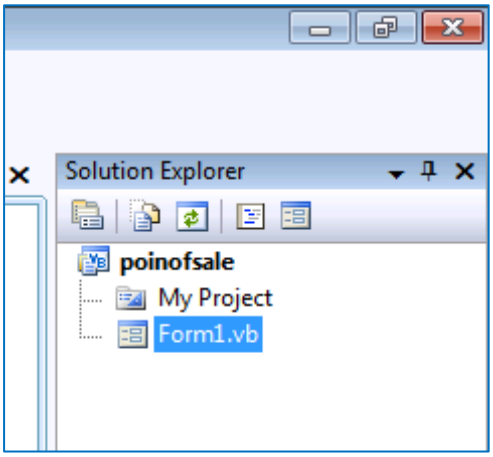
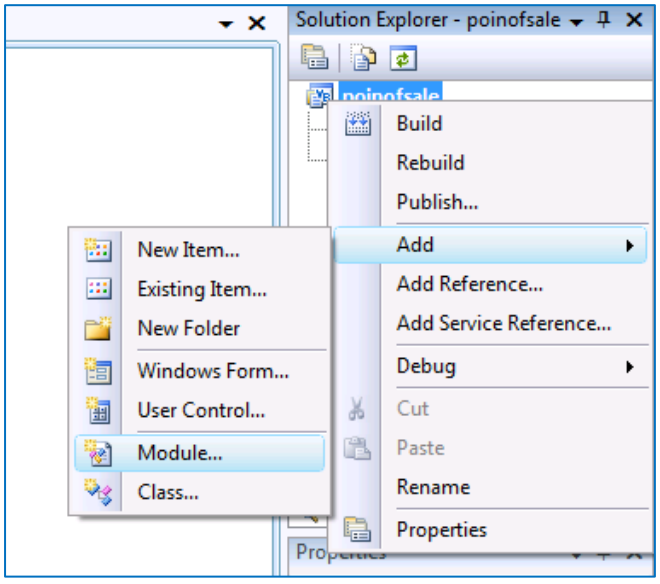
การสร้างโมดูลเพื่อติดต่อกับฐานข้อมูล

