**学 习 笔 记**

**天津园林建设管理部（TJLAPM）**

**2017-2**

目 录

[**通用函数库** 5](#_Toc501369036)

[1. PinYin()拼音码函数 12](#_Toc501369037)

[2. OpenADORecordset 函数 12](#_Toc501369038)

[3. CheckRequired 函数 12](#_Toc501369039)

[4. HasPermission权限功能 13](#_Toc501369040)

[5.窗体加载事件给组合框赋上行来源并实现智能筛选 14](#_Toc501369041)

[6. 商品编码发生重复时提醒并自动跳转到重复项 14](#_Toc501369042)

[7.ApplyTheme Me    '应用主题颜色 15](#_Toc501369043)

[8.SQLText 函数 15](#_Toc501369044)

[9. 刷新子窗体代码示例\RefreshDataList刷新窗体应用 16](#_Toc501369045)

[10.ExportToExcel 函数 17](#_Toc501369046)

[11. 行号能自动显示序号GetLineNumber([Form]) 18](#_Toc501369047)

[12. 删除数据 18](#_Toc501369048)

[13. 添加新商品"命令按钮 19](#_Toc501369049)

[14. 经办人和制单人改为组合框后，赋上默认值 19](#_Toc501369050)

[15. GetAutoNumber 函数 19](#_Toc501369051)

[16.根据单据类型显示相关的订单号 (级联筛选) 20](#_Toc501369052)

[17.日期选择 20](#_Toc501369053)

[18.根据条件改变子窗体数据源 21](#_Toc501369054)

[19. DatePickerFor 函数 22](#_Toc501369055)

[20.树控件 22](#_Toc501369056)

[21. ClearControlValues 函数 24](#_Toc501369057)

[22. 禁止修改“最近采购日期”的相关设置 24](#_Toc501369058)

[23. Dcount 24](#_Toc501369059)

[24.子窗体数据源改变 25](#_Toc501369060)

[25.窗体打开模式、 OpenArgs传递 25](#_Toc501369061)

[26. vba代码太长，换行 26](#_Toc501369062)

[27.数据刷新 26](#_Toc501369063)

[28. 快速书写SQL语句 26](#_Toc501369064)

[29. CurrentDb.Execute执行sql语句 27](#_Toc501369065)

[30.Enabled属性与Locked属性的区别 28](#_Toc501369066)

[31. SelectAllListBoxItems 函数 29](#_Toc501369067)

[32.LTrim，RTrim，及 Trim 函数示例 29](#_Toc501369068)

[33. 在窗体上显示当前记录和总记录数 29](#_Toc501369069)

[34. For Each in ... next 详解用法 30](#_Toc501369070)

[35. DAO使用书签进行搜索并同步窗体的方法 31](#_Toc501369071)

[36. SelectAllListBoxItems 函数 32](#_Toc501369072)

[37. Dirty 属性 32](#_Toc501369073)

[38.主子窗体计算 33](#_Toc501369074)

[39. VBA文件对话框（Application.FileDialog属性）的常用属性与方法的解释 33](#_Toc501369075)

[40. LoadRecord 新平台 36](#_Toc501369076)

[41.表关系 36](#_Toc501369077)

[42.选择并添加商品 37](#_Toc501369078)

[43.审核按钮 41](#_Toc501369079)

[44.完成按钮 43](#_Toc501369080)

[45. 错误'2448'：您不能将数值指定给这个对象 44](#_Toc501369081)

[46.附件改进型 45](#_Toc501369082)

[47. 输入框焦点变色 45](#_Toc501369083)

[48. 撤消审核 46](#_Toc501369084)

[49. 三种来源的区别 46](#_Toc501369085)

[50. 打开窗体图标显示错误 47](#_Toc501369086)

[51. 打印后标记为已打印 47](#_Toc501369087)

[52. 窗体打开后不加载数据 47](#_Toc501369088)

[53.批量更新拼音码 47](#_Toc501369089)

[54. 数据查询模块 48](#_Toc501369090)

[55. 通用查询如何实现默认当天或当月/本月或默认其它指定值 52](#_Toc501369091)

[56. 指定列表框或组合框中默认值的所在行 54](#_Toc501369092)

[57. 智能筛选条件使用变量 55](#_Toc501369093)

[58. 通用附件模块使用 55](#_Toc501369094)

[59. 树控件报错 56](#_Toc501369095)

