình

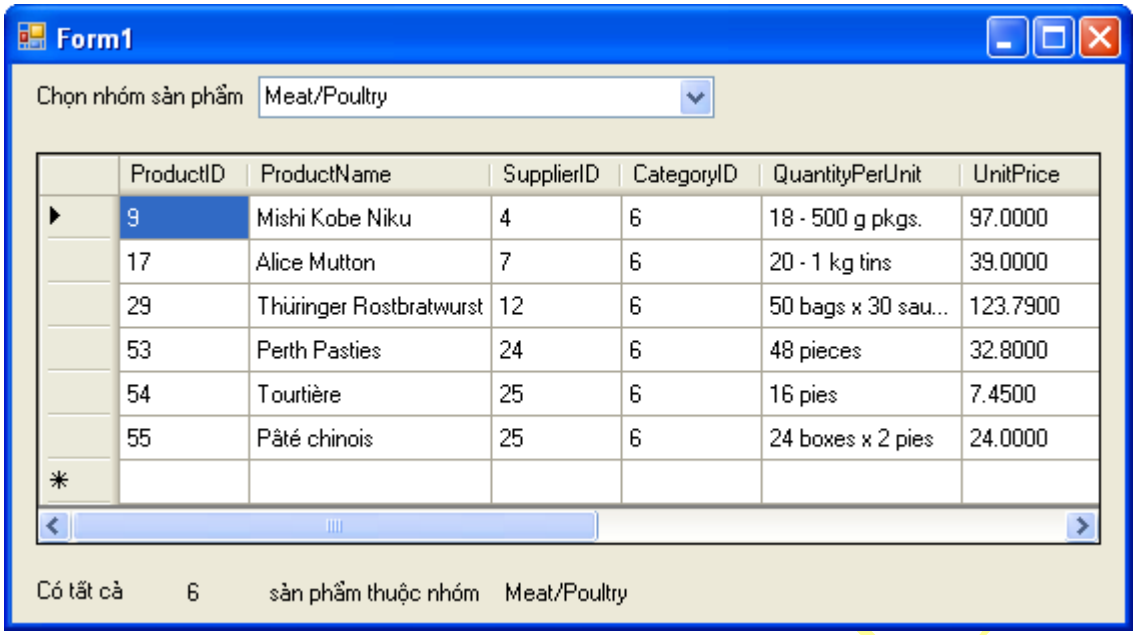


1. *Truyền tham số vào đối tượng Command*

Nhắp phải vào ComboBox, chọn Properties. Trong khung Properties, nhấn nút Events, nhấp đôi chuột vào SelectedIndexChanged. Bổ sung đoạn mã sau vào phương thức cbbCategories_SelectedIndexChanged.

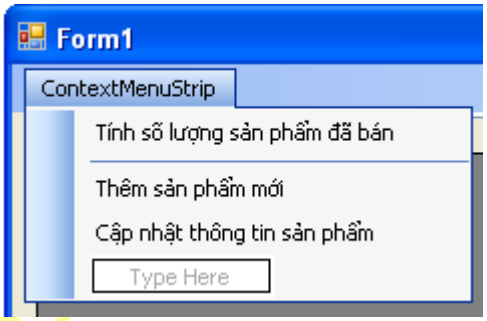
```
private void cbbCategories_SelectedIndexChanged(object sender, EventArgs e)
{
    if (cbbCategories.SelectedIndex == -1) return;

    string connectionString =
        "server=localhost;database=Northwind;uid=sa;pwd=sa;";
    SqlConnection conn = new SqlConnection(connectionString);

    SqlCommand cmd = conn.CreateCommand();
    cmd.CommandText =
        "SELECT * FROM Products WHERE CategoryID = @categoryId";

    // Truyền tham số
    cmd.Parameters.Add("@categoryId", SqlDbType.Int);

    if (cbbCategories.SelectedValue is DataRowView)
    {
        DataRowView rowView = cbbCategories.SelectedValue as DataRowView;
        cmd.Parameters["@categoryId"].Value = rowView["CategoryID"];
    }
    else
        cmd.Parameters["@categoryId"].Value = cbbCategories.SelectedValue;

    // Tạo bộ điều phối dữ liệu
    SqlDataAdapter adapter = new SqlDataAdapter(cmd);
    DataTable dt = new DataTable();

    // mở kết nối csdl
    conn.Open();

    // Lấy dữ liệu từ csdl đưa vào DataTable
    adapter.Fill(dt);

    // đóng kết nói và giải phóng tài nguyên
    conn.Close();
    conn.Dispose();

    // Đưa dữ liệu vào DataGridView
    grwProducts.DataSource = dt;

    // Tính số lượng mẫu tin
    lblNumProds.Text = dt.Rows.Count.ToString();
    lblCatNames.Text = cbbCategories.Text;
}
```

Nhấn phím F5 để chạy chương trình. Nhấp chuột vào ComboBox, chọn nhóm sản phẩm để xem danh sách sản phẩm thuộc nhóm đó.