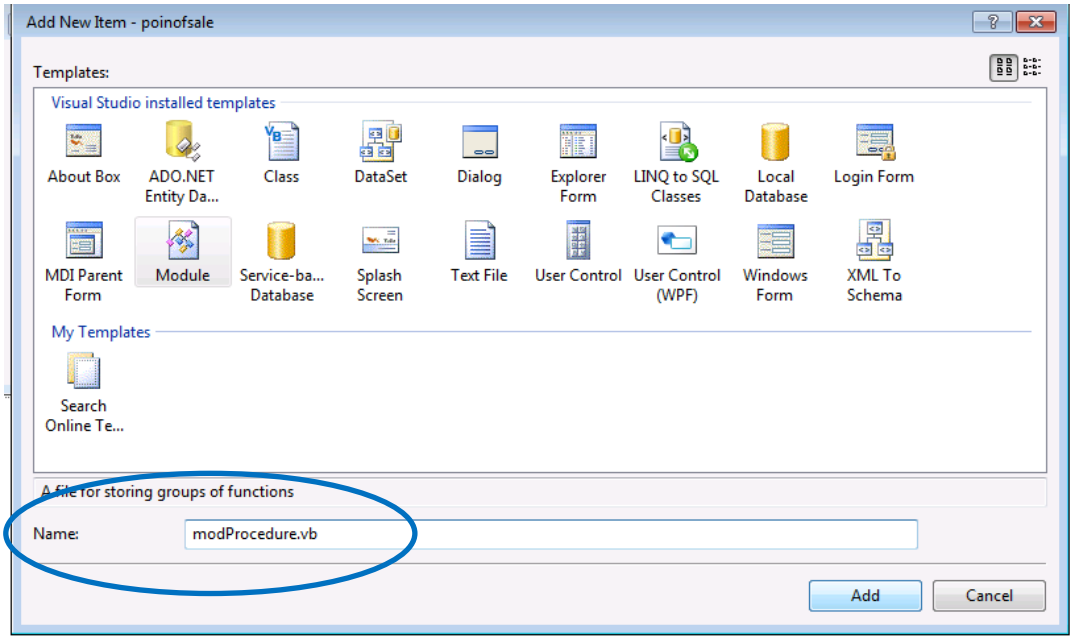
การสร้างโมดูลใน Visual Studio 2008



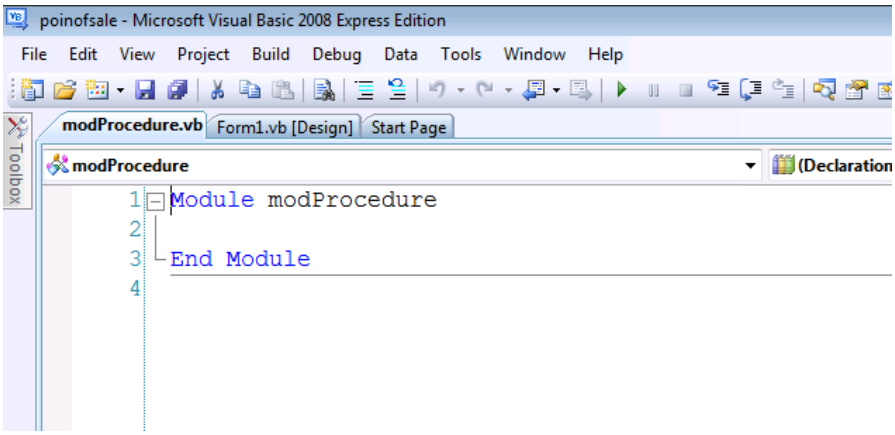
1. คลิกขวาในบริเวณพื้นที่ Solution Explorer
2. เลือกคำสั่ง Add → เลือก Module



3. ตั้งชื่อโมดูล โดยในตัวอย่างนี้ตั้งชื่อว่า modProcedure แล้วกดปุ่ม Add



4. ปรากฏหน้าจอโมดูล เพื่อให้เขียนชุดคำสั่ง



การเขียนชุดคำสั่งเพื่อเชื่อมต่อกับฐานข้อมูล

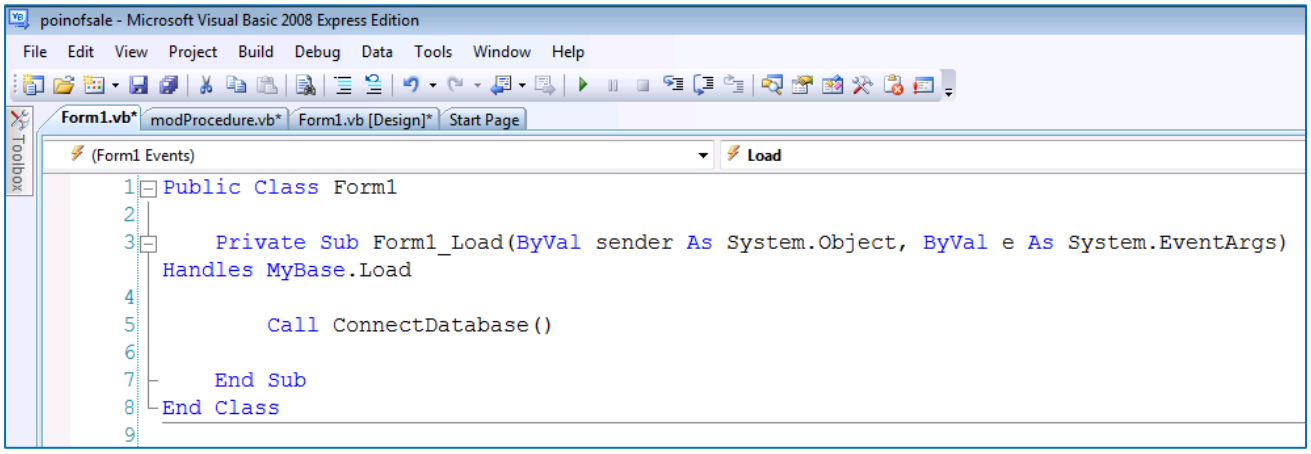
ในการติดต่อกับฐานข้อมูลในบทเรียนนี้ใช้ระบบจัดการฐานข้อมูลเป็น Microsoft Access 2010 ซึ่งสามารถติดต่อกับฐานข้อมูลได้ด้วยการเขียนคำสั่งดังนี้

```
eneral)                                          (Declarations)
 1  Imports System.Data
 2  Imports System.Data.OleDb
 3  Module modProcedure
 4      Public Cnn As New OleDbConnection
 5      Public Sub ConnectDatabase()
 6          Dim strCon = "Provider=Microsoft.ACE.OLEDB.12.0;Data Source=D:\MS-UDRU\BC\58-1\POS
    -V2\POS1.accdb;Persist Security Info=False;"
 7          If Cnn.State = ConnectionState.Open Then Cnn.Close()
 8          Cnn.ConnectionString = strCon
 9          Cnn.Open()
10      End Sub
11
12  End Module
13
```

