BÀI THỰC HÀNH SỐ 7 (4 tiết)

ADVANCED COMMAND EXCUTION

I. Mục tiêu:

Bài thực hành này giúp sinh viên tìm hiểu:

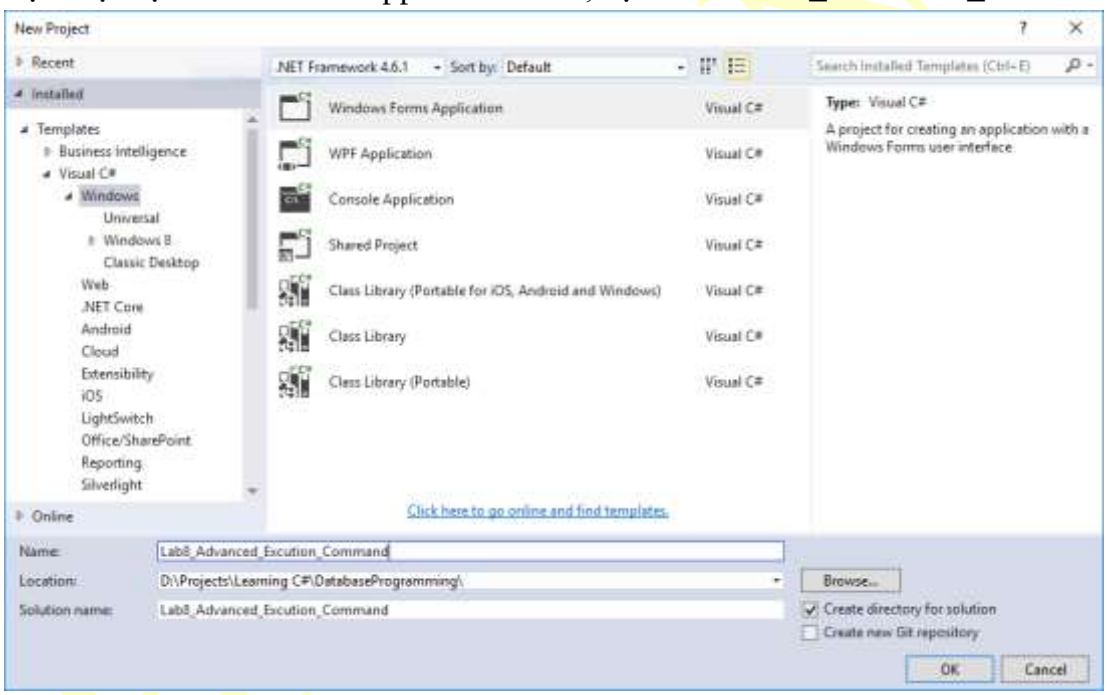
- Cách truyền tham số vào một lệnh truy vấn SQL
- Cách thực thi các lệnh SELECT, INSERT, UPDATE, DELETE có dùng tham số.
- Gọi và thực thi các Stored Procedure

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

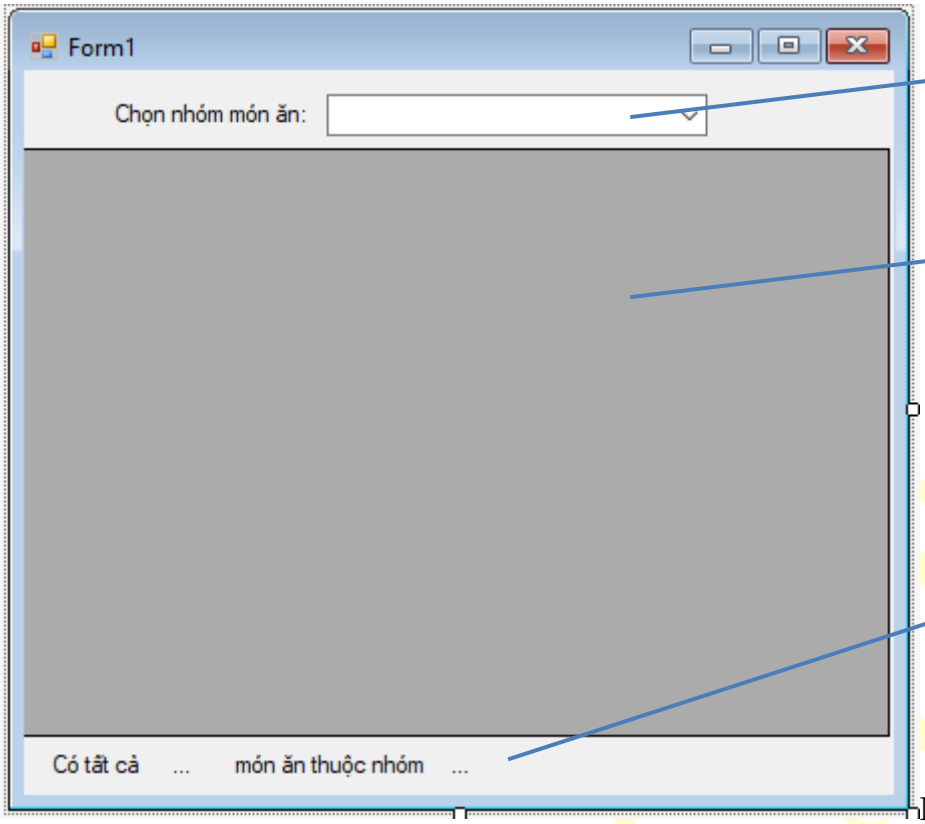
- Sử dụng đối tượng Parameter để truyền tham số vào các lệnh.
- Cách thực thi và nhận kết quả trả về từ các lệnh truy vấn có tham số
- Cách thực thi các thủ tục và nhận dữ liệu trả về.
- Cách xây dựng ứng dụng trên nền Windows Form.

