เรียนรู้
Access VBA

โดย ทองจุล ขันขาว
thongjoon@yahoo.com.sg

ศูนย์การศึกษานอกโรงเรียนภาคกลาง
กระทรวงศึกษาธิการ
http://www.crnfe.ac.th

สงวนลิขสิทธิ์ 2548

คำนำ

เอกสารเรียนรู้ Access VBA นี้ จัดทำขึ้น เพื่อประกอบการอบรมวิชา MS Access ตามหลักสูตรการศึกษาต่อเนื่อง ของ ศูนย์การศึกษานอกโรงเรียนภาคกลาง โดยมีเนื้อหาเน้นเรื่องการใช้ VBA เบื้องต้น ผู้ศึกษาไม่จำเป็นต้องมีพื้นฐานด้านการเขียนโปรแกรมมาก่อนก็สามารถเรียนรู้ได้ แต่ต้องมีความรู้พื้นฐานเกี่ยวกับการใช้โปรแกรม MS Access มาแล้ว เอกสารเล่มนี้จะมีการอ้างอิงถึงไฟล์ mdb ซึ่งมีประกอบมาพร้อมกับเอกสารนี้ ในการเรียนจากเอกสารเล่มนี้ ควรเปิดโปรแกรม MS Access และทำตามเอกสาร หรือตรวจสอบกับไฟล์ตัวอย่าง จะทำให้เข้าใจได้ดียิ่งขึ้น

VBA หรือ Visual Basic for Applications เป็นภาษาที่ช่วยเพิ่มศักยภาพของโปรแกรมให้สูงขึ้น VBA ถูกใช้กับโปรแกรมในชุด Microsoft Office เช่น Access, Excel และ Word เป็นต้น การเรียนรู้ VBA จึงจำเป็นสำหรับผู้ที่ต้องการ ใช้ประโยชน์จากโปรแกรมในชุด Microsoft Office อย่างแท้จริง

ศูนย์การศึกษานอกโรงเรียนภาคกลาง ขอขอบคุณ อาจารย์ทองจุล ขันขาว ผู้เขียน และอาจารย์นฤมล อันตะริกานนท์ บรรณาธิการ และ ศูนย์การศึกษานอกโรงเรียนหวังเป็นอย่างยิ่งว่าเอกสารเล่มนี้ จะก่อประโยชน์ให้แก่ผู้ที่สนใจศึกษาค้นคว้าได้เป็นอย่างดี

(นายชีวิต อุจวาที)

ผู้อำนวยการศูนย์การศึกษานอกโรงเรียนภาคกลาง

สารบัญ

คำนำ 2
สารบัญ 3
ความรู้เบื้องต้น เกี่ยวกับ VBA 4
เรื่องของตัวแปร 4
เรื่องของ Object 4
Objects และ Collections 5
การอ้างถึง Objects 6
การอ้างถึง Form หรือ Report 6
การอ้างถึงคุณสมบัติของ Form หรือ Report 7
การอ้างถึงคุณสมบัติของ Control ที่อยู่บน Form หรือ Report 7
การเรียกค่า จากฟอร์มที่เปิดอยู่ 9
การสร้าง Procedure 9
เรื่อง Control Structure 12
เรื่องของ DAO (Data Access Object) 20
การใช้งาน Recordsets 23
การประกาศตัวแปร Recordset 23
การไปยัง records ต่าง ๆ ใน Recordset 24
ขอบเขตของ Recordset 25
การนับจำนวน Recordset 26
การสร้าง Recordset จากคำสั่ง SQL 27
การเพิ่มข้อมูลในตาราง 27
การลบข้อมูลในตาราง 27
การแก้ไขข้อมูลในตาราง 28
แหล่งอ้างอิง 29

ความรู้เบื้องต้น เกี่ยวกับ VBA

Access VBA (Visual Basic for Applications) เป็นโปรแกรมที่ช่วยขยายขีดการทำงานของ โปรแกรม MS Access ให้มีขีดความสามารถในการทำงานเพิ่มมากยิ่งขึ้น VBA เป็นโปรแกรมเชิงวัตถุ จึงควรเข้าใจหลักการเบื้องต้น ของ วัตถุ เสียก่อน

เรื่องของตัวแปร

ตัวแปรของ VBA แบ่งออกเป็นหลายประเภท ตัวแปรที่ควรรู้จัก มีดังนี้

String	ตัวแปรเก็บตัวอักษร เช่น strName = “มารศรี ดีมาก”
Date	เก็บข้อมูลประเภท วันที่ เช่น dtmOrderDate = #3/3/2547#
Currency	เก็บข้อมูลจำนวนเงิน จะมีทศนิยม สองตำแหน่ง
Boolean	เก็บข้อมูลประเภท จริงหรือเท็จ
Byte	เก็บข้อมูลตัวเลขจำนวนเต็ม ตั้งแต่ 0-255
Integer	เก็บข้อมูลตัวเลขจำนวนเต็ม ตั้งแต่ -32,768 ถึง 32,767
single	เก็บข้อมูลตัวเลขทศนิยม

ใน VBA จะต้องมีการประกาศตัวแปรว่าเป็นตัวแปรชนิดใด เช่น

Dim strName AS string ‘ บอกว่า ตัวแปร strName เป็นตัวแปรประเภท string

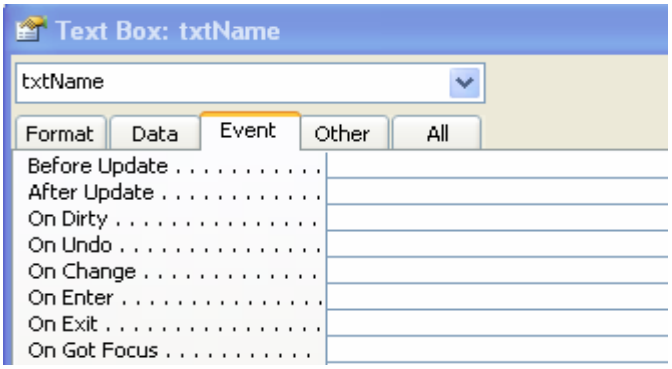
เรื่องของ Object

Object หรือวัตถุ จะมีคุณลักษณะ property ซึ่งจะบอกให้ทราบถึงลักษณะของวัตถุนั้น ๆ ใน MS Access ฟอร์มถือเป็นวัตถุ ๆ หนึ่ง รายงาน ก็เป็นวัตถุ ๆ หนึ่ง นอกจากนี้ Controls ต่าง ๆ ที่อยู่บนฟอร์ม ก็เป็นวัตถุ ๆ หนึ่งเช่นกัน และจะมี property หรือคุณสมบัติที่แตกต่างกันออกไป เช่น สีของ background ของ Control นั้น ๆ ค่า หรือ value ที่มีอยู่ใน control นั้น เป็นต้น

แต่ละ object จะมี method หรือ สิ่งที่วัตถุนั้น ๆ ทำได้ เช่น ฟอร์มสามารถจัดเรียงข้อมูลใหม่ได้ โดยมี method ชื่อ requery ซึ่งสามารถสังเกตได้ ดังนี้ Forms!frmCustomer.requery

นอกจากนี้ แต่ละ object ยังมี event หรือ เหตุการณ์ ของวัตถุนั้น ๆ เช่น เมื่อมีการเปิดฟอร์ม หรือปิดฟอร์ม หรือเมื่อมีการคลิกเมาส์ที่ฟอร์ม หรือที่ control เป็นต้น ในแต่ละ event จะมี

property ควบคู่กันไป event property นี้ สามารถดูได้จาก Event category ใน Property Sheet ของ แต่ละ Object ดังภาพข้างล่างนี้



เราเขียนโปรแกรม โดยกำหนด event property แต่ละ Object ว่าจะให้เกิดอะไรขึ้น เช่น เมื่อมีการคลิกที่ปุ่ม เราจะให้แสดงข้อความ หรือจะให้โปรแกรมทำสิ่งหนึ่งสิ่งใด ก็ได้ ตามที่ต้องการ

กิจกรรมที่ 1 (_click_me.mdb)