**通用函数库**

|  |  |  |  |
| --- | --- | --- | --- |
| 分类 | # | 函数名 | 用途说明 |
| Access对象信息获取 | 1 | GetObjDesc | 取得数据库对象（表、查询、窗体、报表、宏、模块）的说明。 |
| 2 | GetUniqueFieldName | 取得表中的唯一标识字段名，优先取设为主键的字段，如果没有主键，则取第1个存在的设为无重复索引的字段。主键和索引仅限于单字段的情况下。 |
| 3 | GetUnsignedObjectName | 根据表名、查询名，取得一个无标记名称，主要用于去掉应用了命名约定时的一些额外的内容，如tblCustomers，返回Customer。 |
| 4 | LoadFieldList | 将指定表对象、查询对象、或DAO记录集对象中的字段信息，加载到一个ADO记录集中，并将记录集绑定到指定对象。 |
| 5 | LoadRDPObjectList | 加载表、临时表、查询、窗体、报表几种类型的对象名称到一个支持Recordset的对象中（列表框、组合框、 窗体）。对象可以是多种，用+号连接，如rotTable + rotForm，传入的参数中对象只有1种时，加载的记录集只有Name字段，否则会包含Name、Type字段。 |
| Access对象状态判断及控制 | 6 | ExpandRibbon | 最大化/最小化功能区。 |
| 7 | IsChildForm | 判断指定窗体是否以子窗体形式打开。 |
| 8 | IsLoaded | 判断窗体是否已打开。 |
| 9 | NavPannelVisible | 判断导航窗格的可见性。 |
| 10 | ObjectExists | 判断表、查询、窗体等数据库对象是否存在。 |
| 11 | ShowRibbon | 显示/隐藏功能区。 |
| 12 | ShowNavPannel | 显示/隐藏导航窗格。 |
| ADO封装，数据库连接 | 13 | AAvg | DAvg的ADO实现，默认使用到后台数据库的连接实现。 |
| 14 | ACount | DCount的ADO实现，默认使用到后台数据库的连接实现。 |
| 15 | ADORunSQL | 用ADO方式一次运行一个或多个SQL操作查询语句，默认使用到后台数据库的连接。 |
| 16 | ALookup | DLookup的ADO实现，默认使用到后台数据库的连接实现。 |
| 17 | AMax | DMax的ADO实现，默认使用到后台数据库的连接实现。 |
| 18 | AMin | DMin的ADO实现，默认使用到后台数据库的连接实现。 |
| 19 | GetADOConnection | 返回后台数据库的ADO连接。 |
| 20 | GetODBCConnectionString | 取得后台数据库的ODBC连接字串。 |
| 21 | OpenADORecordset | 打开一个ADO记录集，默认使用到后台数据库的连接。 |
| SQL格式化 | 22 | SQLDate | 将一个日期格式化为符合SQL语法的格式。 |
| 23 | SQLText | 将一个字符串格式化为符合SQL语法的格式。 |
| Windows注册表读写 | 24 | DeleteRegKey | 删除注册表键值。 |
| 25 | GetRegKey | 取得注册表键值。 |
| 26 | SaveRegKey | 保存注册表键值。 |
| 参数读写 | 27 | GetDbSetting | 取得自定义数据库属性。 |
| 28 | GetFormProperty | 取得自定义窗体属性。 |
| 29 | GetParameter | 取得本地或全局系统参数的值。 |
| 30 | SaveDbSetting | 保存自定义数据库属性。 |
| 31 | SaveFormProperty | 保存自定义窗体属性。 |
| 32 | SetParameter | 设置本地或全局系统参数。 |
| 窗口控制 | 33 | GetKeyState | 获取指定按键的状态。 |
| 34 | LockWindowUpdate | 锁定窗口，禁止更新。 |
| 35 | MoveWindow | 移动指定窗口（以像素为单位）。 |
| 36 | UpdateWindow | 立即更新窗口。 |
| 对话框 | 37 | DatePickerFor | 在指定控件（一般为文本框）下方显示日历，并返回在日历中点击选择的日期作为控件值。 |
| 38 | GetSaveAsFileName | 显示一个另存为对话框，并返回用户通过对话框选择的另存为文件路径名。 |
| 39 | MsgBoxEx | VBA内置的MsgBox函数的扩展，增加了根据不同的图标自动显示不同的标题，以及指定时间内自动关闭的功能。 |
| 40 | PickColor | 显示一个颜色对话框，并返回用户通过对话框选择的颜色值。 |
| 41 | PickFile | 显示一个文件选择对话框，并返回用户通过对话框选择的一个或多个文件的路径名。 |
| 42 | PickFolder | 显示一个树形结构的目录选择对话框，并返回用户在对话框中选择的文件夹的路径。 |
| 多媒体 | 43 | MessageBeep | 根据不同的消息类型发出声音。 |
| 44 | PlaySound | 播放指定的声音。 |
| 加密/解密 | 45 | DES | 3DES对称加密。 |
| 46 | MD5 | MD5文本摘要计算。 |
| 日期处理 | 47 | FirstDayOfMonth | 返回指定日期所在月份的第一天。 |
| 48 | LastDayOfMonth | 返回指定日期所在月份的最后一天。 |
| 49 | FirstDayOfWeek | 返回指定日期所在周的第一天。 |
| 50 | LastDayOfWeek | 返回指定日期所在周的最后一天。 |
| 51 | GetLunarDate | 将日期转换为中国农历（中文格式或数字格式）。 |
| 52 | LunarToChrist | 将农历日期转换为公历日期。 |
| 上传下载 | 53 | URLDownloadToFile | 根据URL路径从网络下载文件到本地。 |
| 树控件(TreeView) | 54 | CheckAllTreeNodes | 将整个TreeView（或指定节点下）所有节点前面的复选框勾选/取消勾选。 |
| 55 | ExpandAllTreeNodes | 将整个TreeView（或指定节点下）所有节点展开或折叠。 |
| 56 | GetTreeNodeLevel | 取得TreeView中指定节点的层级数。 |
| 57 | TreeNodeExists | 判断TreeView中指定索引或键对应的节点是否存在。 |
| 数据导入导出 | 58 | ExportToExcel | 将当前数据表视图中的数据导出到Excel。 |
| 59 | ImportFromExcel | 从Excel导入数据。 |
| 数据录入辅助 | 60 | CheckRequired | 根据Tag属性中的标记，验证数据编辑窗体中的必填项是否为空，并自动提示。 |
| 61 | CheckTextLength | 检查窗体上的所有设置了标记的输入控件，验证其文本长度是否超过了允许的最大长度，如果有则显示提示消息。 |
| 62 | ClearControlValues | 清空所有控件的值。 |
| 63 | GetLineNumber | 取得窗体记录集中的动态行号。 |
| 64 | RequeryDataObject | 刷新数据，并自动定位到刷新之前的行。 |
| 65 | SelectAllListBoxItems | 全选/取消全选列表框中的项（列表框需要允许全选）。 |
| 数值处理 | 66 | RoundDown | 向下取整，只舍不入。 |
| 67 | RoundOff | 四舍五入。 |
| 68 | RoundUp | 向上取整，只入不舍。 |
| 数组 | 69 | GetArrayItemIndex | 取得指定数组项的索引值。 |
| 文件、目录操作 | 70 | CreateMultiDir | 创建多级目录，不需要像MkDir那样必须一级一级的创建。 |
| 71 | PathFileExists | 判断文件夹或文件是否存在。 |
| 72 | PathFileOperation | 移动、删除、复制、重命名整个文件夹，或者多个文件，主要用于文件的批量操作。 |
| 73 | CAB | 将指定文件进行压缩打包生成 CAB 格式压缩文件。 |
| 74 | UnCAB | 将 CAB 格式压缩文件解压出来另存为指定文件路径名。 |
| 硬件信息获取 | 75 | GeBaseBoardId | 取得主板序列号。 |
| 76 | GetDiskDriveId | 取得硬盘序列号。 |
| 77 | GetIPV4Address | 取得本机的IPV4地址。 |
| 78 | GetIPV6Address | 取得本机的IPV6地址。 |
| 79 | GetMACAddress | 取得网卡的物理地址。 |
| 80 | GetProcessorId | 取得CPU序列号。 |
| 指针形状控制 | 81 | SetCursor | 设置鼠标指针为指定的内置样式。 |
| 82 | SetHandCursor | 设置鼠标指针为手形。 |
| 中文格式转换 | 83 | CnDate | 将日期转换为中文大写日期。 |
| 84 | CnMoney | 将一个货币类型的数字转换为中文大写金额。 |
| 85 | CnNumber | 将阿拉伯数字转换为中文大写数字。 |
| 资源加载 | 86 | LoadLocalLanguage | 加载本地语言。 |
| 87 | LoadPictureToImageList | 加载Images\Icons文件夹中的所有图标到指定ImageList控件。 |
| 88 | LoadString | 根据Key加载本地化字符串。 |
| 89 | RandomPicture | 取得一个存在的随机图片的路径名。 |
| 自动编号 | 90 | ComputerID | 根据CPU、主板、硬盘信息生成一个唯一标识码。 |
| 91 | GetAutoNumber | 根据预定义的规则，生成一个自动文本型编号。 |
| 92 | GetGUID | 生成一个全球唯一标识码。 |
| 字符串处理 | 93 | CountStr | 查询一个字符串在另一个字符串中出现的次数。 |
| 94 | RepeatStr | 将指定字符串进行多次重复。 |
| 其它 | 95 | CompressFile | 将压缩为CAB文件，或从CAB文件中解压出来。使用Windows自带功能，减少依赖性。 |
| 96 | CopyToLocalTable | 将表对象复制为一个本地表对象。 |
| 97 | CreateInstance | 创建一个UMV Support Library.ucl支持库文件中类模块的新实例，仅包括： PopupProgressBar, GeneralQuery, FileSearch, FTPServer。 |
| 98 | HideAllTables | 隐藏/显示所有表。 |
| 99 | IsInternetConnected | 判断是否连接到互联网。 |
| 100 | ReportSheet | 在打印报表时自动画表格。 |
| 101 | ShellExecute | 打开程序、文件、超链接，运行命令行等。 |
| 102 | Sleep | 让程序暂停指定毫秒数。 |
| 103 | WriteOperationLog | 写入操作日志。 |