II. Thực hành:

Tạo một dự án Windows Application mới, đặt tên là Lab7_Advanced_Command



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

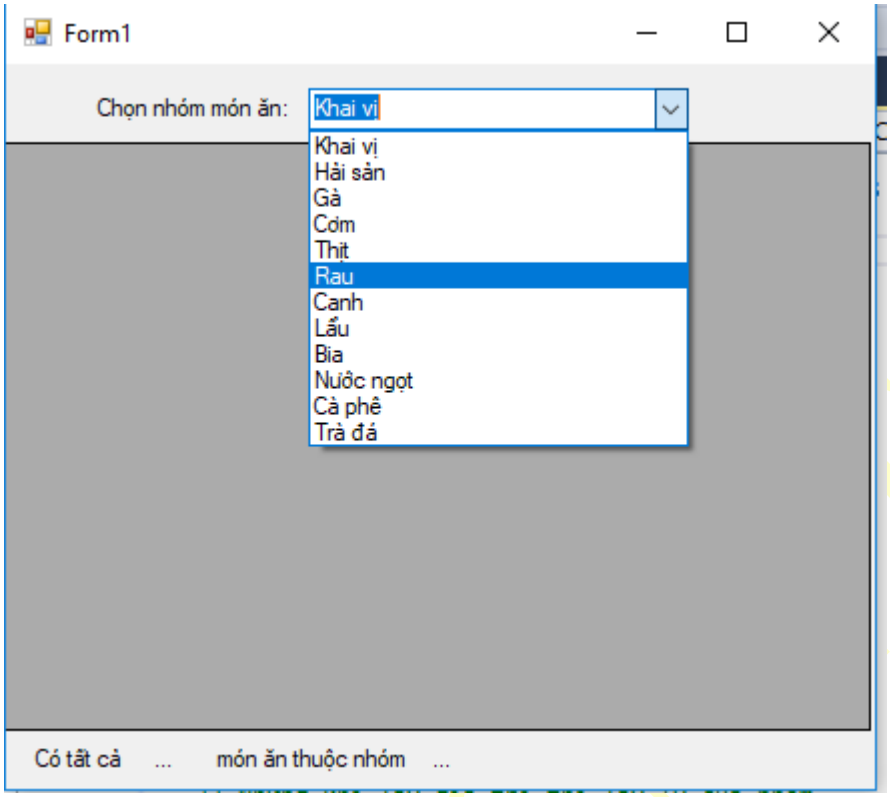


Nhập đoạn chương trình

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 LoadCategory()
{
    string connectionString = "server=.; database = RestaurantManagement; Integrated Security = true; ";
    SqlConnection conn = new SqlConnection(connectionString);

    SqlCommand cmd = conn.CreateCommand();
    cmd.CommandText = "SELECT ID, Name FROM Category";

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

    // Mở kết nối
    conn.Open();

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

    // Đóng kết nối và giải phóng bộ nhớ
    conn.Close();
    conn.Dispose();

    // Đưa dữ liệu vào Combo Box
    cbbCategory.DataSource = dt;

    // Hiển thị tên nhóm sản phẩm
    cbbCategory.DisplayMember = "Name";

    // Nhưng khi lấy giá trị thì lấy ID của nhóm
    cbbCategory.ValueMember = "ID";
}
private void Form1_Load(object sender, EventArgs e)
{
    this.LoadCategory();
}
```

Nhấn F5 để chạy chương trình



1. *Truyền tham số vào đối tượng Command*

Nhấp phải vào ComboBox, chọn Properties. Trong khung Properties, nhấn nút Events, nhấp đôi chuột vào SelectedIndexChanged. Khai báo biến cục bộ foodTable như sau:

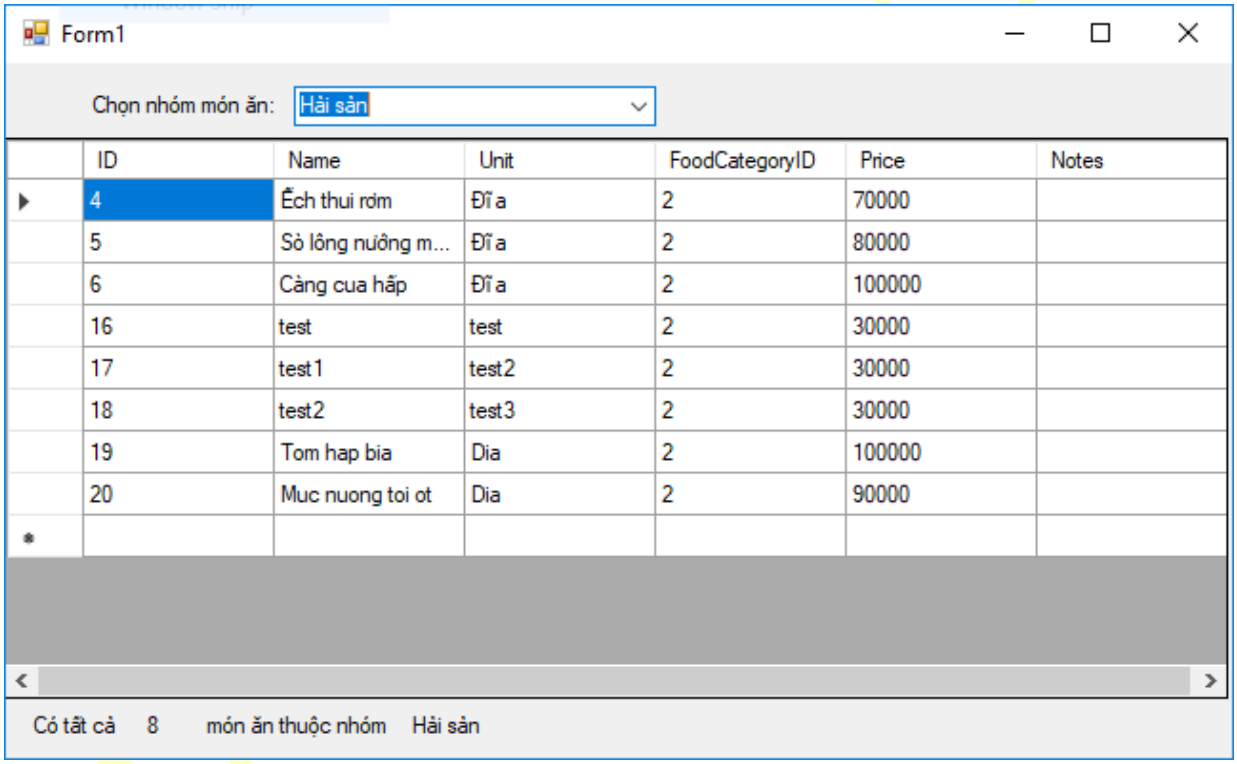
```
public partial class Form1 : Form
{
    private DataTable foodTable;

    1 reference
    public Form1()
    {
        InitializeComponent();
    }
```

Bổ sung đoạn mã sau vào phương thức cbbCategories_SelectedIndexChanged.

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

    string connectionString = "server=.; database = RestaurantManagement; Integrated Security = true; ";
    SqlConnection conn = new SqlConnection(connectionString);

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

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

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

```
    // Tạo bộ điều phiếu dữ liệu
    SqlDataAdapter adapter = new SqlDataAdapter(cmd);
    foodTable = new DataTable();