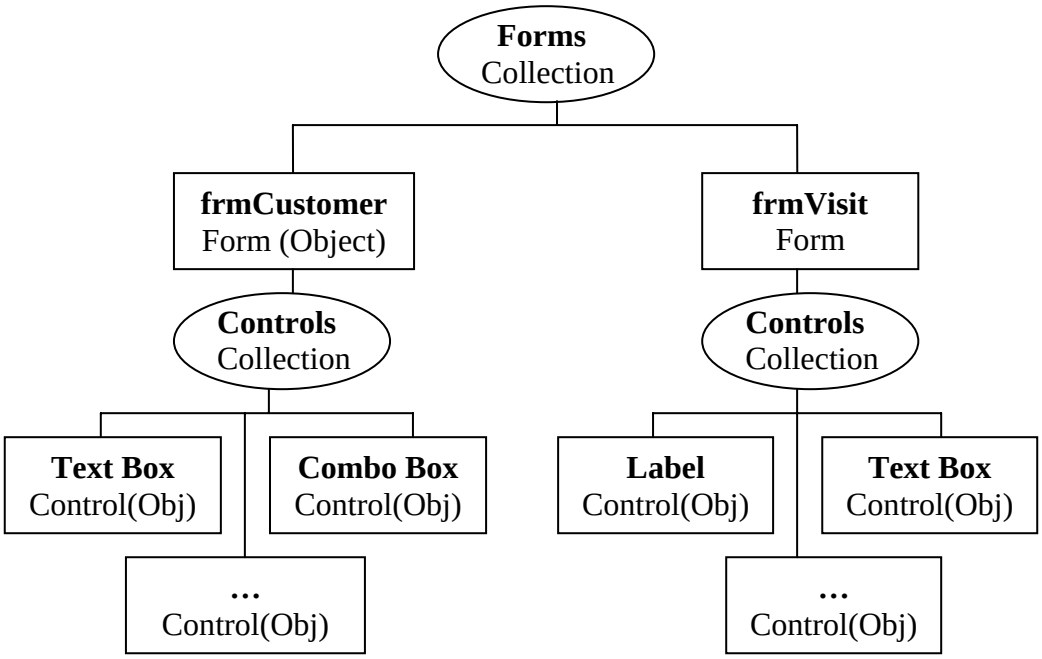
ให้ สร้างฟอร์ม 1 ฟอร์ม ประกอบด้วย ปุ่ม (control button) 2 ปุ่ม เมื่อคลิกที่ปุ่มที่ 1 ให้เกิดข้อความ คุณคลิกปุ่มที่ 1 และเมื่อคลิกอีกปุ่ม ให้มีข้อความว่า คุณคลิกปุ่มที่ 2

วิธีการ ให้ใช้ MsgBox (“.....”) เพื่อแสดงข้อความ จาก event on click ของปุ่ม

Objects และ Collections

ใน Access วัตถุหรือ Object ส่วนใหญ่จะรวมกันอยู่เป็นหมู่ ๆ เราเรียกแต่ละหมู่ว่าเป็น collection เช่น ฐานข้อมูลจะประกอบไปด้วย collection ของ ฟอร์ม ซึ่งจะมีอยู่หลายฟอร์ม โดยที่แต่ละฟอร์ม ก็คือ Object หรือเป็นวัตถุ ๆ หนึ่ง ซึ่ง ก็จะประกอบไปด้วย collection ของ controls ต่าง ๆ ที่อยู่บนฟอร์มนั้น ๆ เช่น ปุ่มส่งข้อมูล (Command Button) ช่องสำหรับกรอกข้อมูล (Text Box) กรอบข้อความ (Label) และ Combo Box เป็นต้น

ตัวอย่าง Form Collection ของฐานข้อมูล



การอ้างถึง Objects

การจัดการข้อมูลต่าง ๆ โดยใช้ VBA จะต้องมีการอ้างถึง objects ต่าง ๆ ให้ถูกต้อง จึงจะสามารถทำการเปลี่ยนแปลงแก้ไขได้ การอ้างอิง เราใช้เครื่องหมาย ! และเครื่องหมาย จุด . โดยมีหลัก ดังนี้

1. ใช้เครื่องหมาย ตกใจ ! คั่น ระหว่าง collection กับ object ดังนี้
   collection_name!object_name
2. ใช้เครื่องหมาย จุด . คั่น ระหว่าง Object และ Collection ดังนี้
   object_name.collection_name

การอ้างถึง Form หรือ Report

สมมุติในโปรแกรมฐานข้อมูลของเรา มีฟอร์ม ชื่อ frmCustomer และ มีแบบรายงาน หรือ report ชื่อ rptCustomer Report เราจะอ้างถึง ดังนี้

การอ้างถึงฟอร์ม frmCustomer จะเริ่มจาก Application Object ดังนี้
Application.Forms!frmCustomer
การอ้างถึง รายงาน อ้างดังนี้
Application.Reports![rptCustomer Report]

ข้อสังเกต

- การอ้างถึงทั้งฟอร์มและรายงาน ต้องเป็นการอ้างถึงในขณะที่กำลังเปิดใช้งานอยู่

- ถ้า ชื่อของ object นั้น มีช่องว่าง จะต้องใช้เครื่องหมาย [ ] มาคร่อมชื่อ ดังตัวอย่าง
- เราสามารถอ้างอิงแบบสั้น โดยละ Application Object เพราะโปรแกรม Access รู้อยู่แล้ว โดยอ้างอิง ดังนี้

  Forms!frmCustomer และ
  Reports![rptCustomer Report]

การอ้างถึงคุณสมบัติของ Form หรือ Report

เราใช้เครื่องหมายจุด . คั่นระหว่าง object และ คุณสมบัติ (property) ของ object นั้น ๆ เช่น ต้องการทราบว่า แหล่งข้อมูล หรือ record source ของฟอร์ม ที่ชื่อ frmCustomer คืออะไร เราสามารถเรียกได้

Forms!frmCustomer.recordSource

การอ้างถึงคุณสมบัติของ Control ที่อยู่บน Form หรือ Report

เราใช้เครื่องหมายจุด . คั่นระหว่าง control และ คุณสมบัติ (property) ของ control นั้น ๆ เช่น ต้องการทราบว่า ในช่องกรอกข้อมูล ที่ชื่อ txtCustName บนฟอร์ม frmCustomer ในขณะนั้นมีข้อความอะไรปรากฏอยู่ เรา ต้องขอดู value ซึ่งเป็น property ของ control นี้ ดังนี้

Application.Forms!frmCustomer.Controls!txtCustName.value หรือ
Forms!frmCustomer!txtCustName.value

กิจกรรมที่ 2 (_find_me.mdb)

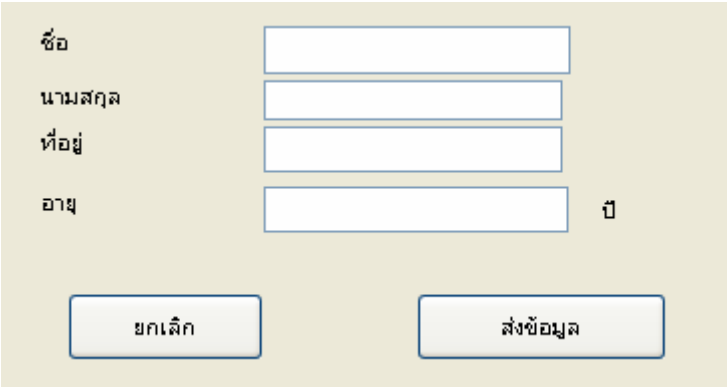
ให้สร้างฟอร์ม ชื่อ frmMyForm และนำ Text Box และ Command Button มาวางไว้ ตั้งชื่อ Text Box ว่า txtName และชื่อปุ่มว่า cmdOK ที่ปุ่มเขียนว่า OK พร้อมทั้งพิมพ์ชื่อของท่านลงใน Text Box จากนั้นให้เปิด Immediate Window (Ctrl+G) และ หาสิ่งต่อไปนี้

1. ชื่อของ Text Box
2. ข้อความที่อยู่ใน Text Box
3. ข้อความที่อยู่บนปุ่มของ Command Button

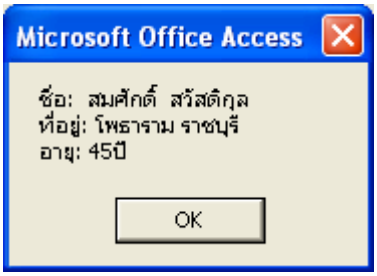
```
print forms!frmMyForm!txtName.name
print forms!frmMyForm!txtName.value
print forms!frmMyForm!cmdOK.caption
```