新增——保存-单击

Dim rst As Object  
Dim strSQL As String  
Dim MaxID As String  
Dim currentID As String

Dim strFrm As String

If IsNull(Me.txtygxm) Then  
  MsgBox "请输入员工姓名!", vbCritical, "提示"  
  Me.txtygxm.SetFocus  
  Exit Sub  
End If  
  
MaxID = DMax("[ygID]", "tblCodeyg")  
currentID = "Y" & Format(Val(Right$(MaxID, 2) + 1), "00")

strSQL = "select \* from tblCodeyg"  
Set rst = CurrentDb.OpenRecordset(strSQL, dbOpenDynaset)  
rst.AddNew  
rst!ygID = currentID  
rst!ygxm = Me.txtygxm  
rst.Update  
rst.Close  
Set rst = Nothing  
Me.txtygxm = Null

strFrm = Form\_frmYg\_sg\_Main!frmChild.SourceObject  
Form\_frmYg\_sg\_Main!frmChild.SourceObject = strFrm   
MsgBox "您录入的数据保存已成功！", vbInformation, "消息"

修改——加载

Dim rst As Object  
Dim strSQL As String  
Dim currentID As String  
currentID = Form\_frmyg\_sg\_list.Form.ygID  
strSQL = "select \* from tblCodeyg where ygID ='" & currentID & "'"  
Set rst = CurrentDb.OpenRecordset(strSQL, dbOpenDynaset)  
rst.MoveFirst  
Me.txtygxm = rst!ygxm  
rst.Close  
Set rst = Nothing

修改——保存-单击

Dim rst As Object  
Dim strSQL As String  
Dim currentID As String

Dim strFrm As string

If IsNull(Me.txtygxm) Then  
MsgBox "员工姓名不允许空缺，请录入相关数据！", vbCritical, "提示"  
Me.txtygxm.SetFocus  
Exit Sub  
End If

currentID = Form\_frmYg\_sg\_List.Form.ygID  
strSQL = "select \* from tblCodeyg where ygID ='" & currentID & "'"  
Set rst = CurrentDb.OpenRecordset(strSQL, dbOpenDynaset)  
rst.MoveFirst  
rst.Edit  
rst!ygxm = Me.txtygxm  
rst.Update  
rst.Close  
Set rst = Nothing  
DoEvents

strFrm = Form\_frmYg\_sg\_Main!frmChild.SourceObject   
Form\_frmYg\_sg\_Main!frmChild.SourceObject =strFrm  
MsgBox "您提交的数据更新已完成！", vbInformation, "消息"  
DoCmd.Close acForm, "frmYg\_sg\_Edit"