จากภาพ คำสั่งในบรรทัดที่ 6 เป็นคำสั่งสำหรับติดต่อกับฐานข้อมูล โดยนักศึกษา ต้องตั้งค่า ในส่วน Data Source ให้เป็นเส้นทางสู่ตำแหน่งที่จัดเก็บเก็บไฟล์ฐานข้อมูลที่สร้างไว้ โดยในตัวอย่างใช้ Data Source เป็นไดรฟ์ D และเก็บไฟล์ในโฟลเดอร์ MS-UDRU โฟลเดอร์ย่อย BC โฟลเดอร์ย่อย 58-1 โฟลเดอร์ย่อย POS-V2 และไฟล์ฐานข้อมูลชื่อ POS1.accdb จึงทำให้ได้คำสั่งของ Data Source เป็น

```
Data Source=D:\MS-UDRU\BC\58-1\POS-V2\POS1.accdb
```

เมื่อตั้งค่าเส้นทางสู่ฐานข้อมูลแล้วให้เปิด Form1.vb ขึ้นมาแล้วดับเบิ้ลคลิกที่ Form แล้วพิมพ์คำสั่ง call ConnectDatabase() เพื่อเชื่อมต่อกับฐานข้อมูลดังภาพ



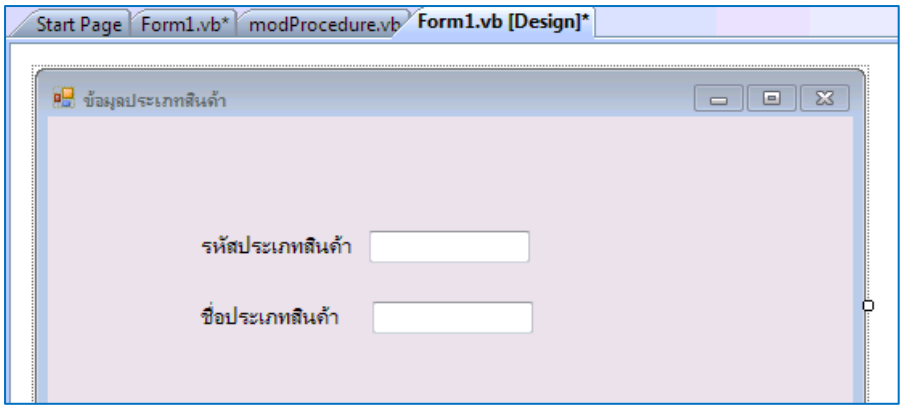
การผูกติดข้อมูล (Binding Data)

การผูกติดข้อมูล คือการแสดงข้อมูลจากฐานข้อมูลมาแสดงบนฟอร์มผ่านคอนโทรลเลอร์ตัวต่างๆ โดยวิธีการผูกติดข้อมูลมีอยู่ 2 แบบ คือ