กิจกรรมที่ 3 (_dataEntry.mdb)

ให้สร้างฟอร์ม ชื่อ frmDataEntry และนำ Control มาวางไว้ ดังภาพ



เมื่อคลิกปุ่ม ส่งข้อมูล ให้ปรากฏข้อความสรุปข้อมูลดังภาพ เมื่อคลิกปุ่ม Reset ให้ลบข้อความที่กรอกทั้งหมดออก



วิธีทำ

1. ตั้งชื่อ control บนฟอร์ม ดังนี้ txtName, txtLastName, txtAddress, txtAge, cmdSend และ cmdReset

2. เขียน โค้ดที่ on click ของปุ่ม cmdSend ดังนี้

```
Private Sub cmdSend_Click()
Dim name As String
Dim lastName As String
Dim address As String
Dim age As String
Dim message As String

message = "ชื่อ:  " & Me.txtName & "  " & Me.txtLastName & vbCrLf
                ‘ Me หมายถึงแบบฟอร์มที่กำลังใช้งานอยู่นี้ หรือ frmDataEntry
                ‘ vbCrLf เป็นเครื่องหมายขึ้นบรรทัดใหม่
message = message & "นามสกุล: " & Me.txtAddress & vbCrLf
message = message & "อายุ: " & Me.txtAge & "ปี"
MsgBox (message)

End Sub
```

3. เขียนโค้ดที่ on click ของปุ่ม cmdReset ดังนี้

```
Private Sub cmdReset_Click()
   Me.txtName = ""
   Me.txtLastName = ""
   Me.txtAddress = ""
   Me.txtAge = ""
End Sub
```

การเรียกค่า จากฟอร์มที่เปิดอยู่

การเรียก จาก Control ของฟอร์มที่เปิดอยู่ สามารถทำได้ โดยการอ้างอิง Form Collection และ บอกชื่อฟอร์ม ไล่ลงมาตามลำดับ จนถึง Control ที่ต้องการ

กิจกรรมที่ 4

1. ให้ดูตัวอย่างการเรียกค่าจากฟอร์ม ในไฟล์ _different_form.mdb
2. เพิ่มเติมข้อมูลในฟอร์มที่ 1 และเมื่อคลิกปุ่มเปิดฟอร์มที่ 2 ให้ข้อมูลนั้นมาปรากฏอยู่ในฟอร์มที่2 ทันที

การสร้าง Procedure

Procedure คือกลุ่มคำสั่งที่อยู่รวมกันเพื่อทำงานอย่างใดอย่างหนึ่ง ตั้งแต่การทำงานอย่างง่าย เช่นการเปิด หรือปิดฟอร์ม ไปจนกระทั่งซับซ้อนมาก ๆ เช่น การจัดการข้อมูล หรือจัดเก็บข้อมูลลงฐานข้อมูล เป็นต้น Procedure สามารถรับข้อมูลจากภายนอก เพื่อนำมาจัดการอย่างใดอย่างหนึ่ง ได้

ประเภทของ Procedures ที่ควรทราบ

1. Sub Procedures เป็นการทำงานของกลุ่มคำสั่ง โดยที่ไม่มีการส่งค่าคืนออกมา ส่วนใหญ่ใช้ Sub Procudures เพื่อรับการเหตุการณ์ หรือ events ของ ฟอร์ม หรือ รายงาน หรือ ของ control ต่าง ๆ รูปแบบของ Sub procedures มีดังนี้

```
Private Sub ชื่อ  ([ค่าที่ส่งเข้าไปใน Sub])
        คำสั่ง
        คำสั่ง
        ...
End Sub
```

Sub Procedure ที่ใช้กับ event ของ ฟอร์ม หรือ รายงาน หรือ ของ Control ต่าง ๆ เราเรียกว่าเป็น event procedure โดยจะมีคำว่า Form หรือ Report หรือชื่อของ control และต่อด้วย event นั้น ๆ เช่น

Private Sub Form_Load() หรือ
Private Sub cmdOK_Click()

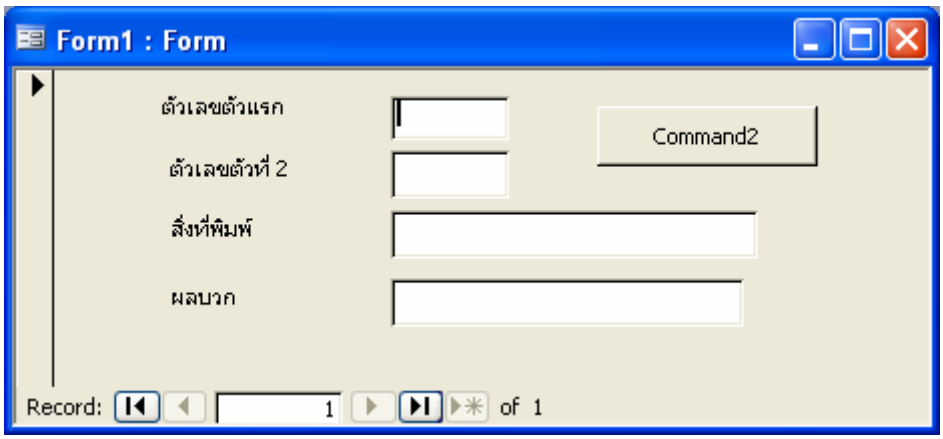
2. Function Procedures เป็น Procedure ที่คืนค่าอย่างใดอย่างหนึ่งออกมาจาก Procedure เราใช้ Function สำหรับการกระทำที่ทำซ้ำ ๆ กัน แต่มีการเปลี่ยนแปลงข้อมูล เราสามารถสร้าง Function ขึ้นใช้เอง Function มักจะมีการรับค่าจากภายนอก เพื่อนำไปประมวลผลบางอย่าง และส่งค่า คืนกลับออกไป รูปแบบของ Function มีดังนี้

```
Private Function_name  ([ค่าที่ส่งเข้าไปใน Sub]) ประเภทข้อมูลที่ส่งออกมา
        คำสั่ง
        คำสั่ง
        ...
        Function_name = expression
End Function
```

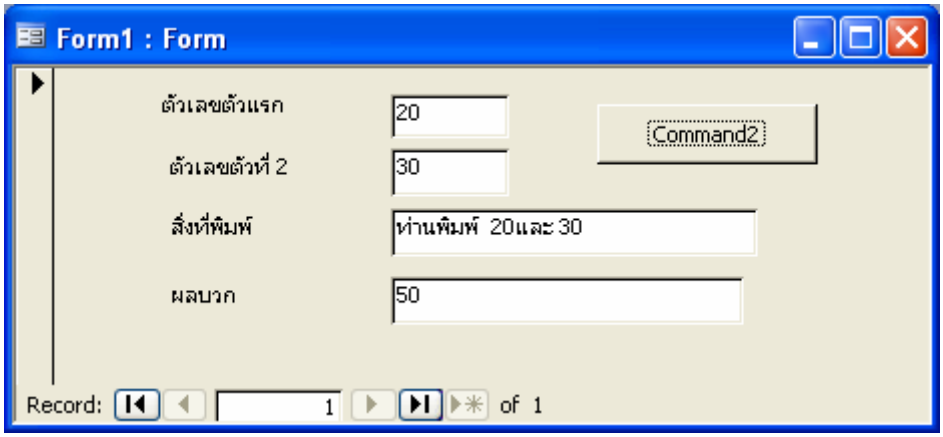
ตัวอย่าง Private Function addit(firstNum AS integer, secondNum AS integer) AS Integer

กิจกรรมที่ 5 (cust_sub_func_expl.mdb)

ให้สร้างฟอร์ม ชื่อform1 และนำ Control มาวางไว้ ดังภาพ



ให้สร้าง Sub และ Function เพื่อรับค่า และเมื่อกรอกตัวเลข ในช่อง ตัวเลขตัวแรก และตัวเลขตัวที่ 2 จากนั้นเมื่อกดปุ่ม Command2 แล้วให้นำค่ามาแสดง ดังรูป