删除-单击

Dim strSQL As String  
Dim currentID As String  
currentID = Form\_frmYg\_sg\_List.Form.ygID  
strSQL = "DELETE tblCodeyg.\* FROM tblCodeyg WHERE ygID='" & currentID & "'"  
If MsgBox("您确认要删除当前已选择的记录么？", vbOKCancel, "提示") = vbOK Then  
DoCmd.SetWarnings False  
DoCmd.RunSQL strSQL  
DoCmd.SetWarnings True  
DoEvents  
Form\_frmYg\_sg\_Main!frmChild.SourceObject = Form\_frmYg\_sg\_Main!frmChild.SourceObject  
MsgBox "您所选择的记录已被删除！", vbInformation, "消息"

End If

## 1. PinYin()拼音码函数

Me!拼音码 = PinYin(Me!供应商名称)

## 2. OpenADORecordset 函数

打开并返回一个 ADO 记录集。

语法

OpenADORecordset (RecordSource[, LockType][, Connection])

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | 必需/可选 | **数据类型** | **说明** |
| RecordSource | 必需 | String | 记录集数据源。一般是一个SQL语句，注意该语句需要符合对应的后台数据库的SQL语法，而不能完全使用 Access 的SQL语法。 |
| LockType | 可选 | ADOLockTypeEnum | 记录集锁定类型。如果省略此参数，将默认使用 adLockReadOnly，即返回一个只读记录集。 |
| Connection | 可选 | Variant | ADO连接对象（ADODB.Connection）或连接字符串。只有当需要打开至其它数据库的连接的时候才用到该参数。如果省略此参数，将默认使用 GetADOConnection 函数的返回值连接至后台数据库。 |

返回值

记录集正确打开时返回对应的 ADO 记录集（ADODB.Recordset），打开失败时返回 Nothing。

说明

对 ADO 记录集的打开所做的二次简化封装，只需要一行代码即可实现 ADO 记录集的打开，从而更加简单易用。

原型

Public Function OpenADORecordset(RecordSource As String, \_

Optional LockType As ADOLockTypeEnum = adLockReadOnly, \_

Optional Connection As Variant \_

) As Object

相关引用

[OpenADORecordset 函数](http://www.accessgood.com/help/functions/OpenADORecordset.html)

示例

'打开 客户信息表 对应的只读记录集

Set rst = OpenADORecordset("SELECT \* FROM 客户信息表")

'打开 客户信息表 对应的只读记录集并将其绑定到组合框

Set Me.cboCustomer.Recordset = OpenADORecordset("SELECT \* FROM 客户信息表")

## 3. CheckRequired 函数

检查窗体上的所有设置了标记的输入控件的值是否为空，如果有为空的输入控件则显示提示消息并自动将焦点移到该控件。

**示例**

Private Sub btnSave\_Click()

If CheckRequired(Me) Then Exit Sub '如果有必填项为空，则退出过程不再继续保存操作

If CheckTextLength(Me) Then Exit Sub '如果有文本字段的长度超过最大长度，则退出过程不再继续保存操作

'其它保存数据相关的代码

End Sub

## 4. HasPermission权限功能

Private mdctPermissionList  As Collection

Private Sub Form\_Open(Cancel As Integer)

'其它代码省略

'btn添加分类、btn修改分类、btn删除分类这3个按钮均通过“商品分类树维护”这一个权限项来控制

Me.btn添加商品分类.Enabled = mdctPermissionList("商品分类树维护")

Me.btn修改商品分类.Enabled = mdctPermissionList("商品分类树维护")

Me.btn删除商品分类.Enabled = mdctPermissionList("商品分类树维护")

End Sub

Private Sub Form\_Open(Cancel As Integer)

Set mdctPermissionList = LoadPermissions("库存查询", Me)

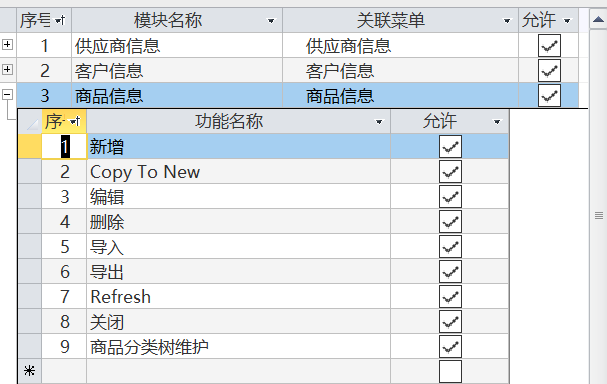
If Not mdctPermissionList("<Access Module>") Then

Exit Sub

End If

ApplyTheme Me '应用主题颜色

End Sub



**EnableButton Me.btn添加商品分类, HasPermission("商品信息", "商品分类树维护")**

**EnableButton Me.btn修改商品分类, HasPermission("商品信息", "商品分类树维护")**

**EnableButton Me.btn删除商品分类, HasPermission("商品信息", "商品分类树维护")**

使用方法：HasPermission("模块名称", "权限名称")  返回：True 或者 False   
应用：Me.btnAudit.Enabled =HasPermission("采购订单管理", "审核")