2. *Nhận giá trị trả về từ tham số*

Từ thanh Toolboxs, kéo một đối tượng ContextmenuStrip vào Form1. Thiết kế menu có dạng sau (tên của các menu lần lượt là tsmCalculateQuantity, tsmSeperator, tsmAddProducts, tsmUpdateProduct):



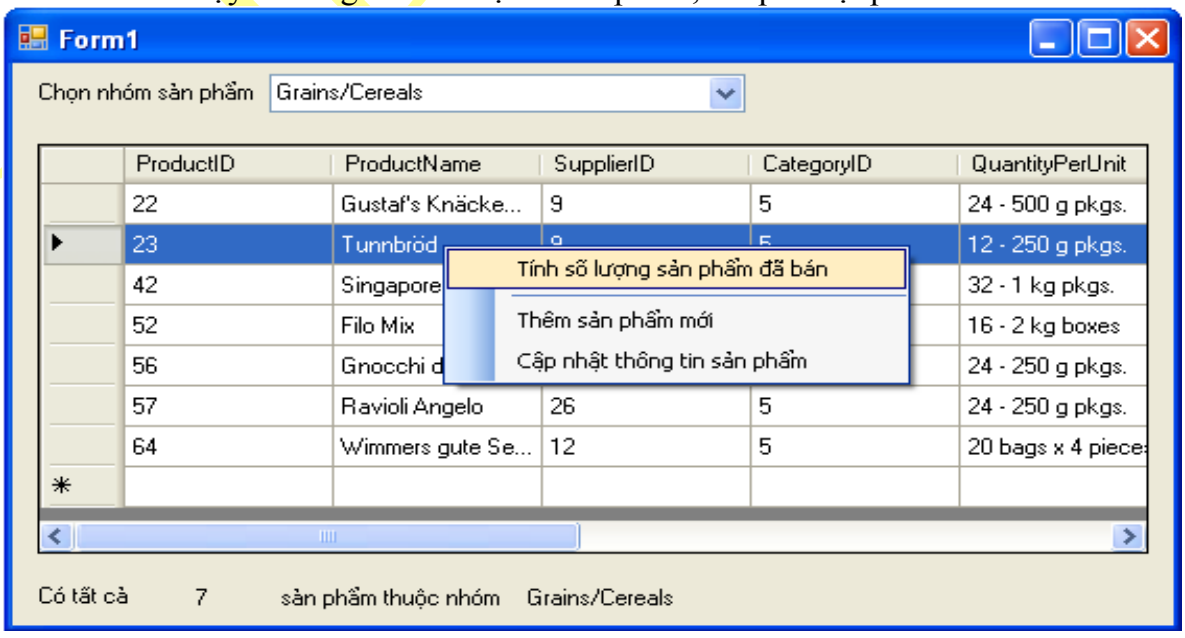
Nhắp phải chuột vào DataGridView, chọn Properties, thiết lập thuộc tính ContextMenuStrip của DataGridView là contextMenuStrip1.