    // Mở kết nối
    conn.Open();

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

    // Đóng kết nối và giải phóng bộ nhớ
    conn.Close();
    conn.Dispose();

    // Đưa dữ liệu vào data gridview
    dgvFoodList.DataSource = foodTable;

    // Tính số lượng mẫu tin
    lblQuantity.Text = foodTable.Rows.Count.ToString();
    lblCatName.Text = cbbCategory.Text;
}
```

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



2. *Nhận giá trị trả về từ tham số*

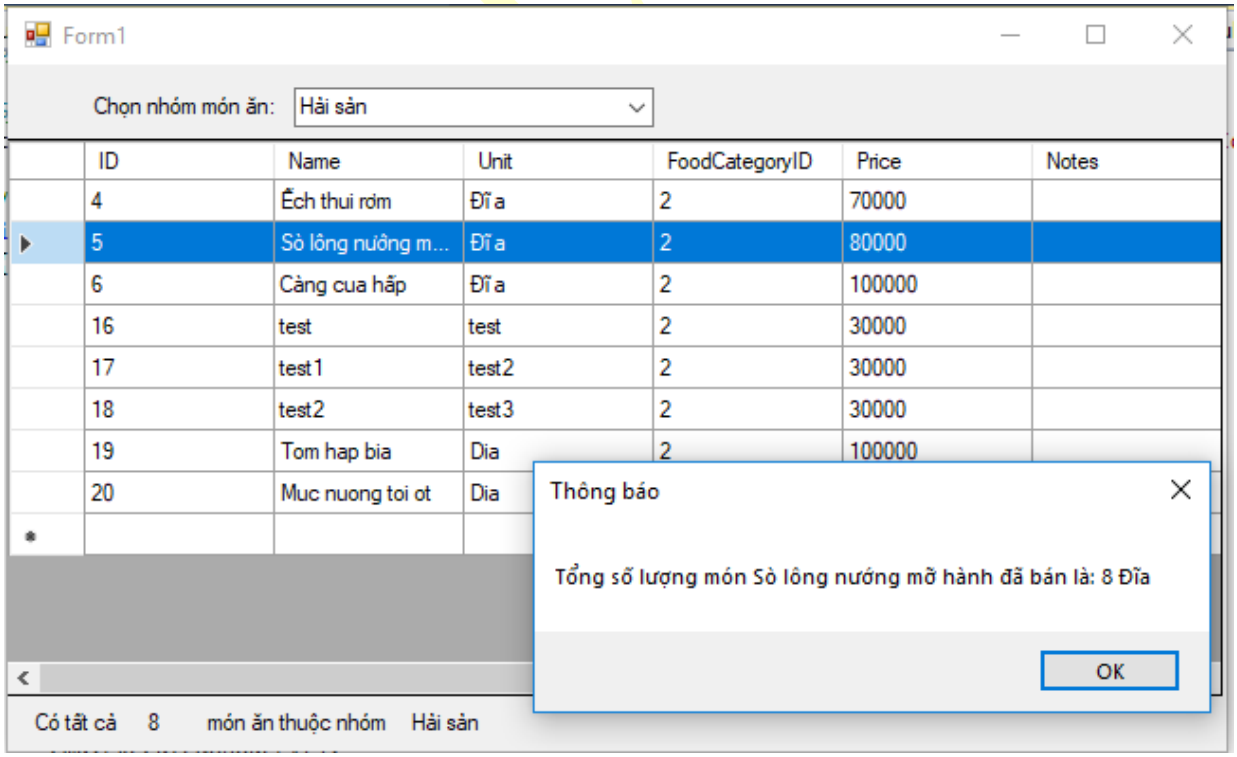
Từ thanh Toolboxs, kéo một đối tượng ContextmenuStrip vào Form1. Thiết kế menu có dạng sau (tên của các menu lần lượt là tsmCalculateQuantity, tsmSeperator, tsmAddFood, tsmUpdateFood):

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

Nhấp đôi chuột lên menu "*Tính số lượng 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=.; database = RestaurantManagement; Integrated Security = true; ";
    SqlConnection conn = new SqlConnection(connectionString);

    SqlCommand cmd = conn.CreateCommand();
    cmd.CommandText = "SELECT @numSaleFood = sum(Quantity) FROM BillDetails WHERE FoodID = @foodId";

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

        DataRowView rowView = selectedRow.DataBoundItem as DataRowView;

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

        cmd.Parameters.Add("@numSaleFood", SqlDbType.Int);
        cmd.Parameters["@numSaleFood"].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["@numSaleFood"].Value.ToString();
        MessageBox.Show("Tổng số lượng món " + rowView["Name"] + " đã bán là: " + result + " " + rowView["Unit"]);

        // Đóng kết nối csdl
        conn.Close();
    }
    cmd.Dispose();
    conn.Dispose();
}
```

Nhấn F5 chạy chương trình, chọn 1 sản phẩm và click chuột phải, kiểm tra kết quả:



Trở lại Form1, nhấp đôi chuột vào 2 menu còn lại trong contextMenuStrip1 để tạo phương thức xử lý sự kiện Click của chúng như sau:

```
private void tsmAddFood_Click(object sender, EventArgs e)
{

}

1 reference
private void tsmUpdateFood_Click(object sender, EventArgs e)
{

}
```