วิธีทำ

สร้าง Sub และ function ตามข้างล่างนี้

```
Private Sub Command2_Click()
Dim thisR As Long
Dim fNum As Long
Dim sNum As Long
 fNum = Me.Text0
 sNum = Me.Text3

 putText "ท่านพิมพ์ "    'เวลาเรียก sub ที่สร้างเอง อย่าใส่เครื่องหมาย วงเล็บคร่อม parameter
                          ' ถ้าเป็นตัวเลขก็ใส่เฉย ๆ เช่น  addme 3, 5
                          'มิฉะนั้น จะนึกว่าเป็น function ซึ่งต้องมีการรับค่า เพราะจะส่งค่าคืนมา
                          ' จึงถามหาเครื่องหมายเท่ากับ

thisR = addThem(fNum, sNum)
Me.Text7 = thisR
End Sub

Function addThem(first As Long, second As Long) As Long
Dim result As Long
result = first + second
addThem = result
End Function

Sub putText(thisText As String)
```

```
Dim f, s As Long
f = Me.Text0
s = Me.Text3
Me.Text5 = thisText & f & "และ " & s
End Sub
```

กิจกรรมที่ 6 (_odd_even.mdb)

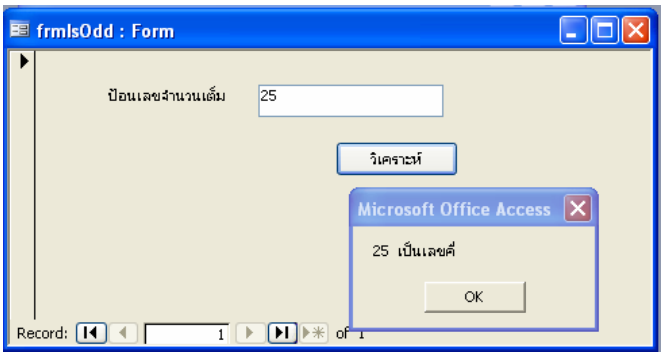
ให้สร้างแบบฟอร์ม เพื่อรับค่าตัวเลขที่ป้อนเข้ามา เมื่อกดปุ่มให้บอกว่า ค่านั้น เป็นเลขคู่ หรือเลขคี่ โดยให้สร้าง Function เพื่อตรวจสอบ และแสดงผล ผ่านทาง MsgBox

สิ่งที่ต้องรู้

1. เลขคู่ คือ เลขที่ 2 หารลงตัวเหลือเศษเป็น 0 และ เลขคี่ คือ เลขที่ 2 หาร เหลือเศษ 1
2. เครื่องหมายหาร ได้ผลลัพธ์ คือเศษที่เหลือจากการหาร เครื่องหมายนั้น ได้แก่ mod เช่น

20 mod 6 = 2

เฉลยหน้าจอ



เรื่อง Control Structure

Control Structure เป็นโครงสร้างในการควบคุมการทำงานของโปรแกรม เช่น มีการกำหนดเงื่อนไข การควบคุมการทำงานซ้ำ เป็นต้น การควบคุมเหล่านี้ในโปรแกรม VBA ใช้โครงสร้างเช่นเดียวกับ Visual Basic โครงสร้างที่น่าสนใจ มีดังนี้

1. การกำหนดเงื่อนไข

1.1 การใช้ if … then … else

เป็นการกำหนดเงื่อนไขการทำงาน ถ้าเงื่อนไขเป็นจริง ให้ทำอะไร ถ้าไม่เป็นจริงให้ทำอะไร ส่วนของ else จะมีหรือไม่มีก็ได้ ถ้าเราไม่สนใจว่า ถ้าไม่เป็นไปตามเงื่อนไขที่กำหนด จะให้ทำอะไร ไม่จำเป็นต้องมี else โครงสร้างการใช้ มีดังนี้

รูปแบบที่ 1 (บรรทัดเดียว)

If (เงื่อนไข) then คำสั่ง [else คำสั่ง]

(เครื่องหมาย [และ] แสดงว่าสิ่งที่อยู่ข้างใน จะมีหรือไม่มี ก็ได้ เครื่องหมายนี้ไม่ได้เป็นส่วนหนึ่งของโปรแกรม)

รูปแบบที่ 2 (หลายบรรทัด)

```
If (เงื่อนไข) Then
        คำสั่ง
        คำสั่ง
        ......
[Else
        คำสั่ง
        คำสั่ง
        .........]
End If
```

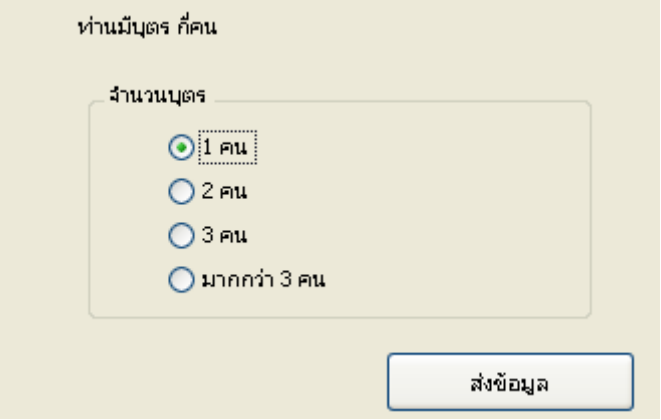
รูปแบบที่ 3 เป็นการกำหนดเงื่อนไขซ้อนกันหลายเงื่อนไข

```
If (เงื่อนไข) Then
        คำสั่ง
        คำสั่ง
        ......
[ElseIf (เงื่อนไข) Then
        คำสั่ง
        ..... ]
[Else
        คำสั่ง
        คำสั่ง
        .........]
End If
```

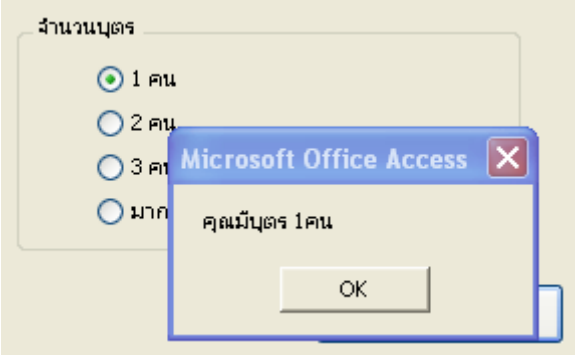
การตรวจสอบเงื่อนไขในลักษณะนี้ จะตรวจสอบทีละเงื่อนไขตามลำดับ ถ้าเงื่อนไขไหนเป็นจริง ก็จะทำคำสั่ง และจะออกจากการตรวจสอบ ถ้าไม่มีเงื่อนไขใดเป็นจริงเลย ก็จะทำตามคำสั่ง ในส่วนของ Else ดังนั้น จึงควรพิจารณาดูก่อนว่า เงื่อนไขใดมีแนวโน้มที่จะเป็นจริงมากที่สุด ก็ให้ตรวจสอบก่อน จะทำให้โปรแกรมมีประสิทธิภาพดีขึ้น

กิจกรรมที่ 7 (_if_exercise.mdb)

ให้สร้างฟอร์มใหม่ ดังภาพข้างล่าง เมื่อผู้ใช้เลือก radio button แล้วคลิกปุ่มส่งข้อมูล ให้บอกว่า เขาคลิกเลือกอะไร



หน้าจอ เมื่อคลิกปุ่มส่งข้อมูลแล้ว



วิธีการทำ

ที่ปุ่ม on click ของปุ่ม ส่งข้อมูล ให้เขียน code ดังนี้

```
Dim numOfChildren As Byte
numOfChildren = Me.frmNumOfChildren.Value

If numOfChildren = 1 Then
   MsgBox ("คุณมีบุตร 1คน")
ElseIf numOfChildren = 2 Then
   MsgBox ("คุณมีบุตร 2 คน")
ElseIf numOfChildren = 3 Then
   MsgBox ("คุณมีบุตร 3 คน")
Else
```

```
   MsgBox ("คุณมีบุตร มากกว่า 3 คน")
End If
End Sub
```

1.2 การใช้ case

ในกรณีที่ต้องการตรวจสอบหลายเงื่อนไข อาจจะไม่ค่อยสะดวกนัก ในการใช้ if แต่ถ้าใช้ case จะสะดวกกว่ามาก รูปแบบมีดังนี้