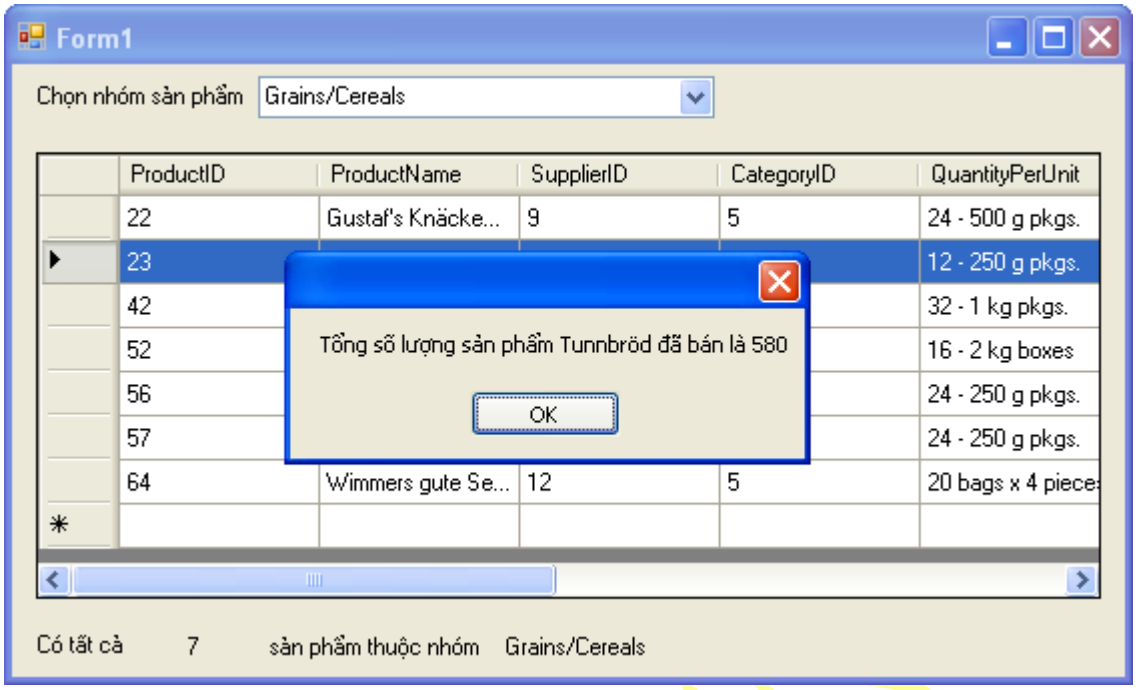
Nhắp đôi chuột lên menu "*Tính số lượng sản phẩm đã bán*" để tạo phương thức xử lý sự kiện Click cho menu. Bổ sung đoạn mã sau:

```
private void tsmCalculateQuantity_Click(object sender, EventArgs e)
{
    string connectionString =
        "server=localhost;database=Northwind;uid=sa;pwd=sa;";
    SqlConnection conn = new SqlConnection(connectionString);

    SqlCommand cmd = conn.CreateCommand();
    cmd.CommandText =
        "SELECT @numSaleProducts = sum(Quantity) FROM [Order Details] " +
        "WHERE ProductID = @productId";
```

```
        // Lấy thông tin sản phẩm được chọn
        if (grwProducts.SelectedRows.Count > 0)
        {
            DataGridViewRow selectedRow =
                grwProducts.SelectedRows[0];

            DataRowView rowView = selectedRow.DataBoundItem as DataRowView;

            // Truyền tham số
            cmd.Parameters.Add("@productId", SqlDbType.Int);
            cmd.Parameters["@productId"].Value =
                                    rowView["ProductID"];

            cmd.Parameters.Add("@numSaleProducts", SqlDbType.Int);
            cmd.Parameters["@numSaleProducts"].Direction =
                                        ParameterDirection.Output;

            // mở kết nối csdl
            conn.Open();

            // thực thi truy vấn và lấy dữ liệu từ tham số
            cmd.ExecuteNonQuery();

            string result = cmd.Parameters["@numSaleProducts"].Value.ToString();
            MessageBox.Show("Tổng số lượng sản phẩm " + rowView["ProductName"] +
                " đã bán là " + result);

            // đóng kết nói và giải phóng tài nguyên
            conn.Close();
        }

        cmd.Dispose();
        conn.Dispose();
    }
```

Nhấn nút F5 để chạy chương trình. Chọn 1 sản phẩm, nhắp chuột phải.





Trở lại Form1, nhấp đôi chuột vào 2 menu còn lại trong contextMenu1 để tạo phương thức xử lý sự kiện Click của chúng như sau:

```
private void tsmAddProduct_Click(object sender, EventArgs e)
{
}

private void tsmUpdateProducts_Click(object sender, EventArgs e)
{
}
```

3. *Thực thi lệnh bằng cách Sử dụng Stored Procedure*

Tạo hai Stored Procedure sau để thực hiện thêm một sản phẩm mới và cập nhật thông tin sản phẩm

```
CREATE PROCEDURE [InsertProducts]
        (@ProductID      [int] OUTPUT,
         @ProductName    [nvarchar](40),
         @SupplierID     [int],
         @CategoryID     [int],
         @QuantityPerUnit [nvarchar](20),
         @UnitPrice      [money],
         @UnitsInStock   [smallint],
         @UnitsOnOrder   [smallint],
         @ReorderLevel   [smallint],
         @Discontinued   [bit])

AS
INSERT INTO [Northwind].[dbo].[Products]
          ( [ProductName],
          [SupplierID], [CategoryID],
          [QuantityPerUnit], [UnitPrice],
          [UnitsInStock], [UnitsOnOrder],
          [ReorderLevel], [Discontinued])
```