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

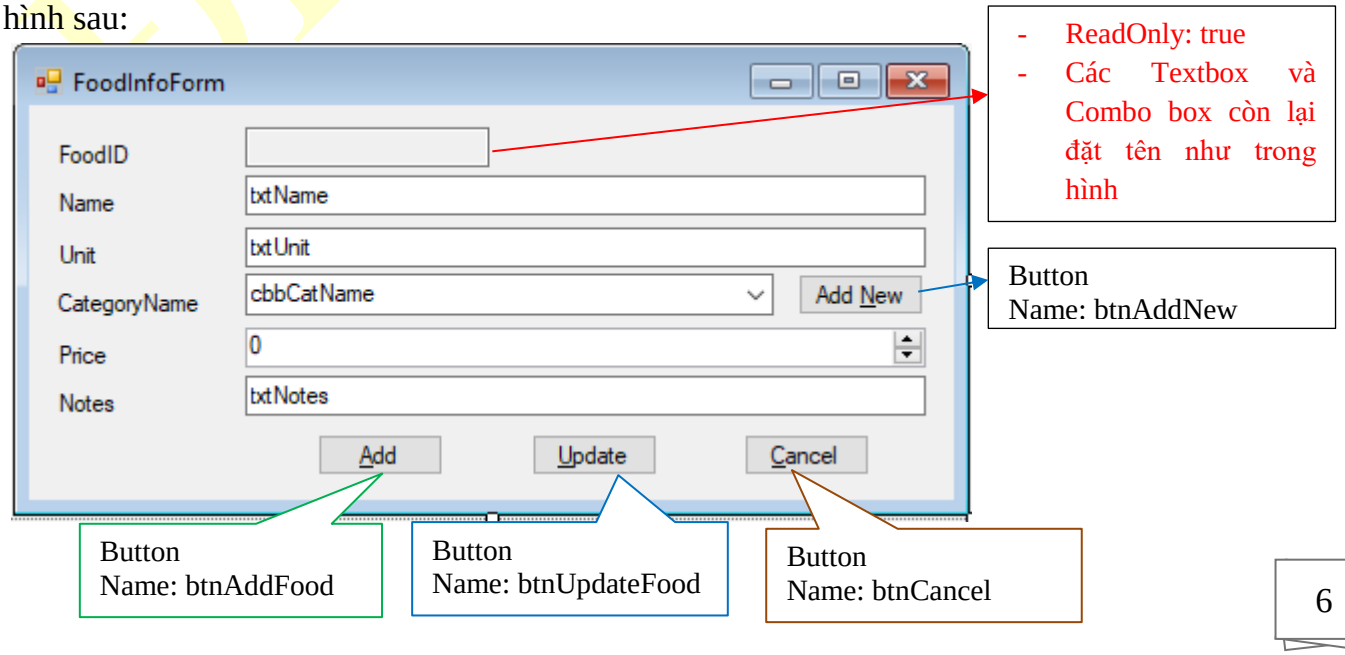
Tạo Stored Procedure sau để thực hiện thêm một món ăn mới và cập nhật thông tin món ăn

```
CREATE PROCEDURE [InsertFood]
    @ID int output,
    @Name nvarchar(1000),
    @Unit nvarchar(100),
    @FoodCategoryID int,
    @Price int,
    @Notes nvarchar(3000)
AS
INSERT INTO [Food]
([Name],[Unit],[FoodCategoryID],[Price],[Notes])
VALUES (@Name, @Unit, @FoodCategoryID, @Price,@Notes)

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

```
CREATE PROCEDURE [UpdateFood]
    @ID int,
    @Name nvarchar(1000),
    @Unit nvarchar(100),
    @FoodCategoryID int,
    @Price int,
    @Notes nvarchar(3000)
AS
UPDATE [Food]
SET
   [Name] = @Name,
   [Unit]=@Unit,
   [FoodCategoryID]= @FoodCategoryID,
   [Price]=@Price,
   [Notes]=@Notes

WHERE ID = @ID

IF @@ERROR <> 0
RETURN 0
ELSE
RETURN 1
GO
```

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 fOODInfoForm. Thiết kế giao diện cho Form mới như hình sau:



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

```
private void FoodInfoForm_Load(object sender, EventArgs e)
{
    this.InitValues();
}
private void InitValues()
{
    string connectionString = "server=.; database = RestaurantManagement; Integrated Security = true; ";
    SqlConnection conn = new SqlConnection(connectionString);

    SqlCommand cmd = conn.CreateCommand();
    cmd.CommandText = "SELECT ID, Name FROM Category";

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

    // mở kết nối
    conn.Open();

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

    // Hiển thị nhóm món ăn
    cbbCatName.DataSource = ds.Tables["Category"];
    cbbCatName.DisplayMember = "Name";
    cbbCatName.ValueMember = "ID";

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

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

```
private void ResetText()
{
    txtFoodID.ResetText();
    txtName.ResetText();
    txtNotes.ResetText();
    txtUnit.ResetText();
    cbbCatName.ResetText();
    nudPrice.ResetText();
}
```

Nhấp đôi chuột vào nút btnAddFood để xử lý hàm thêm một món ăn mới

```
private void btnAddFood_Click(object sender, EventArgs e)
{
    try
    {
        string connectionString = "server=.; database = RestaurantManagement; Integrated Security = true; ";
        SqlConnection conn = new SqlConnection(connectionString);

        SqlCommand cmd = conn.CreateCommand();
        cmd.CommandText = "EXECUTE InsertFood @id OUTPUT, @name, @unit, @foodCategoryID, @price, @notes";

        // Thêm tham số vào đối tượng Command
        cmd.Parameters.Add("@id", SqlDbType.Int);
        cmd.Parameters.Add("@name", SqlDbType.NVarChar, 1000);
        cmd.Parameters.Add("@unit", SqlDbType.NVarChar, 100);
        cmd.Parameters.Add("@foodCategoryId", SqlDbType.Int);
        cmd.Parameters.Add("@price", SqlDbType.Int);
        cmd.Parameters.Add("@notes", SqlDbType.NVarChar,3000);

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

        // Truyền giá trị vào thủ tục qua tham số
        cmd.Parameters["@name"].Value = txtName.Text;
        cmd.Parameters["@unit"].Value = txtUnit.Text;
        cmd.Parameters["@foodCategoryId"].Value = cbbCatName.SelectedValue;
        cmd.Parameters["@price"].Value = nudPrice.Value;
        cmd.Parameters["@notes"].Value = txtNotes.Text;

        //mở kết nối
        conn.Open();

        int numRowAffected = cmd.ExecuteNonQuery();

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

             MessageBox.Show("Successfully adding new food. Food ID = " + foodID, "Message");

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

         // đóng kết nối
         conn.Close();
         conn.Dispose();
    }
    // Bắt lỗi SQL và các lỗi khác
    catch (SqlException exception)
    {
        MessageBox.Show(exception.Message, "SQL Error");
    }

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

Bổ sung thêm hàm sau vào lớp FoodInfoForm.