รูปแบบที่ 1

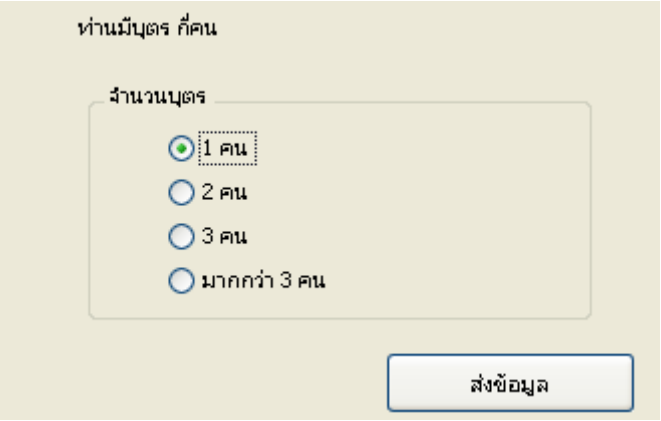
```
Select Case  ตัวแปรที่ต้องการทดสอบ
        Case   ค่าที่ต้องการตรวจสอบว่าเป็นจริงหรือไม่:  คำสั่ง
        Case   ค่าที่ต้องการตรวจสอบว่าเป็นจริงหรือไม่:  คำสั่ง
        ...
        Case  Else:  คำสั่ง
End Select
```

รูปแบบที่ 2

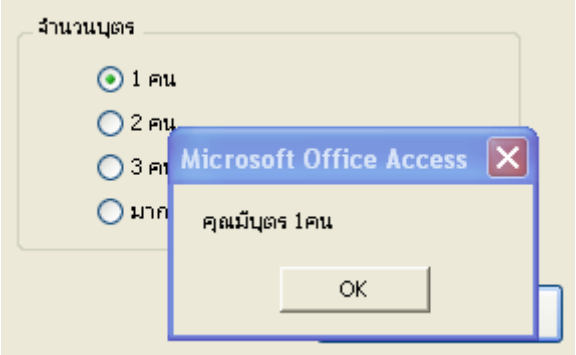
```
Select Cast  ตัวแปรที่ต้องการทดสอบ
        Case   ค่าที่ต้องการตรวจสอบว่าเป็นจริงหรือไม่
                คำสั่ง
                คำสั่ง
                ...
        Case  ค่าที่ต้องการตรวจสอบว่าเป็นจริงหรือไม่
                คำสั่ง
                คำสั่ง
                ...
        ...
        [Case Else
                คำสั่ง
                คำสั่ง
                ... ]
        End Select
```

กิจกรรมที่ 8 (_case_exercise.mdb)

ให้สร้างฟอร์มใหม่ ดังภาพข้างล่าง เมื่อผู้ใช้เลือก radio button แล้วคลิกปุ่มส่งข้อมูล ให้บอกว่า เขาคลิกเลือกอะไร การตรวจสอบเงื่อนไข ให้ใช้ Select Case



หน้าจอ เมื่อคลิกปุ่มส่งข้อมูลแล้ว



วิธีการทำ

ที่ event on click ของปุ่ม ส่งข้อมูล ให้เขียน code ดังนี้

```
Dim numOfChildren As Byte
numOfChildren = Me.frmNumOfChildren.Value
Select Case numOfChildren
   Case 1: MsgBox ("คุณมีบุตร 1  คน")
   Case 2: MsgBox ("คุณมีบุตร 2  คน")
   Case 3: MsgBox ("คุณมีบุตร 3  คน")
   Case Else: MsgBox ("คุณมีบุตร มากกว่า 3 คน")
End Select
End Sub
```

2. การวนซ้ำ

2.1 การใช้ for …. Next

ในกรณีที่เรารู้ว่า จะต้องทำสิ่งที่ซ้ำ ๆ กัน จำนวนกี่ครั้ง เราจะใช้รูปแบบการวนซ้ำในลักษณะ for …..Next เช่นต้องการพิมพ์เลข 1 ถึง 5 เป็นต้น รูปแบบของ for … next มีดังนี้

```
For counter = start To end [Step increment]
        คำสั่ง
        คำสั่ง
        ...
Next [counter]
```

ตัวแปร counter เราจะใช้เป็นเลขสำหรับนับวนจำนวนครั้ง

ตัวแปร start เป็นตัวแปรสำหรับเริ่มการนับ ที่ตัวเลขเท่าไร

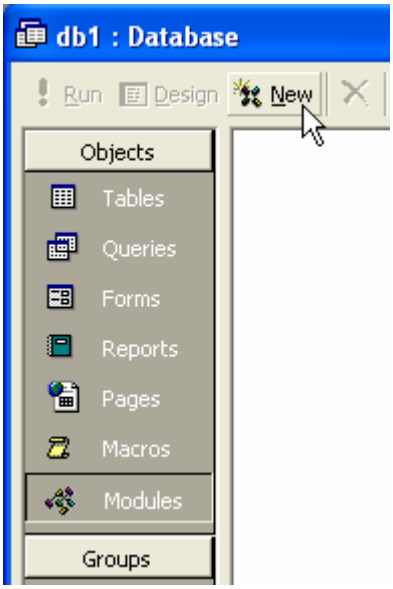
ตัวแปร end เป็นตัวแปรสิ้นสุดการวนรอบ

สำหรับ step increment จะเป็นตัวเลขที่บอกว่าจะให้เพิ่มทีละเท่าไร ถ้าไม่ระบุ จะเพิ่มขึ้นทีละ 1

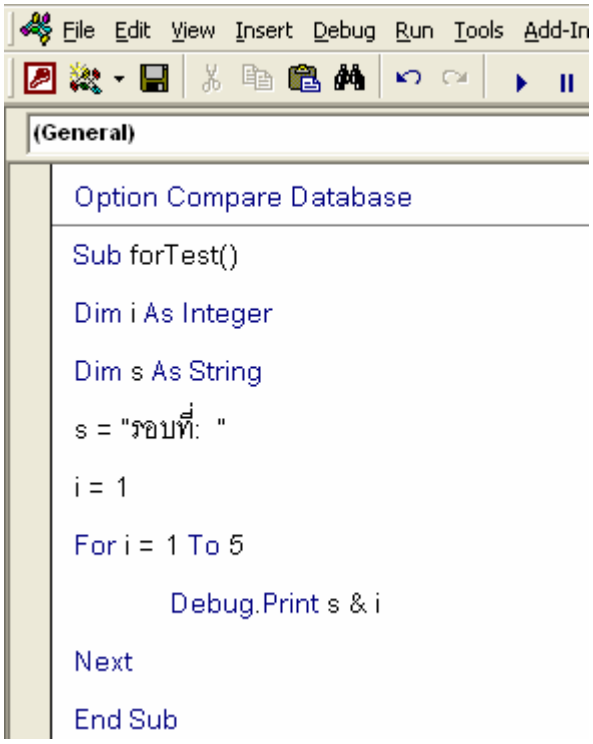
<u>การทดสอบ การใช้ For</u> (_for_next_exercise.mdb)

1. สร้าง Module ใหม่ ดังนี้

1.1 เปิดโปรแกรมใหม่ คลิกที่ Module และคลิก สร้าง หรือ New



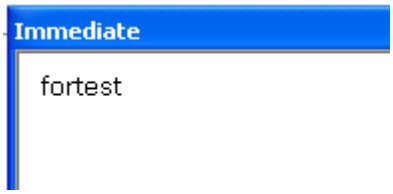
1.2 พิมพ์ข้อความต่อไปนี้



2. ทดสอบการใช้ For โดยการเปิด Immediate window ดังนี้

2.1 กด Ctrl + G หรือ ในขณะที่อยู่ใน Module คลิกที่ View > Immediate Window

2.2 พิมพ์ forTest แล้วกด Enter



2.3 จะเห็นข้อความ แสดงการวนรอบ

กิจกรรมที่ 9 (_for_next_exercise02.mdb)

ให้เขียน Subroutine ใน module และตรวจสอบ ใน Immediate Window ให้ได้ตามภาพข้างล่างนี้

2.2 การใช้ Do while ……..