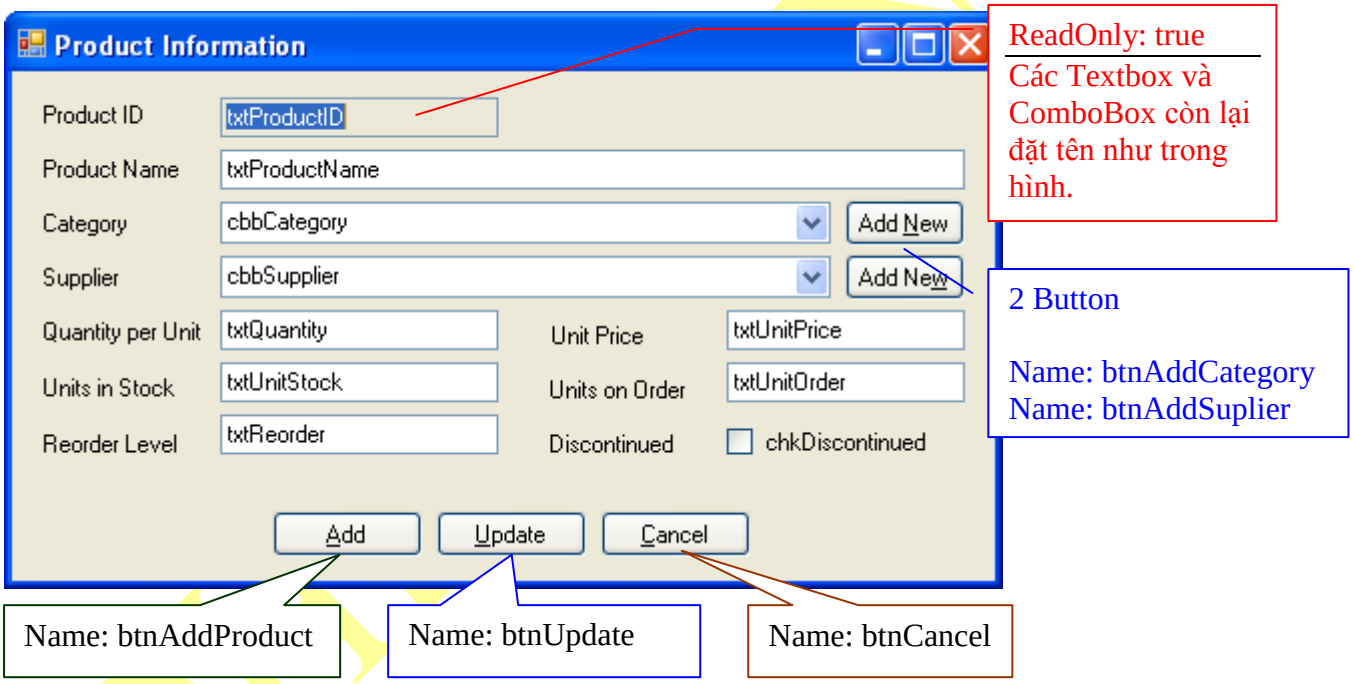
```
CREATE PROCEDURE [UpdateProducts]
        (@ProductID      [int],
         @ProductName    [nvarchar](40),
         @SupplierID     [int],
         @CategoryID     [int],
         @QuantityPerUnit [nvarchar](20),
         @UnitPrice      [money],
         @UnitsInStock   [smallint],
         @UnitsOnOrder   [smallint],
         @ReorderLevel   [smallint],
         @Discontinued   [bit])

AS
UPDATE [Northwind].[dbo].[Products]

SET
          [ProductName]    = @ProductName,
          [SupplierID]     = @SupplierID,
          [CategoryID]     = @CategoryID,
```

```
VALUES
        (@ProductName, @SupplierID,
         @CategoryID, @QuantityPerUnit,
         @UnitPrice, @UnitsInStock,
         @UnitsOnOrder, @ReorderLevel,
         @Discontinued);

SELECT @ProductID = SCOPE_IDENTITY();
GO
```

```
        [QuantityPerUnit] = @QuantityPerUnit,
        [UnitPrice]       = @UnitPrice,
        [UnitsInStock]    = @UnitsInStock,
        [UnitsOnOrder]    = @UnitsOnOrder,
        [ReorderLevel]    = @ReorderLevel,
        [Discontinued]    = @Discontinued

WHERE
        ( [ProductID]     = @ProductID)
IF @@ERROR <> 0
        RETURN  0
ELSE
        RETURN  1
```

Trong phần này, ta sẽ sử dụng 2 cách khác nhau để gọi một thủ tục trong SQL Server và học cách bắt lỗi SQL (hay ngoại lệ - Exception) bằng C#.