```
public void DisplayFoodInfo(DataRowView rowView)
{
    try
    {
        txtFoodID.Text = rowView["ID"].ToString();
        txtName.Text = rowView["Name"].ToString();
        txtUnit.Text = rowView["Unit"].ToString();
        txtNotes.Text = rowView["Notes"].ToString();
        nudPrice.Text = rowView["Price"].ToString();

        cbbCatName.SelectedIndex = -1;

        // chọn nhóm món ăn tương ứng
        for (int index = 0; index < cbbCatName.Items.Count; index++)
        {
            DataRowView cat = cbbCatName.Items[index] as DataRowView;
            if (cat["ID"].ToString() == rowView["FoodCategoryID"].ToString())
            {
                cbbCatName.SelectedIndex = index;
                break;
            }
        }
    }
    catch (Exception exception)
    {
        MessageBox.Show(exception.Message, "Error");
        this.Close();
    }
}
```

Nhấp đôi chuột vào nút btnUpdateFood để xử lý hàm cập nhật thông tin món ăn

```
private void btnUpdateFood_Click(object sender, EventArgs e)
{
    try
    {
        string connectionString = "server=.; database = RestaurantManagement; Integrated Security = true;";
        SqlConnection conn = new SqlConnection(connectionString);

        SqlCommand cmd = conn.CreateCommand();
        cmd.CommandText = "EXECUTE UpdateFood @id, @name, @unit, @foodCategoryID, @price, @notes";

        // Thêm tham số vào đối tượng Command
        cmd.Parameters.Add("@id", SqlDbType.Int);
        cmd.Parameters.Add("@name", SqlDbType.NVarChar, 1000);
        cmd.Parameters.Add("@unit", SqlDbType.NVarChar, 100);
        cmd.Parameters.Add("@foodCategoryId", SqlDbType.Int);
        cmd.Parameters.Add("@price", SqlDbType.Int);
        cmd.Parameters.Add("@notes", SqlDbType.NVarChar, 3000);

        // Truyền giá trị vào thủ tục qua tham số
        cmd.Parameters["@id"].Value = int.Parse(txtFoodID.Text);
        cmd.Parameters["@name"].Value = txtName.Text;
        cmd.Parameters["@unit"].Value = txtUnit.Text;
        cmd.Parameters["@foodCategoryId"].Value = cbbCatName.SelectedValue;
        cmd.Parameters["@price"].Value = nudPrice.Value;
        cmd.Parameters["@notes"].Value = txtNotes.Text;

        //mở kết nối
        conn.Open();

        int numRowAffected = cmd.ExecuteNonQuery();
```

```
            // Thông báo kết quả
            if (numRowAffected > 0)
            {

                MessageBox.Show("Successfully updating food", "Message");

                this.ResetText();
            }
            else
            {
                MessageBox.Show("Updating food failed");
            }

            // đóng kết nối
            conn.Close();
            conn.Dispose();
        }
        // Bắt lỗi SQL và các lỗi khác
        catch (SqlException exception)
        {
            MessageBox.Show(exception.Message, "SQL Error");
        }

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

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

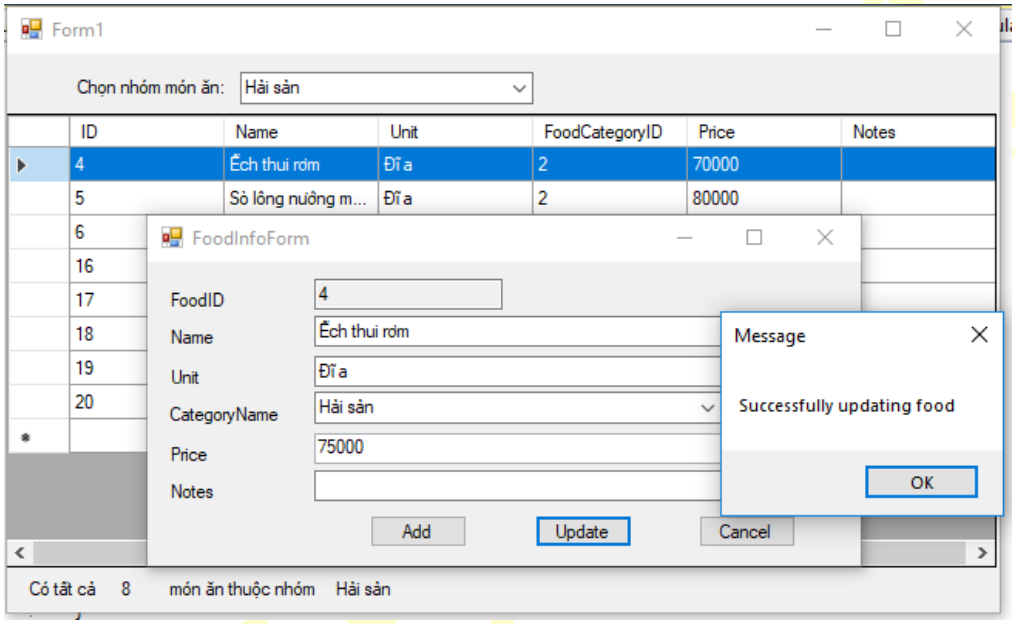
```
private void btnCancel_Click(object sender, EventArgs e)
{
    this.Close();
}
```

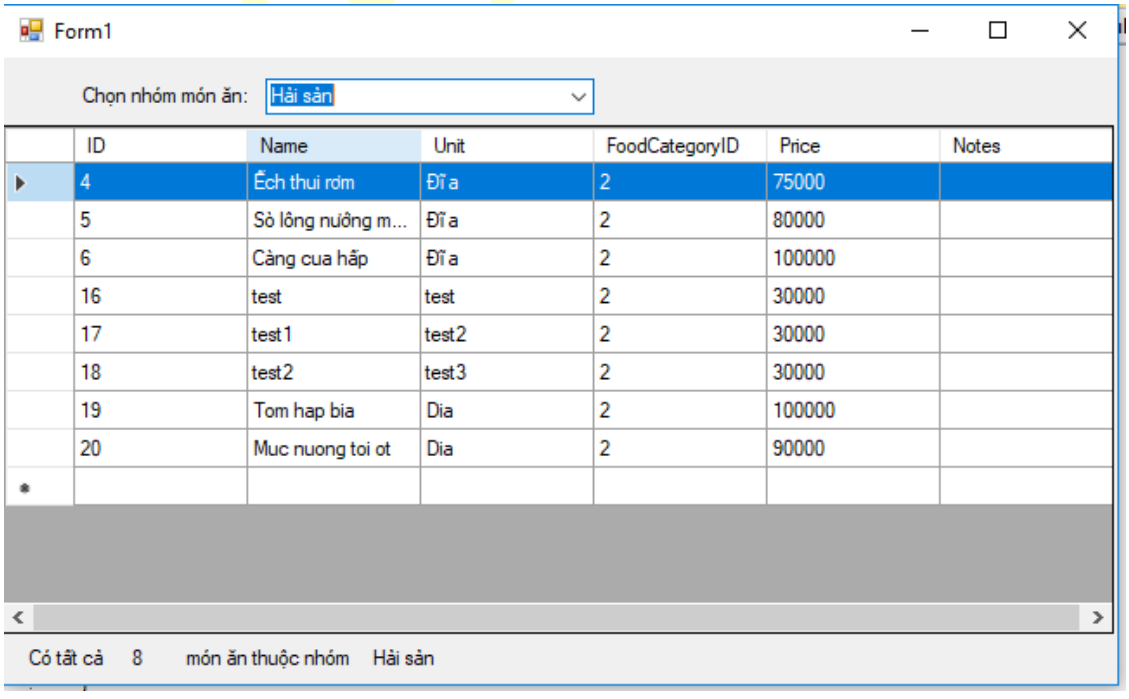
Trở lại Form1, bổ sung đoạn mã sau vào phương thức xử lý sự kiện Click của 2 menu Thêm mới và Cập nhật thông tin món ăn