## 5.窗体加载事件给组合框赋上行来源并实现智能筛选

Private mclsSC商品ID As New SearchComboBox '声明一个组合框动态筛选类并将其实例化

Private Sub Form\_Load()

mclsSC商品ID.Init Combo:=Me.商品ID, \_

SearchField:="商品编码 & 品名规格 & 拼音码", \_

SQLSELECT:="商品ID, 商品编码, 品名规格, 单位, 最新进价 AS 单价", \_

SQLFROM:="商品信息表", \_

SQLWHERE:="已停用=0", \_

SQLORDERBY:="品名规格, 商品编码"

End Sub

Private Sub Form\_Close()

mclsSC单位名称 = Nothing ‘对象清空

End Sub

说明：

SearchField:指要搜索的字段有哪些

SQLSELECT:指SQL代码

SQLWHERE:指条件

SQLORDERBY：指排序方式

Private Sub 商品ID\_AfterUpdate()

Me.品名规格 = Me.商品ID.Column(2) '2代表第3列，下同

Me.单位 = Me.商品ID.Column(3)

Me.库存数量 = Me!商品ID.Column(4)

Me.成本价 = Me!商品ID.Column(5)

Me.售价 = Me!商品ID.Column(6)

Me.单价 = Nz(Me.售价, Me.单价)

Me.数量 = Nz(Me.数量, 1)

Me.已出库数量 = 0

Me.Dirty = False

End Sub

## 6. 商品编码发生重复时提醒并自动跳转到重复项

将字段“商品ID”设置为主键

Private Sub Form\_Error(DataErr As Integer, Response As Integer)

Const conDuplicateKey = 3022

Dim strMsg As String

Debug.Print DataErr

If DataErr = conDuplicateKey Then

Response = acDataErrContinue

strMsg = "商品编码不允许重复，请重新选择！"

Me.商品ID.SetFocus

MsgBox strMsg, vbCritical, "提示"

Me.商品ID = Null

Me.undo ‘可使用 **Undo** 方法清除对某个包含无效输入的记录的更改

End If

End Sub

**自动跳转方法：**

Private Sub 商品ID\_AfterUpdate()

On Error GoTo ErrorHandler

Dim strID As String

Me!商品编码 = Me.商品ID.Column(1)

Me.品名规格 = Me.商品ID.Column(2)

Me.单位 = Me.商品ID.Column(3)

Me.单价 = Me.商品ID.Column(4)

Me.数量 = Nz(Me!数量, 1)

Me.已入库数量 = 0

Me.Dirty = False

ExitHere:

Exit Sub

ErrorHandler:

Select Case Err

Case 3022

MsgBoxEx "同样的商品不能重复添加，请直接修改数量。", vbExclamation, "提示"

strID = Me.商品ID

Me.Undo

Me.Recordset.MoveFirst

Me.Recordset.FindFirst "商品ID=" & stride ‘& SQLText(strID)

Me.数量.SetFocus

Case Else

MsgBoxEx Err.Description, vbCritical

End Select

Resume ExitHere

End Sub

## 7.ApplyTheme Me    '应用主题颜色

## 8.SQLText 函数

返回经过SQL语法格式化处理的字符串。

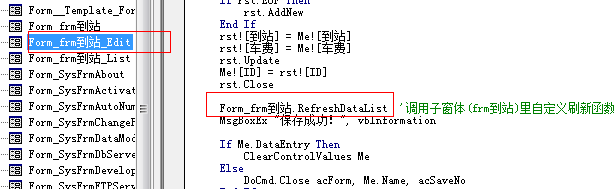
strSQL = "DELETE FROM Customers WHERE CustomerName='" & Me.txtCustomerName & "'" '常规写法，含有单引号时会出错

strSQL = "DELETE FROM Customers WHERE CustomerName=" & SQLText(Me.txtCustomerName) '使用SQLText函数的写法，不会出错

## 9. 刷新子窗体代码示例\RefreshDataList刷新窗体应用

用示例中子窗体frm到站 来举例:

第一步.在编辑窗体《保存按钮单击事件里》加上         
Form\_frm到站.RefreshDataList  '调用子窗体里自定义刷新函数 。如下图

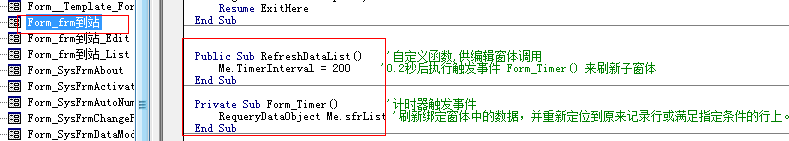


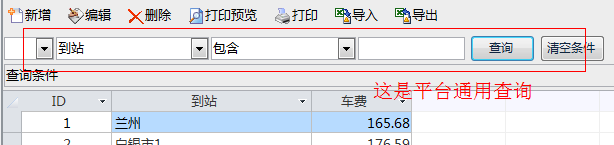
第二步.在子窗体frm到站 里加上自定义刷新函数及触发事件就OK了  
Public Sub RefreshDataList()    '自定义函数,供编辑窗体调用  
    Me.TimerInterval = 200     '0.2秒后执行触发事件 Form\_Timer() 来刷新子窗体

End Sub

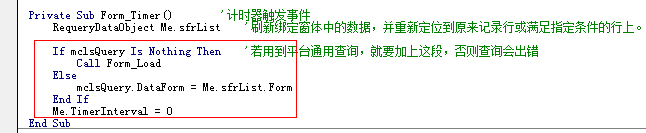
Private Sub Form\_Timer()  
     RequeryDataObject Me.sfrList  '刷新绑定窗体中的数据