โปรแกรมจะทำงาน ในขณะที่เงื่อนไขบางอย่างเป็นจริง หรือเป็นเท็จ เช่น ให้อ่านข้อมูลจากไฟล์ จนกระทั่งหมดไฟล์ หรือ ถึงจุดสิ้นสุดของไฟล์ หรือ End Of File ในลักษณะนี้ เราจะเขียนเป็นคำสั่งว่า

Do while NOT EndOfFile

รูปแบบการใช้งาน มีดังนี้

```
Do While ……
        คำสั่ง
        คำสั่ง
        ...
Loop
```

การตรวจสอบการใช้ Do While

1 สร้าง Sub ใหม่ ใน Module

2. เขียน Code ดังต่อไปนี้

```
Sub do_while_Test()
Dim Round As Integer
Dim s As String
Dim intDone As Integer
Dim EndOfFile As Boolean

s = "รอบที่ : "

Round = 1
EndOfFile = False

Do While Not EndOfFile
   If Round >= 11 Then
        Debug.Print "จบแล้ว จำนวน " & Round - 1 & "  รอบ"
        EndOfFile = True
   Else
        Debug.Print s & Round
        Round = Round + 1
   End If
Loop

End Sub
```

3. เปิด Immediate Window โดยกด Ctrl + G

4. เรียก Sub ที่สร้างขึ้น โดยพิมพ์ do_while_Test แล้วกด Enter

เรื่องของ DAO (Data Access Object)

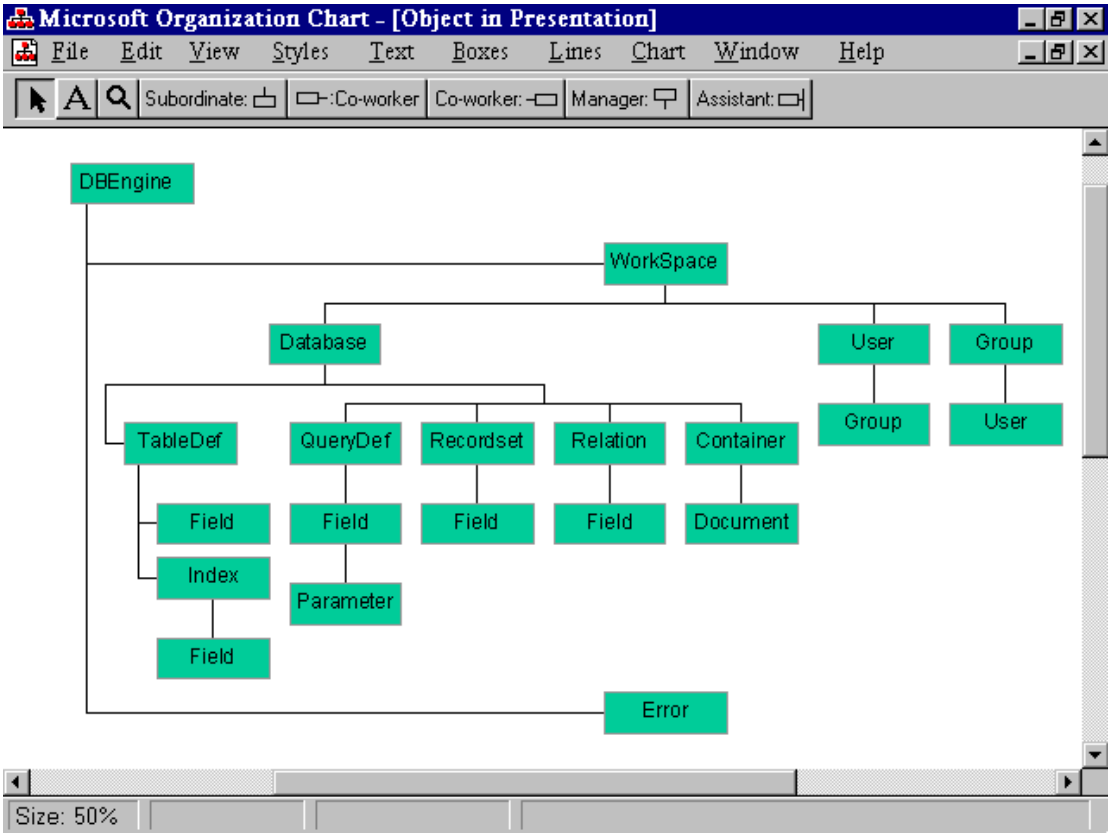
ฐานข้อมูลของ Access คือ Jet Database Engine: (Jet ย่อมาจาก Joint Engine Technology) ซึ่งเป็นหัวใจสำคัญที่เก็บรักษาข้อมูลต่าง ๆ ตัว Access เป็นเพียงการสร้างหน้าจอเพื่อรับข้อมูลเท่านั้น เราใช้ VBA เพื่อจัดการกับวัตถุของ Access เช่น แบบฟอร์ม และรายงาน เป็นต้น สำหรับ ตัวเชื่อมระหว่าง Jet Database Engine และ Access ก็คือ DAO หรือ Data Access Object ซึ่งมีสภาพเป็น วัตถุ ๆ หนึ่ง ประกอบขึ้นด้วย วัตถุย่อย ๆ อีกหลายวัตถุ

ปัจจุบัน Microsoft ได้จัดทำ ADO (ActiveX Data Objects) แต่ในการอบรมครั้งนี้ จะเลือกใช้ DAO เนื่องจาก DAO ได้ถูกออกแบบมาใช้กับ Access และ Jet Database ซึ่งมีการเชื่อมต่อกันอย่างเหนียวแน่น และมีประสิทธิภาพ อีกทั้งยังรวดเร็วกว่าการต่อเชื่อมผ่าน ADO อย่างไรก็ตาม ถ้าในอนาคต จะมีการเลือกใช้ฐานข้อมูลจากภายนอก ที่ไม่ใช่ Jet Database แล้วละก็ ADO จะกลายเป็นตัวเลือกแทนที่ DAO ทันที

MS Access 2003 จะมี DAO 3.6 มาพร้อมด้วยแล้ว DAO 3.6 ออกแบบมาให้ใช้กับ Jet 4.0 Database engine อย่างไรก็ตาม ก่อนที่จะมีการเรียกใช้งานได้ เราต้องเพิ่ม DAO 3.6 Object Library เสียก่อน โดยการเปิดหน้าจอ Code แล้วไปที่ Tools > References แล้วเลือก Microsoft DAO 3.6 Library จึงจะสามารถใช้งานได้

Database Access Object Model

DAO Model มีแผนผัง ดังนี้



จากแผนผัง จะเห็นว่า DAO Model ประกอบด้วย DBEngine หรือ Jet Database อยู่บนสุด และถัดลงมา จะประกอบไปด้วย Objects ต่าง ๆ ในที่นี้จะกล่าวถึงเฉพาะบาง Objects ดังนี้

Workspaces

Workspaces collection ประกอบไปด้วย Workspace objects ใน Workspace object แต่ละตัวเป็นขอบเขตการทำงานของผู้ใช้แต่ละคน การกำหนดสิทธิและความปลอดภัยของแต่ละคน มีอยู่ใน Workspace ของแต่ละคน

Databases

Databases Collections ประกอบด้วย ฐานข้อมูลต่าง ๆ หลายฐานข้อมูล ใน Workspaces ของผู้ใช้นั้น ๆ ฐานข้อมูลอาจจะเป็น Jet Databases หรือ ฐานข้อมูลภายนอก ก็ได้ Database Object แต่ละตัว ก็คือฐานข้อมูลแต่ละฐาน ถ้าต้องการอ้างถึง database ที่ใช้ในปัจจุบันขณะนั้น เราสามารถใช้ฟังชั่น CurrentDB() แทนได้ ดังนี้

```
Sub UseCurrentDBFunc()
    Dim db As DAO.DATABASE
    Set db = CurrentDb()
    Debug.Print db.Version
End Sub
```

ถ้าไม่ใช้ ฟังก์ชั่น currentDB() จะต้องอ้างอิง ดังนี้

```
Sub ReferToCurrentDB()
   Dim ws As Workspace
   Dim db As DATABASE
   Set ws = DBEngine(0)
   Set db = ws.OpenDatabase("tpo")   'tpo เป็นชื่อ database (tpo.mdb)
   Debug.Print db.Version
End Sub
```

TableDefs

TableDefs collect ประกอบไปด้วยตารางต่าง ๆ ในฐานข้อมูลนั้น ๆ ทั้งที่กำลังเปิดใช้งานอยู่ และทั้งที่ไม่ได้เปิด TableDefs จะบรรจุรายละเอียดข้อมูลของตารางแต่ละตาราง