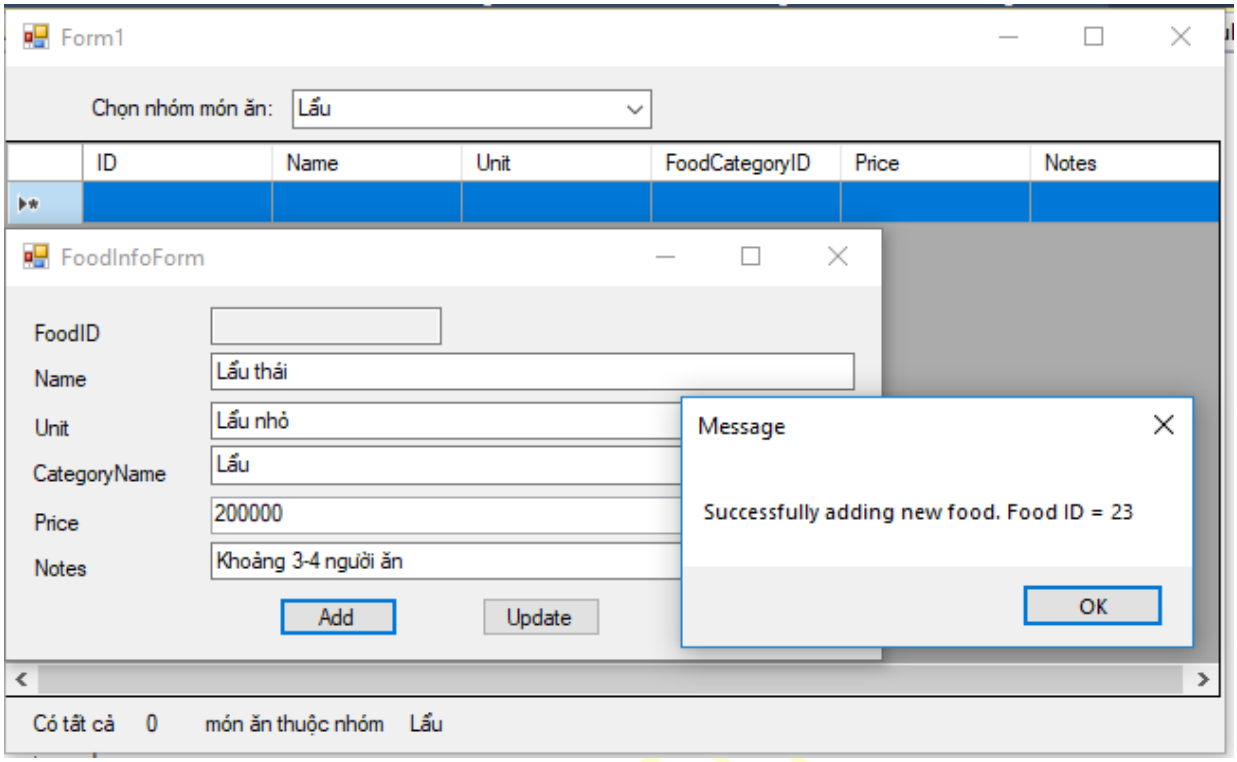
```
private void tsmAddFood_Click(object sender, EventArgs e)
{
    FoodInfoForm foodForm = new FoodInfoForm();
    foodForm.FormClosed += new FormClosedEventHandler(foodForm_FormClosed);

    foodForm.Show(this);
}

2 references
void foodForm_FormClosed(object sender, FormClosedEventArgs e)
{
    int index = cbbCategory.SelectedIndex;
    cbbCategory.SelectedIndex = -1;
    cbbCategory.SelectedIndex = index;
}
```

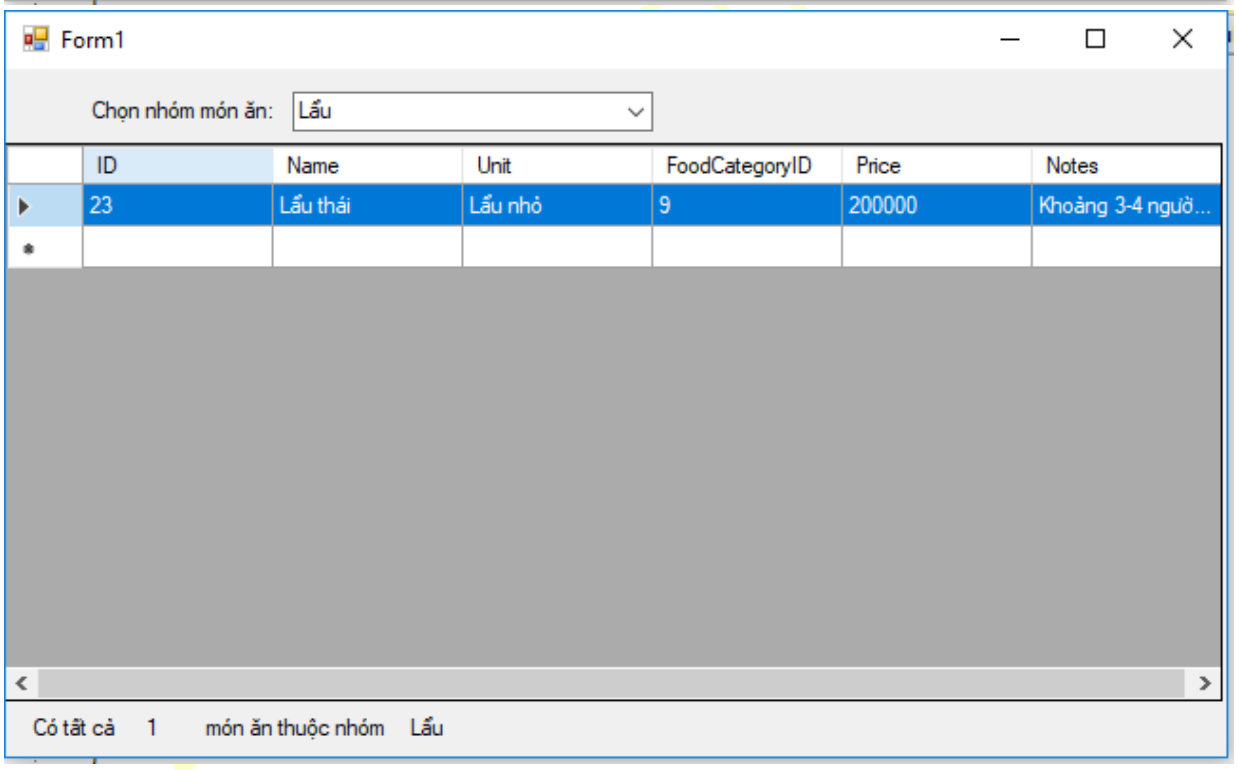
```
private void tsmUpdateFood_Click(object sender, EventArgs e)
{
    // Lấy thông tin sản phẩm được chọn
    if (dgvFoodList.SelectedRows.Count > 0)
    {
        DataGridViewRow selectedRow = dgvFoodList.SelectedRows[0];
        DataRowView rowView = selectedRow.DataBoundItem as DataRowView;

        FoodInfoForm foodForm = new FoodInfoForm();
        foodForm.FormClosed += new FormClosedEventHandler(foodForm_FormClosed);

        foodForm.Show(this);
        foodForm.DisplayFoodInfo(rowView);
    }
}
```