    Me.TimerInterval = 0  
 End Sub  
如下图





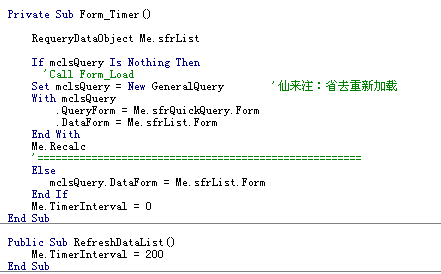
另外:假如子窗体还带上平台通用查询，那还要加上一段代码  
    If mclsQuery Is Nothing Then    '若用到平台通用查询，就要加上这段，否则查询会出错  
        Call Form\_Load  
    Else  
        mclsQuery.DataForm = Me.sfrList.Form  
    End If  
 如下图:



**总结:**

此文自定义RefreshDataList函数作用是调用 RequeryDataObject刷新函数

RequeryDataObject才是快速平台刷新窗体真正函数，它定义在代码库里。  
 意思是刷新绑定窗体中的数据，并重新定位到原来记录行或满足指定条件的行上。  
相关示例  
Sub Test()  
    **RequeryDataObject Me                                 '刷新当前窗体中的数据  
    RequeryDataObject Me.sfrOrderDetail                  '刷新sfrOrderDetail子窗体中的数据  
    RequeryDataObject Me.sfrOrderDetail, "OrderID = 144" '刷新sfrOrderDetail子窗体中的数据，并重新定位到OrderID等于144的记录行**End Sub



Public gsfrList As SubForm

RequeryDataObject gsfrList

## 10.ExportToExcel 函数

将窗体中的绑定数据源中的数据导出至一个Excel文件。

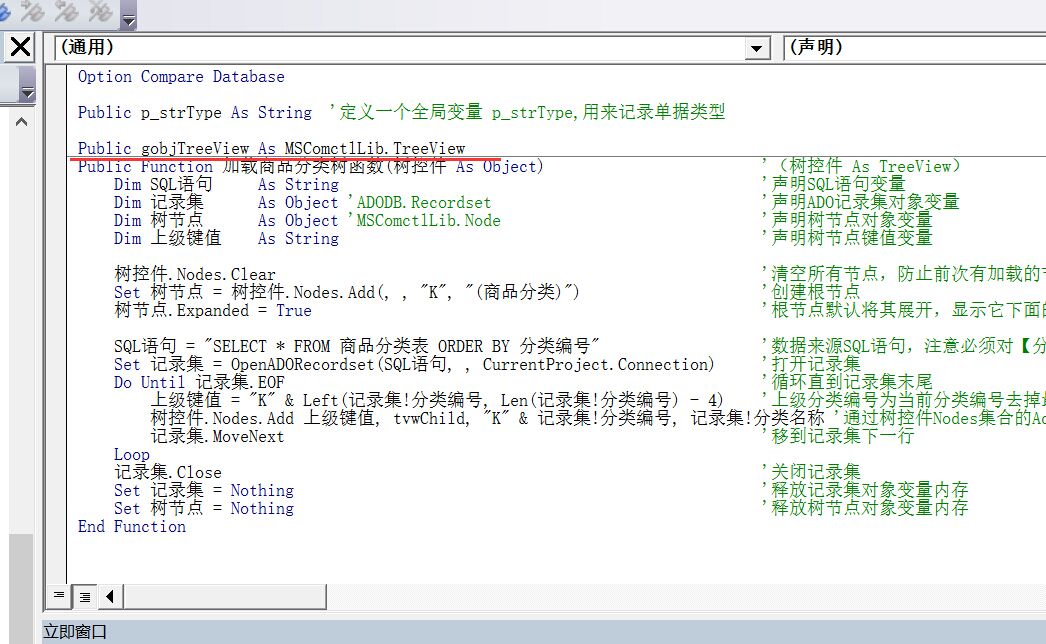
此函数实际上就是将打开Excel，复制窗体数据，粘贴到Excel，调整格式，然后保存这一系列的手工操作进行了自动化实现封装。使用复制、粘贴的方式比使用CopyFromRecordset等方式兼容性更好，且可以将Access中的格式一同复制过来。数据复制完成后会进行调整行高列宽、取消网格线显示、添加框线、冻结行首（StartRange所在的行）等格式化处理。

Private Sub btnExport\_Click()

Me.明细子窗体.SetFocus '必须先让绑定数据的窗体或子窗体获得焦点

ExportToExcel Me.明细子窗体.Form.Caption '取子窗体源对象对应的窗体的标题（假设设置好了标题）

End Sub



模块1 里这句是：Public gobjTreeView As object

后期绑定是：Public gobjTreeView As MSComctlLib.TreeView

隐式声明：其实就是不声明，或者是声明的很笼统，比如声明为Objec,Variant。

显示声明：声明类型很精准

## 11. 行号能自动显示序号GetLineNumber([Form])

“行号”文本框的控件来源中输入：=GetLineNumber([Form])

行号第一行显示错误时，将“Tab键索引值”设为除0以外的其它数字。

## 12. 删除数据

Private Sub btnDelete\_Click()

On Error Resume Next

If Not Me.NewRecord Then

DoCmd.SetWarnings False '屏弊系统的警告

RunCommand acCmdDeleteRecord '删除当前记录

DoCmd.SetWarnings True '取消屏弊系统的警告

End If

End Sub

Private Sub btnDeleteAll\_Click()

On Error Resume Next

If MsgBoxEx("确定要清空当前列表中的所有商品吗？", vbQuestion + vbOKCancel) = vbOK Then

CurrentDb.Execute "DELETE FROM TMP\_采购订单明细表" '删除该表全部数据

Me.Requery '刷新数据

End If

End Sub

## 13. 添加新商品"命令按钮

Private Sub btnAddProduct\_Click()