Thêm một Form mới, đặt tên là ProductInfoForm. Thiết kế giao diện cho Form mới như hình sau:



Nhấp đôi chuột vào ProductInfoForm để xử lý phương thức Load. Thêm các hàm sau:

```
private void ProductInfoForm_Load(object sender, EventArgs e)
{
    this.InitValues();
}

private void InitValues()
{
    string connectionString =
        "server=localhost;database=Northwind;uid=sa;pwd=sa;";
    SqlConnection conn = new SqlConnection(connectionString);
```

```
        SqlCommand cmd = conn.CreateCommand();
        cmd.CommandText =
            "SELECT CategoryID, CategoryName FROM Categories;" +
            "SELECT SupplierID, CompanyName FROM Suppliers;";

        SqlDataAdapter adapter = new SqlDataAdapter(cmd);
        DataSet ds = new DataSet();

        // mở kết nối csdl
        conn.Open();

        // Lấy dữ liệu từ csdl đưa vào DataTable
        adapter.Fill(ds);

        // Hiển thị nhóm sản phẩm và nhà cung cấp
        cbbCategory.DataSource = ds.Tables[0];  // Name: Table
        cbbCategory.DisplayMember = "CategoryName";
        cbbCategory.ValueMember = "CategoryID";

        cbbSupplier.DataSource = ds.Tables[1];  // Name: Table1
        cbbSupplier.DisplayMember = "CompanyName";
        cbbSupplier.ValueMember = "SupplierID";

        // đóng kết nối và giải phóng tài nguyên
        conn.Close();
        conn.Dispose();
    }
```

Bổ sung hàm sau để xóa dữ liệu trên các control của Form.

```
    private void ResetText()
    {
        txtProductID.ResetText();
        txtProductName.ResetText();
        cbbCategory.ResetText();
        cbbSupplier.ResetText();
        txtQuantity.ResetText();
        txtUnitOrder.ResetText();
        txtUnitPrice.ResetText();
        txtUnitStock.ResetText();
        txtReorder.ResetText();
        chkDiscontinued.Checked = false;
    }
```

Nhấp đôi chuột vào nút btnAddProduct để xử lý hàm thêm một sản phẩm mới

```
    private void btnAddProduct_Click(object sender, EventArgs e)
    {
        try
        {
            string connectionString =
                "Data Source=localhost;Initial Catalog=Northwind;" +
                "User ID=sa;Password=sa;";
            SqlConnection conn = new SqlConnection(connectionString);
```

```
    SqlCommand cmd = conn.CreateCommand();
    cmd.CommandText =
        "EXECUTE InsertProducts @productId OUTPUT, @proName, " +
        "@supId, @catId, @quantiy, @uprice, @ustock, @uorder, " +
        "@reorder, @discontinued";

    // Thêm tham số vào đối tượng Command
    cmd.Parameters.Add("@productId", SqlDbType.Int);
    cmd.Parameters.Add("@proName", SqlDbType.NVarChar, 40);
    cmd.Parameters.Add("@catId", SqlDbType.Int);
    cmd.Parameters.Add("@supId", SqlDbType.Int);
    cmd.Parameters.Add("@quantiy", SqlDbType.NVarChar, 20);
    cmd.Parameters.Add("@uprice", SqlDbType.Money);
    cmd.Parameters.Add("@ustock", SqlDbType.SmallInt);
    cmd.Parameters.Add("@uorder", SqlDbType.SmallInt);
    cmd.Parameters.Add("@reorder", SqlDbType.SmallInt);
    cmd.Parameters.Add("@discontinued", SqlDbType.Bit);

    cmd.Parameters["@productId"].Direction = ParameterDirection.Output;

    // truyền giá trị vào thủ tục qua tham số
    cmd.Parameters["@proName"].Value = txtProductName.Text;
    cmd.Parameters["@catId"].Value = cbbCategory.SelectedValue;
    cmd.Parameters["@supId"].Value = cbbSupplier.SelectedValue;
    cmd.Parameters["@quantiy"].Value = txtQuantity.Text;
    cmd.Parameters["@uprice"].Value = txtUnitPrice.Text;
    cmd.Parameters["@ustock"].Value = txtUnitStock.Text;
    cmd.Parameters["@uorder"].Value = txtUnitOrder.Text;
    cmd.Parameters["@reorder"].Value = txtReorder.Text;
    cmd.Parameters["@discontinued"].Value = chkDiscontinued.Checked;

      // mở kết nối csdl
      conn.Open();

      int numRowAffected = cmd.ExecuteNonQuery();

      // Thông báo kết quả
      if (numRowAffected > 0)
      {
          string proId = cmd.Parameters["@productId"].Value.ToString();

          MessageBox.Show("Successfully adding new product. " +
              "Product ID = " + proId, "Message");

          this.ResetText();
      }
      else
          MessageBox.Show("Adding product failed.");

      // đóng kết nối và giải phóng tài nguyên
      conn.Close();
      conn.Dispose();
  }
  // Bắt lỗi SQL và các lỗi khác
```

```
        catch (SqlException sex)
        {
            MessageBox.Show(sex.Message, "SQL Error");
        }

        catch (Exception ex)
        {
            MessageBox.Show(ex.Message, "Error");
        }
    }
```