Chạy chương trình và kiểm tra kết quả



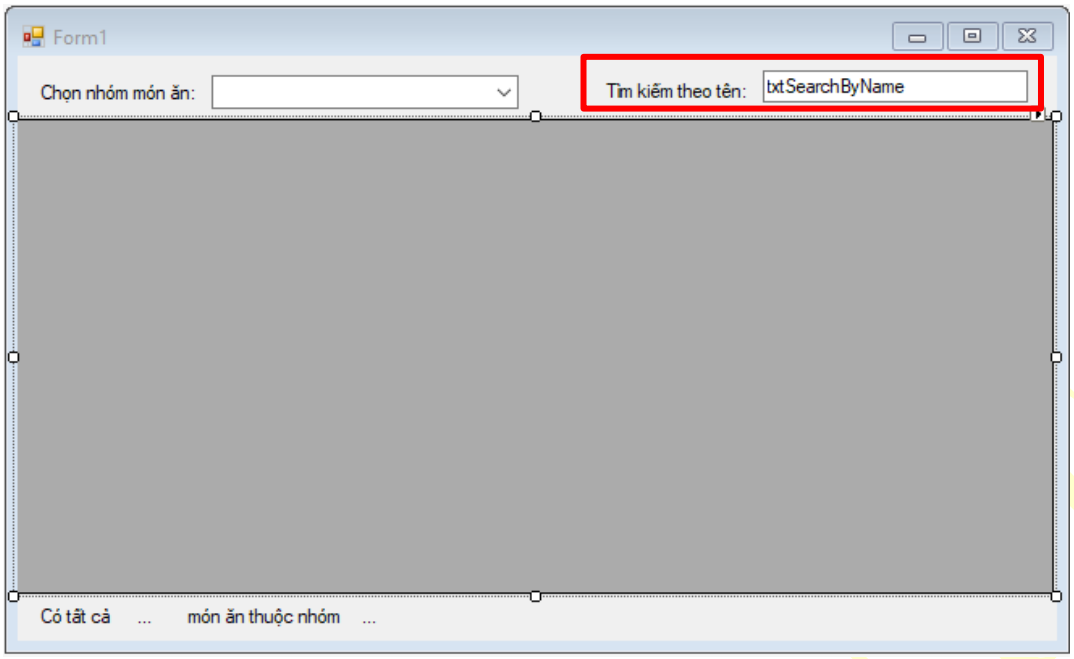






4. ***Sử dụng DataView để lọc dữ liệu***

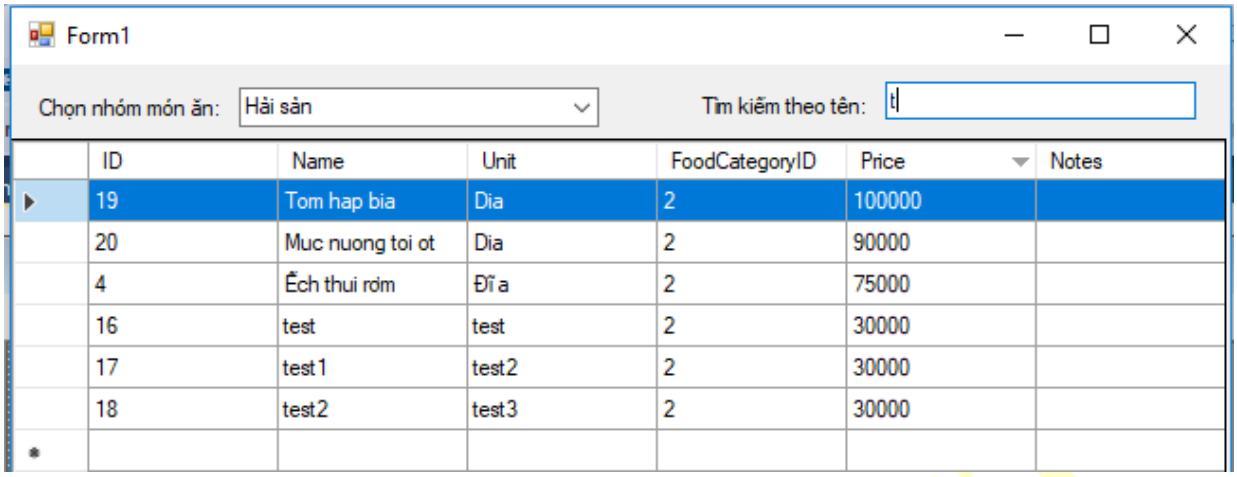
Bổ sung phần tìm kiếm theo tên món ăn vào Form1 như hình sau



Như vậy, bên cạnh việc hiển thị danh sách món ăn theo nhóm món ăn, người dùng có thể lọc món ăn dựa vào tên bằng cách sử dụng Dataview để sắp xếp và trích lọc. Bổ sung sự kiện TextChange cho txtSearchByName như sau:

```
private void txtSearchByName_TextChanged(object sender, EventArgs e)
{

    if (foodTable == null) return;

    // create filter and sort expression
    string filterExpression = "Name like '%" + txtSearchByName.Text + "%'";
    string sortExpression = "Price DESC";
    DataViewRowState rowStateFilter = DataViewRowState.OriginalRows;

    // Create a data view object to view the data in foodTable data table
    // filter by Name (contain 'ng') and sort descending by Price
    DataView foodView = new DataView(foodTable,
        filterExpression, sortExpression, rowStateFilter);

    // Assign foodTable as Data Source of data grid view
    dgvFoodList.DataSource = foodView;

}
```