DoCmd.OpenForm "frm商品信息\_Edit", , , , acFormAdd, acDialog

Me!商品ID.Requery

End Sub

## 14. 经办人和制单人改为组合框后，[赋上默认值](http://www.accessoft.com/rdp/tutorial/app_full/jxc/300606.asp#DesignStep2)

Dim rstUser As Object

Set rstUser = OpenADORecordset("SELECT Sys\_Users.Nickname FROM Sys\_Users", adLockReadOnly)

Set Me.经办人.Recordset = rstUser

Set Me.制单人.Recordset = rstUser

……

If IsLoaded(**acForm,** "SysFrmMain") = True Then

Me.制单人 = Forms!SysFrmMain.Nickname '该昵称在SysFrmMain窗体,所以必须让SysFrmMain窗体处于打开状态

End If

Me.采购日期 = Date '也可以直接在控件属性中设置

Me.制单时间 = Date '也可以直接在控件属性中设置

Me.供应商ID.SetFocus '这样一打开,就直接定位到选择供应商

strUsername = GetParameter("Current User Nickname", dbText, "")

## 15. GetAutoNumber 函数

根据“[自定义自动编号管理](http://www.accessgood.com/help/AutonumberManage.html)”界面中定义的编号规则，生成一个自定义的自动编号

'根据管理界面的配置参数直接生成销售单号

Me.txtOrderNo = GetAutoNumber("销售单号") '假设生成的编号是 XS20130101-001

'例如需要将部门编号加到前缀中，调用时给出编号前缀参数，假设这里的部门编号文本框 txtDepartID 的值是003

Me.txtOrderNo = GetAutoNumber("销售单号", Prefixal:="XS" & Me.txtDepartID & "-") '生成的编号是 XS003-20130101-001

## 16.根据单据类型显示相关的订单号 (级联筛选)

Private Sub 单据类型\_AfterUpdate()

Select Case Me!单据类型

Case "采购入库单"

Me!相关订单号.RowSource = " SELECT 采购订单号,采购日期 & ' ' & 供应商名称 & '（经办人：' & 经办人 & '）' As 摘要" \_

& " FROM 采购订单查询 WHERE 状态='已审核'"

Me!相关订单号.Visible = True

Case "销售出库单"

Me!相关订单号.RowSource = " SELECT 销售订单号,销售日期 & ' ' & 客户名称 & '（经办人：' & 经办人 & '）' As 摘要" \_

& " FROM 销售订单查询 WHERE 状态='已审核'"

Me!相关订单号.Visible = True

Case Else

Me!相关订单号.RowSource = ""

Me!相关订单号.Visible = False

End Select

Me.Caption = Me!单据类型 & "明细"

If Me!相关订单号.ListIndex < 0 Then

Me!相关订单号 = Null

End If

End Sub

## 17.日期选择

Private Sub cbo区间\_AfterUpdate()

If IsNull(Me.cbo区间) Then

Me.cbo区间 = "自定义"

End If

Select Case Me.cbo区间

Case "今天"

Me.txt开始日期 = Date

Me.txt截止日期 = Date

Case "本周"

Me.txt开始日期 = FirstDayOfWeek()

Me.txt截止日期 = Date

Case "本月"

Me.txt开始日期 = FirstDayOfMonth()

Me.txt截止日期 = Date

Case "本年"

Me.txt开始日期 = DateSerial(Year(Date), 1, 1)

Me.txt截止日期 = Date

End Select

Me.btn查询.Visible = (Me.cbo区间 = "自定义")

Me.txt开始日期.Enabled = Me.btn查询.Visible

Me.txt截止日期.Enabled = Me.btn查询.Visible

If Not Me.btn查询.Visible Then

Call btn查询\_Click

End If

End Sub

## 18.根据条件改变子窗体数据源

参数查询类似于存储过程。

Private Sub btn查询\_Click()

On Error GoTo ErrorHandler

Dim strSQL As String

Dim strWhere As String

If Me.sfrList.SourceObject <> Me.Name & "\_" & Me.cbo统计方式 Then

Me.sfrList.SourceObject = Me.Name & "\_" & Me.cbo统计方式

End If

If Me.txt开始日期 <> "" Then strWhere = strWhere & " AND 采购统计查询.日期>=" & SQLDate(Me.txt开始日期)

If Me.txt截止日期 <> "" Then strWhere = strWhere & " AND 采购统计查询.日期<=" & SQLDate(Me.txt截止日期)

If strWhere <> "" Then strWhere = " WHERE " & Mid(strWhere, 6)

Select Case Me.cbo统计方式

Case "按商品统计" '下面的SQL代码其实是根据 “采购统计查询\_按商品统计”查询适当修改而成

strSQL = " SELECT 商品信息表.分类编号, 商品信息表.商品编码" \_

& ",商品信息表.品名规格, 商品信息表.单位" \_

& ",Sum(采购统计查询.数量) AS 采购数量, Sum(采购统计查询.金额) AS 采购金额" \_

& " FROM 采购统计查询" \_

& " INNER JOIN 商品信息表 ON 采购统计查询.商品ID = 商品信息表.商品ID" \_

& strWhere \_

& " GROUP BY 商品信息表.分类编号, 商品信息表.商品编码" \_

& ",商品信息表.品名规格, 商品信息表.单位" \_

& " ORDER BY Sum(采购统计查询.金额) DESC;"

Case "按商品分类统计" '下面的SQL代码其实是根据 “采购统计查询\_按商品分类统计”查询适当修改而成

strSQL = " SELECT 商品分类表.分类名称" \_

& ",Sum(采购统计查询.数量) AS 采购数量, Sum(采购统计查询.金额) AS 采购金额" \_

& " FROM (采购统计查询" \_

& " INNER JOIN 商品信息表 ON 采购统计查询.商品ID = 商品信息表.商品ID)" \_

& " INNER JOIN 商品分类表 ON 商品分类表.分类编号 = 商品信息表.分类编号" \_

& strWhere \_

& " GROUP BY 商品分类表.分类名称" \_

& " ORDER BY Sum(采购统计查询.金额) DESC;"