Bổ sung thêm hàm sau vào lớp ProductInfoForm.

```
public void DisplayProductInfo(DataRowView rowView)
{
    try
    {
        // Hiển thị thông tin sản phẩm lên Form
        txtProductID.Text = rowView["ProductID"].ToString();
        txtProductName.Text = rowView["ProductName"].ToString();
        txtQuantity.Text = rowView["QuantityPerUnit"].ToString();
        txtReorder.Text = rowView["ReorderLevel"].ToString();
        txtUnitOrder.Text = rowView["UnitsOnOrder"].ToString();
        txtUnitPrice.Text = rowView["UnitPrice"].ToString();
        txtUnitStock.Text = rowView["UnitsInStock"].ToString();

        // Không chọn mục nào trong 2 ComboBox
        cbbCategory.SelectedIndex = -1;
        cbbSupplier.SelectedIndex = -1;

        // chọn nhóm sản phẩm tương ứng
        for (int index = 0; index < cbbCategory.Items.Count; index++)
        {
            DataRowView cat = cbbCategory.Items[index] as DataRowView;
            if (cat["CategoryID"].ToString() == rowView["CategoryID"].ToString())
            {
                cbbCategory.SelectedIndex = index;
                break;
            }
        }

        // Chọn nhà cung cấp tương ứng
        for (int index = 0; index < cbbSupplier.Items.Count; index++)
        {
            DataRowView cat = cbbSupplier.Items[index] as DataRowView;
            if (cat["SupplierID"].ToString() == rowView["SupplierID"].ToString())
            {
                cbbSupplier.SelectedIndex = index;
                break;
            }
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Error");
        this.Close();
    }
}
```

Nhấp đôi chuột vào nút btnUpdate để xử lý hàm cập nhật thông tin sản phẩm

```
private void btnUpdate_Click(object sender, EventArgs e)
{
    try
    {
        string connectionString =
            "Data Source=localhost;Initial Catalog=Northwind;" +
            "User ID=sa;Password=sa;";
        SqlConnection conn = new SqlConnection(connectionString);

        SqlCommand cmd = conn.CreateCommand();
        cmd.CommandText = "UpdateProducts";
        cmd.CommandType = CommandType.StoredProcedure;

        // Thêm tham số vào đối tượng Command
        cmd.Parameters.Add("@ProductId", SqlDbType.Int);
        cmd.Parameters.Add("@ProductName", SqlDbType.NVarChar, 40);
        cmd.Parameters.Add("@CategoryId", SqlDbType.Int);
        cmd.Parameters.Add("@SupplierId", SqlDbType.Int);
        cmd.Parameters.Add("@QuantityPerUnit", SqlDbType.NVarChar, 20);
        cmd.Parameters.Add("@UnitPrice", SqlDbType.Money);
        cmd.Parameters.Add("@UnitsInStock", SqlDbType.SmallInt);
        cmd.Parameters.Add("@UnitsOnOrder", SqlDbType.SmallInt);
        cmd.Parameters.Add("@ReorderLevel", SqlDbType.SmallInt);
        cmd.Parameters.Add("@Discontinued", SqlDbType.Bit);

        // truyền giá trị vào thủ tục qua tham số
        cmd.Parameters["@ProductId"].Value = txtProductID.Text;
        cmd.Parameters["@ProductName"].Value = txtProductName.Text;
        cmd.Parameters["@CategoryId"].Value = cbbCategory.SelectedValue;
        cmd.Parameters["@SupplierId"].Value = cbbSupplier.SelectedValue;
        cmd.Parameters["@QuantityPerUnit"].Value = txtQuantity.Text;
        cmd.Parameters["@UnitPrice"].Value = txtUnitPrice.Text;
        cmd.Parameters["@UnitsInStock"].Value = txtUnitStock.Text;
        cmd.Parameters["@UnitsOnOrder"].Value = txtUnitOrder.Text;
        cmd.Parameters["@ReorderLevel"].Value = txtReorder.Text;
        cmd.Parameters["@Discontinued"].Value = chkDiscontinued.Checked;

        // mở kết nối csdl
        conn.Open();

        int numRowAffected = cmd.ExecuteNonQuery();

        // Thông báo kết quả
        if (numRowAffected > 0)
        {
            MessageBox.Show("Successfully updating product. ", "Message");
            this.Close();
        }
        else
            MessageBox.Show("Adding product failed.");
```

```
            // đóng kết nối và giải phóng tài nguyên
            conn.Close();
            conn.Dispose();
        }
        // Bắt lỗi SQL và các lỗi khác
        catch (SqlException sex)
        {
            MessageBox.Show(sex.Message, "SQL Error");
        }

        catch (Exception ex)
        {
            MessageBox.Show(ex.Message, "Error");
        }
    }
```