Chạy chương trình, và kiểm tra kết quả



ID	Name	Unit	FoodCategoryID	Price	Notes
19	Tom hap bia	Dia	2	100000	
20	Muc nuong toi ot	Dia	2	90000	
4	Ếch thui rơm	Đĩa	2	75000	
16	test	test	2	30000	
17	test1	test2	2	30000	
18	test2	test3	2	30000	

III. Bài tập

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 FoodInfoForm, có một Button dùng để thêm mới nhóm món ăn. Hãy thiết kế Form để thêm mới nhóm món ăn. Sau khi nhấn nút Add New để thêm mới nhóm món ăn rồi tắt Form này, nhóm món ăn mới thêm phải được đưa vào ComboBox (cbbCatName).

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). Hiển thị tổng doanh thu trong ngày
   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: AccountForm và viết các hàm xử lý để
   a. Hiển thị danh sách tài khoản
   b. Thêm một tài khoản mới
   c. Cập nhật thông tin tài khoản
   d. Khi nhấp phải chuột vào một tài khoản, hiển thị menu sau

Xem danh sách các vai trò
Xem nhật ký hoạt động

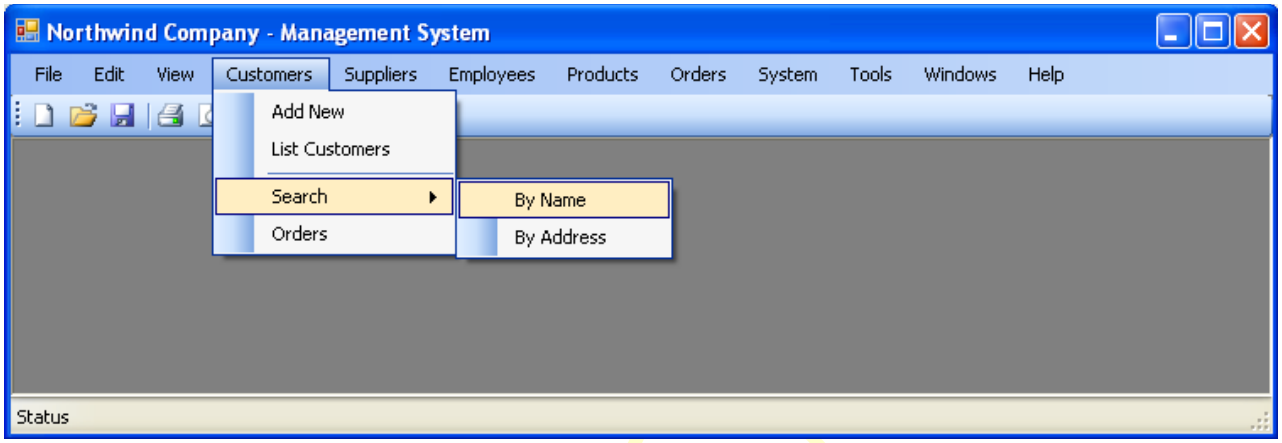
Trong đó:

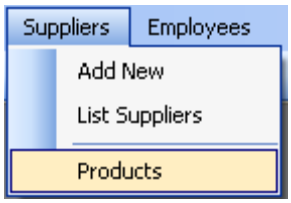
- Xem danh sách các vai trò: Mở một Form mới, hiển thị danh sách chi tiết các vai trò chủa hệ thống.Tài khoản được gán vai trò nào thì cột đầu tiên của danh sách sẽ được check. Form này cũng chứa 3 button là AddNew để thêm mới vai trò, Update để thay đổi danh sách vai trò gán cho người dùng, và Cancel để đóng Form.
- Xem nhật ký hoạt động: phần bên trái của Form là một ListBox chứa ngày lập của các hóa đơn. Khi nhấn chuột vào ngày nào thì

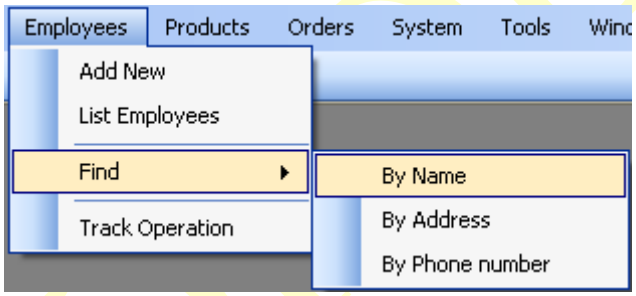
hiển thị thông tin chi tiết (cả danh mục món ăn) của hóa đơn ở phần bên phải Form. Phía dưới Form, liệt kê số lượng hóa đơn tài khoản đã lập), tổng số tiền của tất cả các hóa đơn.

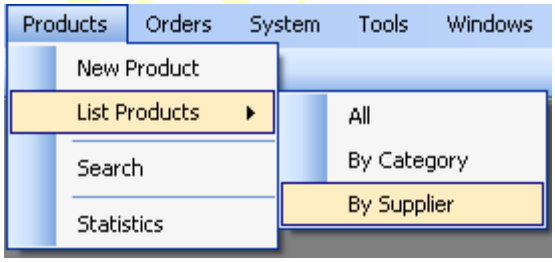
4. Tạo một dự án mới, đặt tên là ABC Restaurant. Thiết kế giao diện ứng dụng như hình sau:

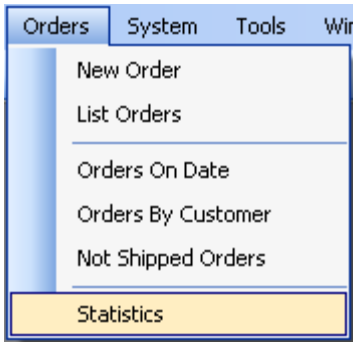
Form chính (Loại MDI FORM)

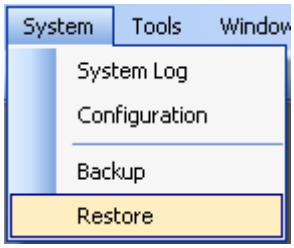












Viết hàm xử lý sự kiện Click cho từng menu tương ứng với các chức năng đã xây dựng từ bài thực hành số 2 đến bài thực hành số 3.