QueryDefs

QueryDefs collect ประกอบไปด้วย queries .ในฐานข้อมูลนั้น ๆ Code ข้างล่าง จะพิมพ์รายชื่อ queries และ คำสั่ง SQL ของ Query นั้น ๆ

```
Sub EnumerateQueries()
   Dim db As DAO.DATABASE
   Dim qry As DAO.QueryDef
   Set db = CurrentDb
   For Each qry In db.QueryDefs
      Debug.Print qry.Name
      Debug.Print qry.SQL
```

```
    Next qry
End Sub
```

Fields

Field collection มีอยู่ใน TableDef, QueryDef และ Recordset objects การหารายละเอียดของ Field สามารถทำได้ โดยการอ้างอิงจาก TableDef, QueryDef และ Recordset objects เช่น

```
Sub EnumerateFields()

    Dim db As DAO.DATABASE
    Dim tbl As DAO.TableDef
    Dim fld As DAO.Field
    Set db = CurrentDb
    For Each tbl In db.TableDefs
        For Each fld In tbl.Fields
            Debug.Print fld.Name
            Debug.Print fld.Type
        Next fld
    Next tbl
End Sub
```

Recordsets

Recordset Object เป็นกลุ่มของ Records ที่มาจากตารางต่าง ๆ Recordset collection จะประกอบไปด้วย Recordset Objects ที่กำลังเปิดอยู่ ในฐานข้อมูลปัจจุบัน

Recordsets แบ่งออกเป็น 3 ประเภท คือ

1. Dynasets เป็น Recordsets ที่ชี้ไปยังตารางต่าง ๆ ในฐานข้อมูล ทำให้สามารถดึงข้อมูลในตารางมาแสดง และเปลี่ยนแปลงแก้ไขข้อมูลได้ Dynasets สามารถเรียกข้อมูลจากฐานข้อมูลอื่น เช่น Microsoft SQL Server, FoxPro, Paradox, และ dBASE เป็นต้น

Dynaset มีลักษณะเป็น Dynamic หมายความว่า การเปลี่ยนแปลงที่กระทำจะไปเปลี่ยนแปลงข้อมูลในตารางข้อมูลทันที และในทางกลับกัน การเปลี่ยนแปลงข้อมูลในตารางโดยผู้ใช้คนอื่น ก็จะเปลี่ยนแปลงใน Dynasets ทันทีเช่นเดียวกัน Recordsets ชนิดนี้มีความยืดหยุ่นมากที่สุด

2. Snapshots เป็น Recordsets ที่คัดลอกข้อมูลจากตาราง ณ เวลานั้นมาใช้งาน การเปลี่ยนแปลงข้อมูลในตาราง โดยผู้ใช้คนอื่น หลังจากที่สร้าง Recordsets ชนิดนี้แล้ว จะไม่มีการปรับปรุง Recordsets ที่เรากำลังใช้งานอยู่ การใช้งาน ถ้าข้อมูลไม่มากนัก (ต่ำกว่า 500 records) ควรกำหนดเป็น Snapshots

3. Table เป็น Recordsets ที่ใช้กับตารางของ Microsoft Access หรือ Jet Database Engine เพื่อจัดการกับข้อมูลในตารางโดยตรง

การเลือกใช้ Recordsets ชนิดใดขึ้นอยู่กับลักษณะการใช้ เช่น ถ้าต้องการความเร็วสูง และสามารถที่จะเอา records ทั้งหมดมาได้ ควรเลือกเป็น table แต่ถ้าต้องการนำผลของ query มาใช้และจำเป็นต้องมีการแก้ไขข้อมูลด้วย ในลักษณะนี้ จะใช้เป็น Dynaset ส่วน Snapshot ควรใช้เมื่อไม่ต้องการ update ข้อมูล และข้อมูลมีจำนวนไม่มากนัก ประมาณไม่เกิน 500 records

การใช้งาน Recordsets

การปรับเปลี่ยนข้อมูลในฐานข้อมูล จะกระทำผ่าน Recortdset Objects โดยการปรับแต่ง property และ เรียก method ต่าง ๆ ของ Recordsets มาใช้งาน property บางอย่าง สามารถแก้ไขได้ขณะ runtime บางอย่าง แก้ไขไม่ได้ ส่วน method เป็นการกระทำที่สามารถ ทำกับ Recordset ได้

การประกาศตัวแปร Recordset

ก่อนที่จะมีการเรียกใช้ Recordset ใด ๆ ต้องมีการกำหนดตัวแปรให้กับ Recrodset เสียก่อน จากนั้นจึงใช้ OpenRecordSet method เพื่อสร้าง Recordset และชี้ไปยัง Recordset ที่ต้องการ เช่น

```
Sub OpenTable()

    Dim dbInfo As DAO.DATABASE
    Dim rstClients As DAO.Recordset
    Set dbInfo = CurrentDb()
    Set rstClients =
dbInfo.OpenRecordset("tblPatient")
    Debug.Print rstClients.Updatable

End Sub
```

(ไฟล์ tpo.mdb)

โค้ดข้างต้น เริ่มจากการกำหนดตัวแปร dbInfo ให้เป็นตัวแปรชนิด Database ถ้าท่านยังไม่ได้อ้างอิงถึง DAO ตรงนี้จะเกิด ERROR พร้อมกับบอกว่าเป็น ประเภทตัวแปร ชนิด Database เป็นชนิดที่ผู้ใช้กำหนดเอง (User-defined type) ซึ่งยังไม่ได้กำหนด ดังนั้นต้องอ้างอิง DAO เสียก่อน โดยไปที่ Tools > References และเลือก Microsoft DAO 3.5 และเพื่อป้องกันไม่ให้เกิดการสับสน เราต้องระบุ Object ให้ชัดเจนว่าเป็น DAO เช่น DAO.Database หรือ DAO.Recordset เป็นต้น มิฉะนั้น อาจจะเกิด Error ได้

กำหนดตัวแปร rstClients เป็นตัวแปรชนิด Recordset และกำหนดให้ dbInfo เป็นฐานข้อมูลที่ใช้ปัจจุบันด้วย ฟังก์ชั่น CurrentDB() แล้วจึงเรียกใช้ method OpenRecordset() เพื่อนำ recordset ที่เป็นข้อมูลตารางของ tblClients มาใส่ไว้ในตัวแปร rstClients

Recordset ที่กำหนดขึ้นนี้ ไม่ได้ระบุประเภท ถ้า recordset ใช้กับตาราง โปรแกรมจะกำหนดให้เป็นชนิด ตาราง ถ้าใช้กับ query จะเป็นประเภท Dynaset แต่ถ้าเราจะกำหนดเองก็สามารถระบุได้ เช่น

```
Sub OpenQuery()
    Dim dbInfo As DAO.Database
    Dim rstClients As DAO.Recordset
    Set dbInfo = CurrentDb()
    Set rstClients =
dbInfo.OpenRecordset("qryVisits", dbOpenSnapshot)
    Debug.Print rstClients.Updatable
End Sub
```

(ไฟล์ tpo.mdb)

หรือ การเปิดตาราง ในลักษณะ Dynaset เช่น

```
Sub OpenDynaset()
    Dim dbInfo As DAO.DATABASE
    Dim rstClients As DAO.Recordset
    Set dbInfo = CurrentDb()
    Set rstClients = dbInfo.OpenRecordset
     ("tblPatient", dbOpenDynaset)
    Debug.Print rstClients.Updatable
End Sub
```

(ไฟล์ tpo.mdb)

ถ้าต้องการสร้าง Recordset ในลักษณะอื่น ก็ใช้ dbOpenTable, dbOpenSnapshot แทน dbOpenDynaset .ในตัวอย่างข้างบน อย่างไรก็ตาม Query จะเปิดได้เฉพาะ Dynaset หรือ Snapshot เท่านั้น

การไปยัง records ต่าง ๆ ใน Recordset

เมื่อสร้าง recordset Object ได้แล้ว ถ้าต้องการวนดูใน Recordset ว่า Record ไหน มีข้อมูลอย่างไร เราต้องระบุให้ถูกต้อง Method ที่ใช้ในการเคลื่อนไปยังตำแหน่งต่าง ๆ ใน Recordset มีดังนี้

MoveFirst	ไปยัง record แรก ของ recordset
MoveLast	ไปยัง record สุดท้าย ของ recordset
MovePrevious	ไปยัง record ก่อนหน้านี้

MoveNext	ไปยัง record ต่อไป
Move[0]	ไปก่อนหน้านี้ หรือหลังจากนี้ เท่ากับจำนวน record ที่ระบุ