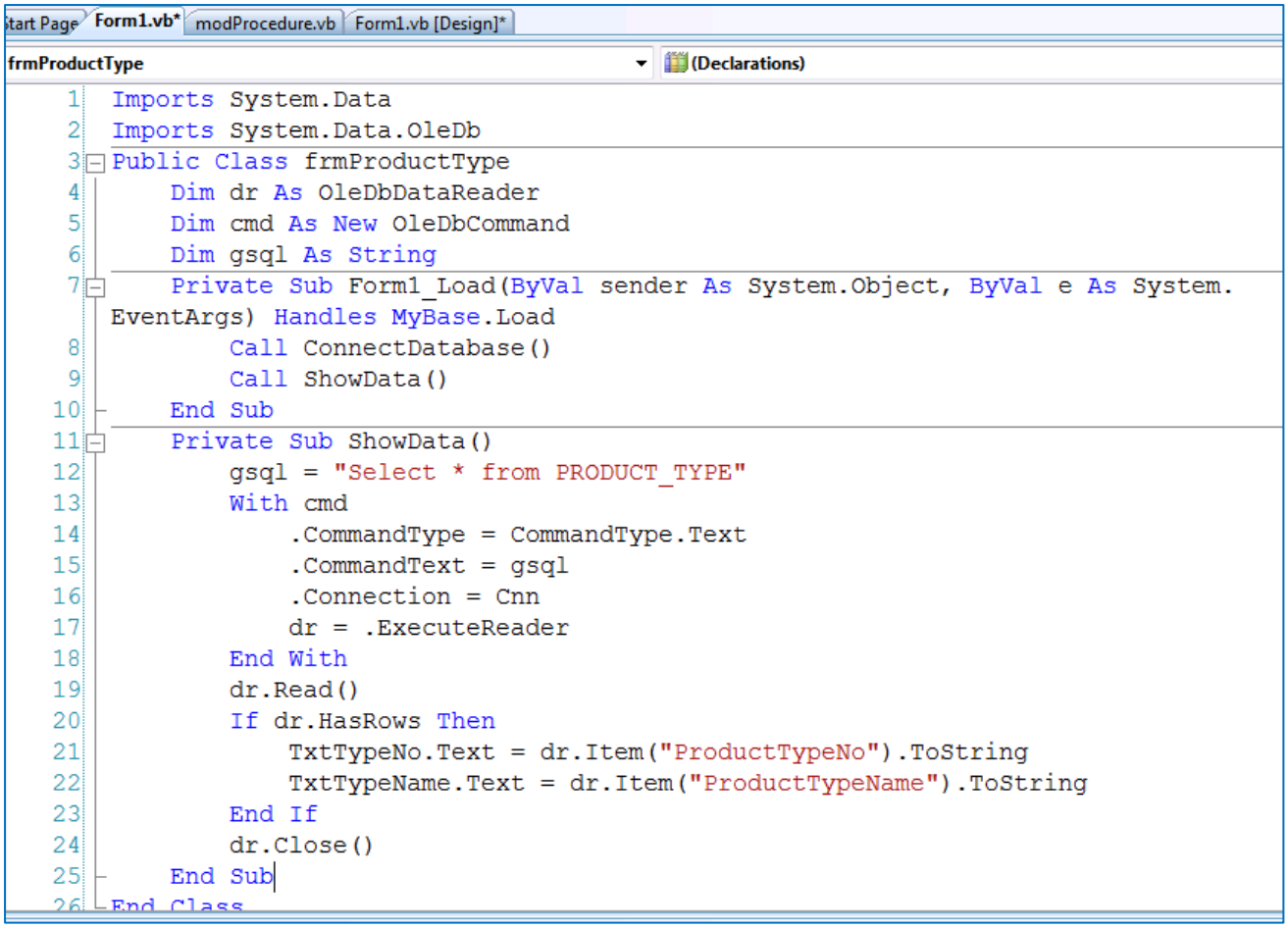
1. Simple Binding Data เป็นการแสดงข้อมูลออกมาจากตารางเดียว
2. Complex Binding Data เป็นการแสดงข้อมูลจากตารางที่สัมพันธ์กัน

การสร้างฟอร์มประเภทสินค้า

ให้นักศึกษาสร้างฟอร์มและออกแบบฟอร์มโดยใช้คอนโทรลดังภาพ



จากนั้นให้เขียนชุดคำสั่งที่เหตุการณ์ Form Load ด้วยการดับเบิ้ลคลิกที่ฟอร์มแล้วพิมพ์คำสั่ง call ShowData() ในบรรทัดที่ 9 ซึ่งเป็นคำสั่งเรียกใช้โพรซีเยอร์ที่ชื่อ ShowData มาใช้งาน และเราต้องเพิ่มชุดคำสั่งในโพรซีเยอร์ ShowData โดยโพรซีเยอร์ ShowData นี้จะใช้สำหรับการเลือกข้อมูลจากตาราง PRODUCT_TYPE มาแสดงในคอนโทรลที่ได้สร้างขึ้นมาก่อนหน้านี้ ให้นักศึกษาเขียนชุดคำสั่งเพิ่มดังภาพ