Nhấp đôi chuột vào nút btnCancel để xử lý việc thoát khỏi Form

```
private void btnCancel_Click(object sender, EventArgs e)
{
    this.Close();
}
```

Trở lại Form1, bổ sung đoạn mã sau vào phương thức xử lý sự kiện Click của 2 menu Thêm mới và Cập nhật thông tin sản phẩm

```
private void tsmAddProduct_Click(object sender, EventArgs e)
{
    ProductInfoForm productForm = new ProductInfoForm();
    productForm.FormClosed +=
        new FormClosedEventHandler(productForm_FormClosed);

    productForm.Show(this);
}

void productForm_FormClosed(object sender, FormClosedEventArgs e)
{
    int index = cbbCategories.SelectedIndex;
    cbbCategories.SelectedIndex = -1;
    cbbCategories.SelectedIndex = index;
}

private void tsmUpdateProducts_Click(object sender, EventArgs e)
{
    // Lấy thông tin sản phẩm được chọn
    if (grwProducts.SelectedRows.Count > 0)
    {
        DataGridViewRow selectedRow =
            grwProducts.SelectedRows[0];

        DataRowView rowView =
            selectedRow.DataBoundItem as DataRowView;
```

```
            ProductInfoForm productForm = new ProductInfoForm();
            productForm.FormClosed +=
                new FormClosedEventHandler(productForm_FormClosed);

            productForm.Show(this);
            productForm.DisplayProductInfo(rowView);
        }
    }
```

IV. Bài tập thực hành

Trong các bài tập sau, yêu cầu sinh viên tạo Stored Procedure và dùng đối tượng Command để gọi, thực thi các thủ tục đó.

1. Trong Form ProductInfoForm, có hai Button dùng để thêm mới nhóm sản phẩm và nhà cung cấp.
    a. Thiết kế Form để thêm mới nhóm sản phẩm. Sau khi nhấn nút Add New để thêm mới nhóm sản phẩm rồi tắt Form này, nhóm sản phẩm mới thêm phải được đưa vào ComboBox (cbbCategory).
    b. Thiết kế Form để thêm mới một nhà cung cấp. Sau khi nhấn nút Add New để thêm mới nhà cung cấp và tắt Form này, nhà cung cấp mới thêm phải được đưa vào ComboBox (cbbSupplier).

2. Thiết kế Form: OrdersForm và viết hàm xử lý để
    a. Hiển thị danh sách hóa đơn được bán trong một ngày nào đó (yêu cầu có ô chọn ngày – sử dụng control DateTimePicker)
    b. Khi nhấp đôi chuột vào một hóa đơn nào đó thì mở một Form mới (OrderDetailsForm) để hiển thị danh mục các mặt hàng mua bởi hóa đơn đó.

3. Thiết kế Form: CustomersForm và viết các hàm xử lý để
    a. Hiển thị danh sách khách hàng
    b. Xem thông tin chi tiết một khách hàng
    c. Thêm một khách hàng mới
    d. Cập nhật thông tin khách hàng
    e. Xóa một khách hàng.
    f. Khi nhấp phải chuột vào một khách hàng, hiển thị menu sau