ตัวอย่าง

```
Sub RecordsetMovements()
    Dim db As DAO.DATABASE
    Dim rst As DAO.Recordset
    Set db = CurrentDb
    Set rst = db.OpenRecordset("qryVisits", dbOpenDynaset)
    Debug.Print rst!ID
    rst.MoveNext
    Debug.Print rst!ID
    rst.MoveLast
    Debug.Print rst!ID
    rst.MovePrevious
    Debug.Print rst!ID
    rst.MoveFirst
    Debug.Print rst!ID
    rst.Close
End Sub
```

(ไฟล์ tpo.mdb)

ขอบเขตของ Recordset

คุณสมบัติ หรือ property ของ Recordset 2 ตัว คือ BOF และ EOF จะบอกขอบเขตของ Recordset BOF มีค่าเป็นจริง เมื่อตัวชี้เรคคอร์ด (record pointer) อยู่ก่อนหน้า Record ที่ 1 ของ Recordset และ ค่า EOF มีค่าเป็นจริง เมื่อตัวชี้เรคคอร์ด (record pointer) อยู่หลัง Record สุดท้ายของ Recordset

ตัวอย่าง

```
Sub DetermineLimits()
    Dim db As DAO.DATABASE
    Dim rstClients As DAO.Recordset
    Set db = CurrentDb()
    Set rstClients = db.OpenRecordset("tblPatient",
dbOpenSnapshot)
    Do While Not rstClients.EOF
        Debug.Print rstClients![ClientID]
        rstClients.MoveNext
    Loop
    rstClients.Close
End Sub
```

(ไฟล์ tpo.mdb)

ข้อเท็จจริงเกี่ยวกับ BOF และ EOF

- ถ้า Recordset นั้นไม่มีข้อมูล ค่าของ BOF และ EOF จะมีค่าเป็นจริง
- เมื่อเปิด Recordset ที่มีข้อมูลอย่างน้อย 1 Record ค่าของ BOF และ EOF จะมีค่าเป็นเท็จ
- ถ้าอยู่ในตำแหน่ง Record ที่ 1ของ Recordset และใช้ method ถอยหลัง หรือ MovePrevious ค่าของ BOF จะเป็นจริง และถ้าเรียกใช้ method นี้อีกครั้ง จะเกิด runtime error
- ถ้าอยู่ในตำแหน่ง Record สุดท้ายของ Recordset และใช้ method ไปข้างหน้า หรือ MoveNext ค่าของ EOF จะเป็นจริง และถ้าเรียกใช้ method นี้อีกครั้ง จะเกิด runtime error
- ถ้า Recordset มีเพียง Record เดียว และ record นั้นถูกลบออกไป ค่า ของ EOF และ BOF จะยังคงเป็น False จนกว่าจะมีคำสั่งเคลื่อนไปยังที่อื่น

การนับจำนวน Recordset

การนับจำนวน record ใน Recordset เราเรียกดูคุณสมบัติ RecordCount แต่ก่อนจะเรียกจะต้องมีการไปยังตำแหน่งแรกและตำแหน่งสุดท้ายของ Recordset เสียก่อน มิฉะนั้น ค่าที่ได้จะมีค่าเป็น 1 หรือ 0 ซึ่งไม่ถูกต้อง

ตัวอย่าง

```
Sub CountRecords()
    Dim db As DAO.DATABASE
    Dim rstProjects As DAO.Recordset
    Set db = CurrentDb()
    Set rstVisits = db.OpenRecordset("tblVisits",
dbOpenSnapshot)
    Debug.Print rstVisits.RecordCount  'Prints 0 Or
1
    rstProjects.MoveLast
    Debug.Print rstVisits.RecordCount 'Prints an
accurate record Count
    rstVisits.Close
End Sub
```

(ไฟล์ tpo.mdb)

ปัญหาของ RecordCount คือทำงานค่อนข้างช้า และถ้าเป็นระบบเครือข่าย เมื่อมีการลบ Record ก็จะได้ข้อมูลที่ไม่เป็นจริง อย่างไรก็ตาม ประโยชน์อย่างหนึ่ง คือ ใช้ตรวจสอบว่า Record นั้น มีข้อมูลอยู่หรือไม่ เพราะบางครั้ง Recordset ที่ได้ตามเงื่อนไข ไม่มีข้อมูลอยู่เลย จึงต้องตรวจสอบเสียก่อน ก่อนที่จะทำอย่างอื่น

การสร้าง Recordset จากคำสั่ง SQL

เราสามารถสร้าง Recordset โดยใช้คำสั่ง SQL เพื่อกรองข้อมูลจากตาราง นำมาไว้ใน Recordset เพื่อจะได้นำมาดำเนินการอย่างใดอย่างหนึ่ง ได้ ดังตัวอย่างต่อไปนี้

```
Sub sqlDemo()
Dim dbs as DAO.Database
Dim rst as DAO.Recordset
Set dbs = CurrentDb
Set rst = dbs.OpenRecordset("SELECT * FROM
tblPatient Where age >= 32")

If Not rst.EOF Then
        rst.MoveFirst
        Do While Not rst.EOF
            Debug.Print rst!ID & “  “ & rst!name &
“อายุ “ & rst!age
            rst.MoveNext
        Loop
End If
rst.Close
End Sub
```

การเพิ่มข้อมูลในตาราง

การเพิ่มข้อมูลลงตารางทำได้โดยใช้ method AddNew ดังตัวอย่างต่อไปนี้

```
Dim dbs As DAO.Database
Dim rst As DAO.Recordset

Set dbs = CurrentDb
Set rst = dbs.OpenRecordset("SELECT * FROM tblPatient")
 With rst
        ' Edit the Recordset object.
  .AddNew
   !name] = thisName
   ![lastName] = thisLName
   ![age] = thisAge
  .Update      ' Save changes.
 End With
 rst.Close
```

(ดูตัวอย่างใน ไฟล์ tpo_2.mdb)

การลบข้อมูลในตาราง

การลบข้อมูล ใช้ method Delete ของ Recordset Object ดังตัวอย่างต่อไปนี้

```
Sub DeleteData()
```

```
    Dim db As DAO.DATABASE
    Dim rstProjects As DAO.Recordset
    Dim intCounter As Integer
    Set db = CurrentDb
    Set rstProjects = db.OpenRecordset("tblProjectsChange",
dbOpenDynaset)
    intCounter = 0
    Do While Not rstProjects.EOF
        If rstProjects!ProjectTotalEstimate < lngProjEst Then
            rstProjects.Delete
            intCounter = intCounter + 1
        End If
        rstProjects.MoveNext
    Loop

    Debug.Print intCounter & " Customers Deleted"
End Sub
```

การแก้ไขข้อมูลในตาราง

การแก้ไขข้อมูลในตาราง จะต้องหา record ข้อมูลที่ต้องการจะแก้ไข ตามเงื่อนไขที่กำหนดเสียก่อน จากนั้นจึงแก้ไข ลองดูตัวอย่างต่อไปนี้

```
Sub IncreaseEstimate()
    Dim db As DATABASE
    Dim rstProjectst As Recordset
    Dim sSQL As String
    Dim intUpdated As Integer
    Set db = CurrentDb()
    Set rstProjectst = db.OpenRecordset("tblProjectsChange", dbOpenDynaset)
    sSQL = "ProjectTotalEstimate < 30000"
    intUpdated = 0
    rstProjectst.FindFirst sSQL
    Do While Not rstProjectst.NoMatch
        intUpdated = intUpdated + 1
        rstProjectst.Edit
        rstProjectst!ProjectTotalEstimate = rstProjectst!ProjectTotalEstimate _
        * 1.1
        rstProjectst.UPDATE
        rstProjectst.FindNext sSQL
    Loop
    Debug.Print intUpdated & " Records Updated"
    rstProjectst.Close
End Sub
```

แหล่งอ้างอิง

Cardoza, Patricia and others (2004) Access2003 VBA Programer's Refgerence , Wiley Publishing, Inc., Indianapolis.

Prague, Cary and Irwin, Michael (2001) Access2002 Bible, Hungry Mind Inc., New York.

http://cma.zdnet.com/book/masteringaccess/abafmfi.htm Retrieved April 2005.

http://www.functionx.com/access Retrieved January 2005.

http://www.profsr.com/access/accless01.htm Retrieved December 2004.