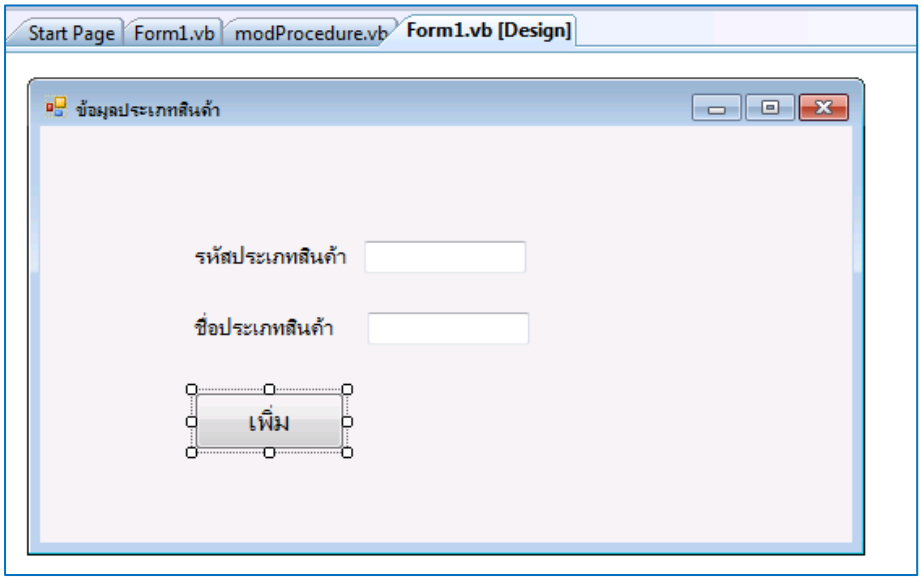



```
1   Imports System.Data
2   Imports System.Data.OleDb
3   Public Class frmProductType
4       Dim dr As OleDbDataReader
5       Dim cmd As New OleDbCommand
6       Dim gsql As String
7       Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.
        EventArgs) Handles MyBase.Load
8           Call ConnectDatabase()
9           Call ShowData()
10      End Sub
11      Private Sub ShowData()
12          gsql = "Select * from PRODUCT_TYPE"
13          With cmd
14              .CommandType = CommandType.Text
15              .CommandText = gsql
16              .Connection = Cnn
17              dr = .ExecuteReader
18          End With
19          dr.Read()
20          If dr.HasRows Then
21              TxtTypeNo.Text = dr.Item("ProductTypeNo").ToString
22              TxtTypeName.Text = dr.Item("ProductTypeName").ToString
23          End If
24          dr.Close()
25      End Sub
26  End Class
```

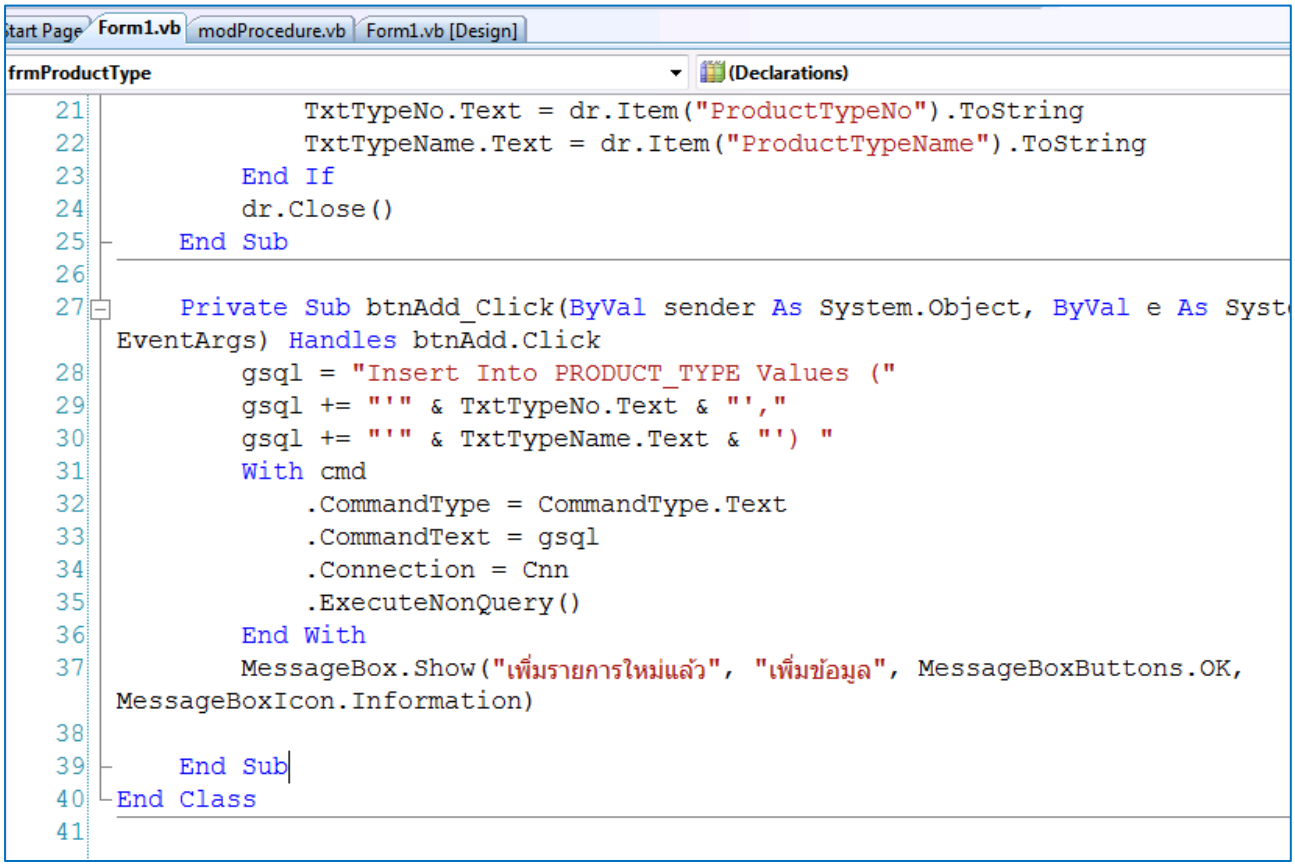
จากชุดคำสั่งข้างต้นสามารถอธิบายการทำงานของ โพรซีเยอร์ ShowData ได้ดังนี้