Xóa khách hàng
Xem danh mục hóa đơn
Xem nhật ký mua hàng

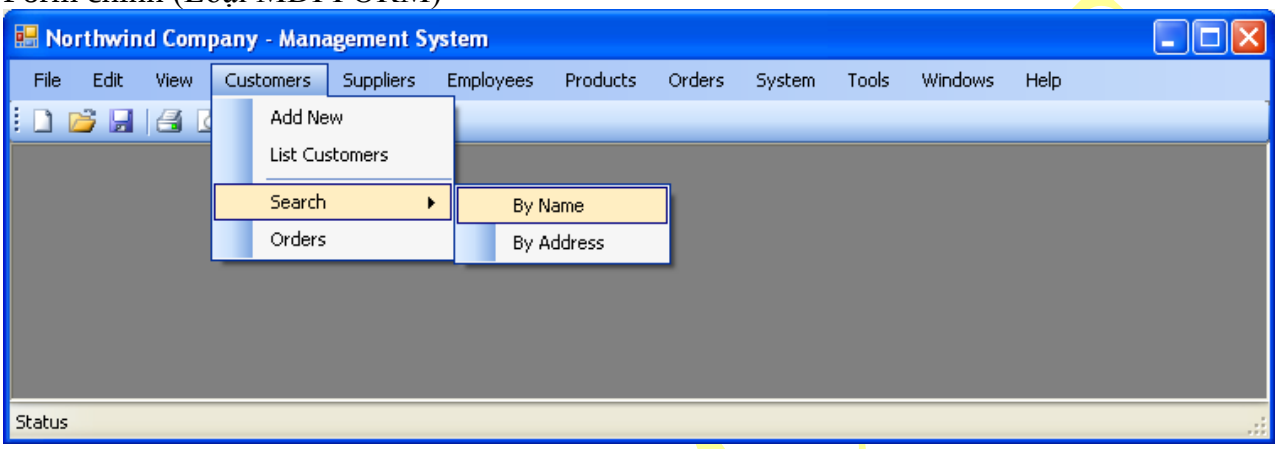
Trong đó:

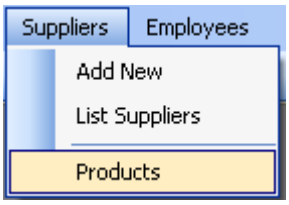
- Nếu chọn Xóa khách hàng thì dữ liệu về khách hàng đó sẽ bị xóa khỏi cơ sở dữ liệu
- Xem danh mục hóa đơn: Mở một Form mới, phần bên trái của Form là một ListBox chứa ngày lập của các hóa đơn. Khi nhấn chuột vào ngày nào thì hiển thị thông tin chi tiết (cả danh mục sản phẩm được mua) của hóa đơn ở phần bên phải Form.

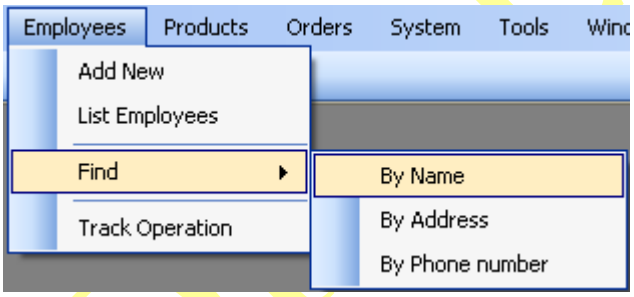
- Xem nhật ký mua hàng: Liệt kê số lượng hóa đơn (tức là số lần mua), tổng số tiền của tất cả các hóa đơn, thông tin liên quan đến từng hóa đơn như ngày lập, ngày giao hàng, địa điểm giao hàng, tên nhân viên lập hóa đơn. (sử dụng ListView hoặc DataGridView)

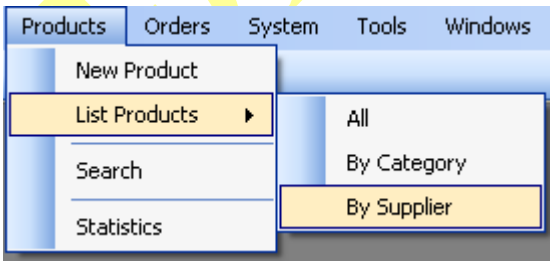
4. Tạo một dự án mới, đặt tên là NorthwinCompany. Thiết kế giao diện ứng dụng như hình sau:

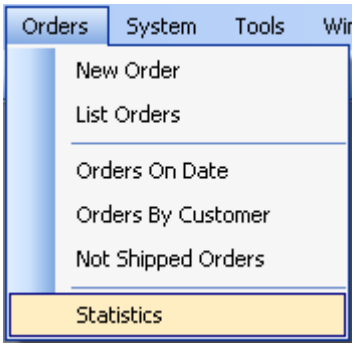
Form chính (Loại MDI FORM)

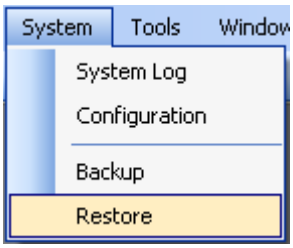












Viết hàm xử lý sự kiện Click cho từng menu tương ứng với các chức năng đã xây dựng từ bài thực hành số 2 đến bài thực hành số 3.