คำสั่งบรรทัดที่	**คำอธิบาย**
11	โพรซีเยอร์ชื่อ ShowData
12	กำหนดค่าให้กับตัวแปร gsql เป็นคำสั่ง SQL
13	กำหนดการทำงานเกี่ยวกับคลาส OleDbCommand เพื่อเก็บข้อมูลที่เป็นคำสั่งการทำงานกับฐานข้อมูล
14	กำหนดประเภทคำสั่งให้เป็นชนิดข้อความ
15	กำหนดข้อความคำสั่งให้ = ตัวแปร gsql (ซึ่ง gsql ถูกกำหนดไว้ในบรรทัดที่ 12)
16	กำหนดการติดต่อกับฐานข้อมูลให้ = ตัวแปร Cnn (Cnn ถูกกำหนดค่าจากโมดูล modProcedure)
17	กำหนดให้ตัวแปร dr ทำงานด้วยค่าการทำงานในข้างต้น
18	จบการทำงานของกลุ่มคำสั่งคลาส OleDbCommand
19	อ่านค่าข้อมูลและพักข้อมูลไว้ที่ตัวแปร dr
20	ทดสอบเงื่อนไข โดยตรวจสอบว่า ถ้าตัวแปร dr มีข้อมูลที่ถูกอ่านขึ้นมาจริง ให้ทำงานตามคำสั่งบรรทัดที่ 21,22
21	ให้คอนโทรล ชื่อ TxtTypeNo มีค่า= ค่าที่ dr อ่านขึ้นมาได้โดยให้มีค่า =ฟิล์ด ProductTypeNo ในฐานข้อมูล
22	ให้คอนโทรล ชื่อ TxtTypeName มีค่า= ค่าที่ dr อ่านขึ้นมาได้โดยให้มีค่า =ฟิล์ด ProductTypeName ในฐานข้อมูล
23	ปิดการอ่านข้อมูลจากฐานข้อมูลจาก dr
24	จบการทำงานของโพรซีเยอร์ ShowData

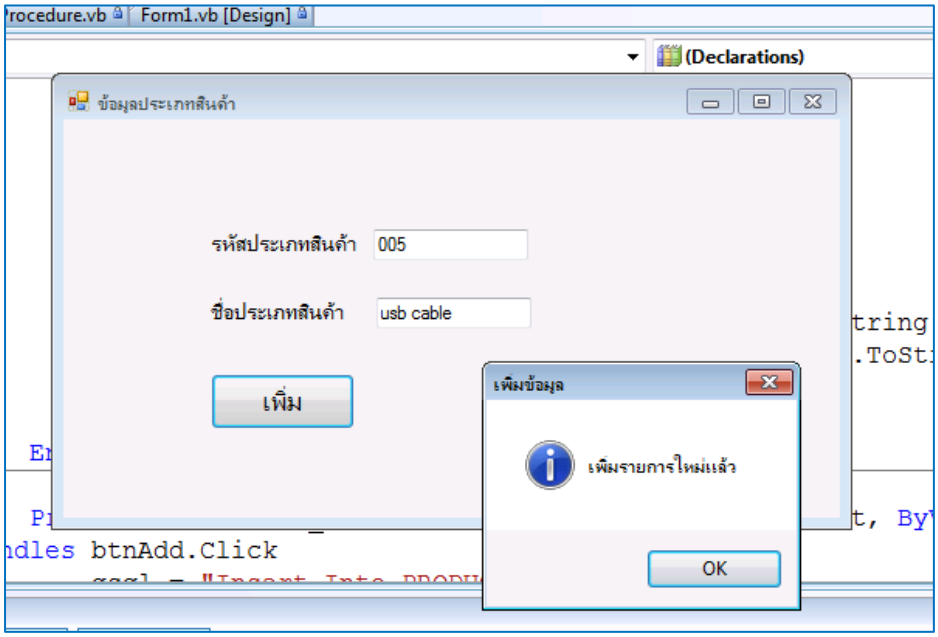
การสร้างคำสั่งสำหรับบันทึกข้อมูล



เขียนชุดคำสั่งที่ปุ่ม “เพิ่ม” ดังนี้

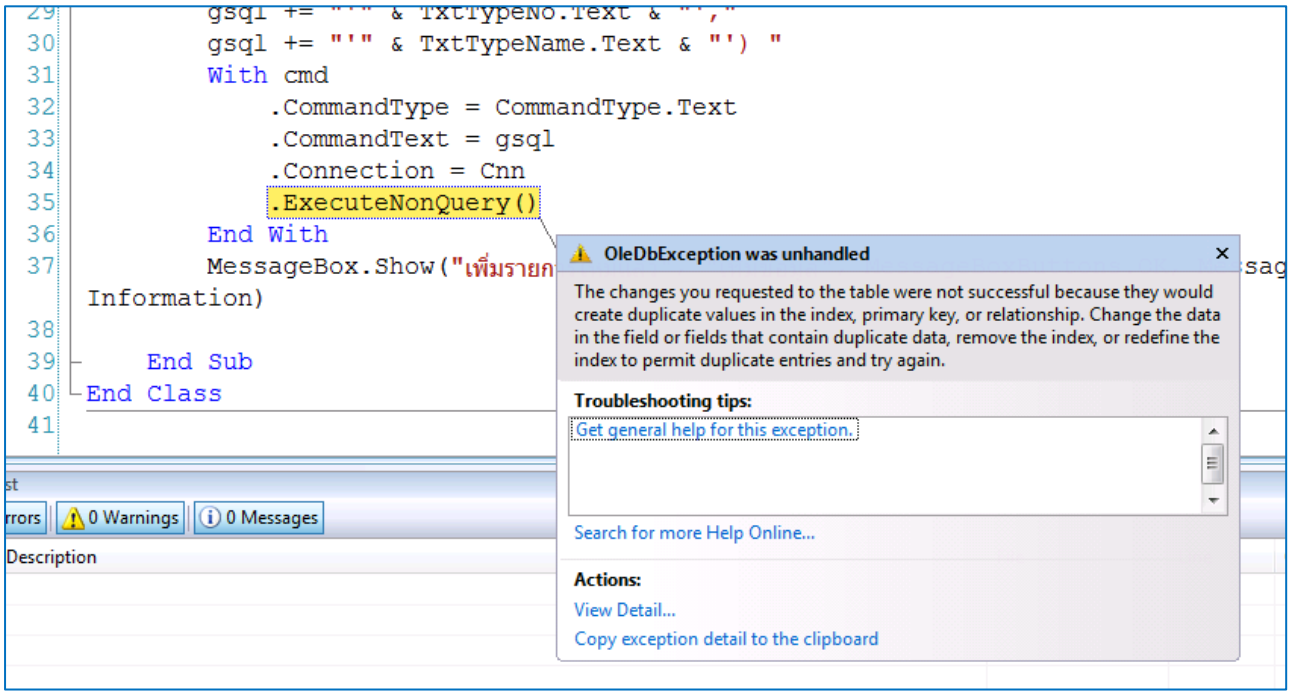


ทดสอบการทำงานของปุ่มเพิ่ม

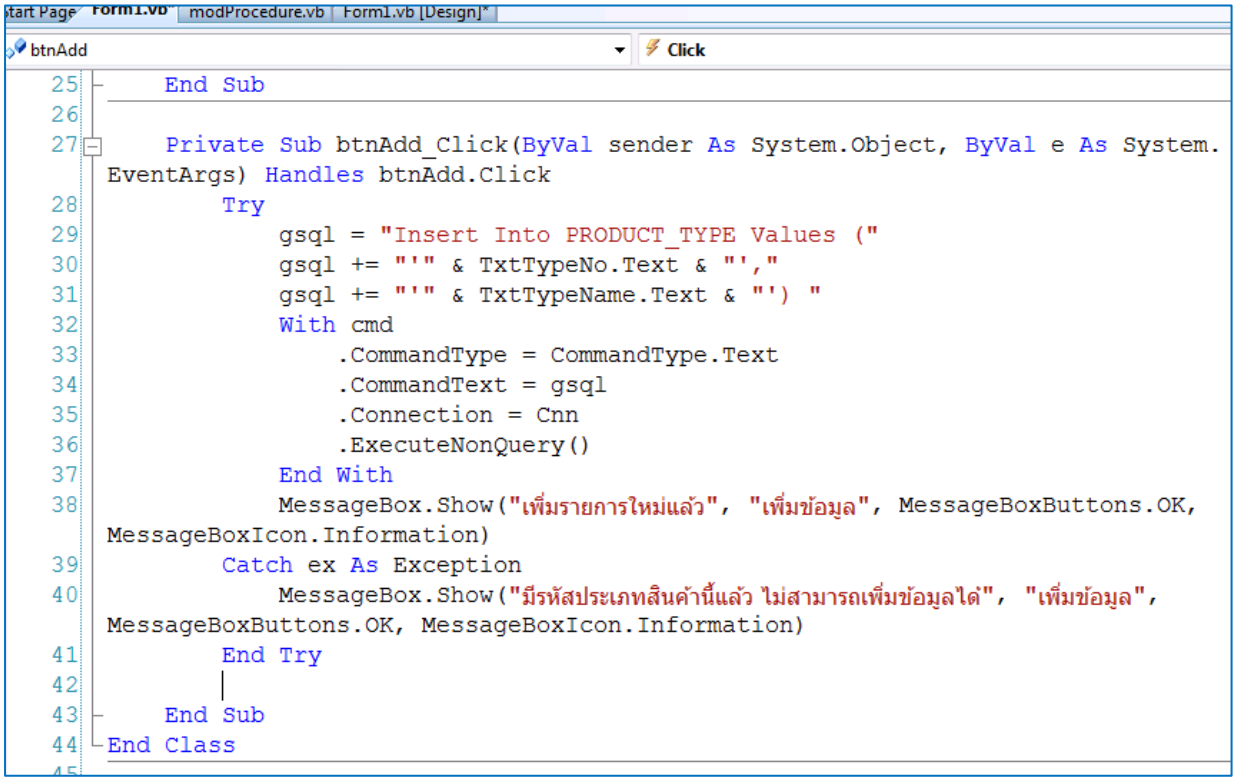


การปรับปรุงชุดคำสั่งเพื่อป้องกันข้อผิดพลาด

หลังจากที่ทดสอบว่าชุดคำสั่งนี้สามารถเพิ่มข้อมูลลงในฐานข้อมูลได้จริงแล้ว ชุดคำสั่งข้างต้นยังมีโอกาสเกิดข้อผิดพลาดได้ในกรณีที่ผู้ใช้งานอาจป้อนรหัสประเภทสินค้าที่ซ้ำกับรหัสประเภทสินค้าที่มีอยู่ในฐานข้อมูลอยู่แล้ว โดยหากมีข้อผิดพลาดในการป้อนข้อมูลซ้ำจะเกิดการ Error ดังภาพ



เนื่องจากในฐานข้อมูลกำหนดให้รหัสประเภทสินค้าเป็นคีย์หลัก ซึ่งไม่สามารถมีข้อมูลที่ซ้ำกันได้ เราจึงสามารถเขียนโค้ดเพื่อป้องกันข้อผิดพลาดได้ดังภาพ โดยเพิ่มโค้ด Try ... Catch เข้าไป



จากภาพมีการแทรกคำสั่ง Try ในบรรทัดที่ 28 และคำสั่ง Catch . . . . ในบรรทัดที่ 39 – 41

ประโยค Try ... Catch เป็นคำสั่งที่ทำให้ชุดคำสั่งที่อยู่หลัง Try ทำงาน แต่หากชุดคำสั่งที่อยู่หลัง try เกิดข้อผิดพลาด จะให้โปรแกรมไปทำงานในคำสั่งที่อยู่หลัง Catch แทน ซึ่งในตัวอย่างได้สั่งให้แสดงกล่องข้อความแจ้งเตือนว่า “มีรหัสประเภทสินค้าแล้ว ไม่สามารถเพิ่มข้อมูลได้” โดยวิธีนี้จะทำให้โปรแกรมไม่หยุดการทำงานเนื่องจากการเกิด Error