Case "按供应商统计" '下面的SQL代码其实是根据 “采购统计查询\_按供应商统计”查询适当修改而成

strSQL = " SELECT 供应商信息表.供应商名称" \_

& ", Sum(采购统计查询.数量) AS 采购数量, Sum(采购统计查询.金额) AS 采购金额" \_

& " FROM (采购统计查询" \_

& " INNER JOIN 采购订单表 ON 采购统计查询.相关订单号 = 采购订单表.采购订单号)" \_

& " INNER JOIN 供应商信息表 ON 供应商信息表.供应商ID = 采购订单表.供应商ID" \_

& strWhere \_

& " GROUP BY 供应商信息表.供应商名称" \_

& " ORDER BY Sum(采购统计查询.金额) DESC;"

End Select

CurrentDb.QueryDefs("采购统计查询\_" & Me.cbo统计方式).SQL = strSQL '动态修改查询的SQL代码

Me.sfrList.Form.RecordSource = Me.sfrList.Form.RecordSource

ExitHere:

Exit Sub

ErrorHandler:

MsgBoxEx Err.Description, vbCritical

Resume ExitHere

End Sub

## 19. DatePickerFor 函数

在指定控件（一般为文本框）下方显示日历，并返回在日历中点击选择的日期作为控件值。“通用日期选择器”组件的接口函数。

推荐通过将文本框的事件属性（如单击或双击）设置为表达式“=DatePickerFor([txtOrderDate])”的方式来进行调用，这样无需写代码。

**提示** 主要用于Access 2003，Access 2007及之后的版本，已有内置的相同功能。

## 20.树控件

Public gobjTreeView As MSComctlLib.TreeView

Public Function 加载商品分类树函数(树控件 As Object) '（树控件 As TreeView）

Dim SQL语句 As String '声明SQL语句变量

Dim 记录集 As Object 'ADODB.Recordset '声明ADO记录集对象变量

Dim 树节点 As Object 'MSComctlLib.Node '声明树节点对象变量

Dim 上级键值 As String '声明树节点键值变量

树控件.Nodes.Clear '清空所有节点，防止前次有加载的节点未被释放

Set 树节点 = 树控件.Nodes.Add(, , "K", "(商品分类)") '创建根节点

树节点.Expanded = True '根节点默认将其展开，显示它下面的子节点

SQL语句 = "SELECT \* FROM 商品分类表 ORDER BY 分类编号" '数据来源SQL语句，注意必须对【分类编号】字段排升序

Set 记录集 = OpenADORecordset(SQL语句, , CurrentProject.Connection) '打开记录集

Do Until 记录集.EOF '循环直到记录集末尾

上级键值 = "K" & Left(记录集!分类编号, Len(记录集!分类编号) - 4) '上级分类编号为当前分类编号去掉最后4位

树控件.Nodes.Add 上级键值, tvwChild, "K" & 记录集!分类编号, 记录集!分类名称 '通过树控件Nodes集合的Add方法将每个商品分类以节点的形式添加

记录集.MoveNext '移到记录集下一行

Loop

记录集.Close '关闭记录集

Set 记录集 = Nothing '释放记录集对象变量内存

Set 树节点 = Nothing '释放树节点对象变量内存

End Function

Private Sub Form\_Load()

'其它代码此处省略

Me.Recalc

加载商品分类树函数 Me.ocx分类树.Object '调用自定义函数“加载商品分类树函数”将数据加载到树控件

Me.ocx分类树.SetFocus   '本行代码是加入的，让 ocx分类树 控件获得焦点

ExitHere:

Exit Sub

'其它代码此处省略

End Sub

Public Function RunQuery(WhereCondition As String)

Dim strWhere As String

Dim objNode As Object 'MSComctlLib.Node

Set objNode = Me.ocx分类树.SelectedItem

'自定义的多条件查询中，每个条件生成表达式的时候，可以把If Then写到一行，这样比较紧凑

'条件再多也基本上一屏之内可以显示全，方便阅读

If Not objNode Is Nothing Then

If objNode.Key <> "K" Then strWhere = " AND 分类编号 Like '" & Mid(objNode.Key, 2) & "\*'"

End If

If Len(WhereCondition) > 0 Then strWhere = strWhere & " AND " & WhereCondition

'前面根据每个条件生成的表达式前面都有 “ AND ”，因此到最后最前面的就是多余的，需要去掉

'因此这里用 Mid 函数从左起第6个字符开始取（AND加上前后的空格共5个字符），前面的就被去掉了

strWhere = Mid(strWhere, 6)

'如果条件不为空，则在前面加上 WHERE 关键字，用于生成 SQL 语句的 WHERE 子句

If Len(strWhere) > 0 Then strWhere = " WHERE " & strWhere

'通过重置数据列表子窗体“记录源”属性的方式来实现查询

Me.sfrList.Form.RecordSource = "SELECT \* FROM 商品信息表" & strWhere

End Function

Private Sub ocx分类树\_NodeClick(ByVal Node As Object) '树控件“节点单击”事件过程

Me.lbl当前分类.Caption = "当前分类：" & Node.FullPath '通过“lbl当前分类”标签，来显示当前点击的商品分类节点的路径

mclsQuery.ClearConditions '调用通用查询类模块的 ClearConditions 方法清空通用查询模块界面中的条件

RunQuery "" '点击节点后自动查询，查询条件就只有当前点击选中的商品分类

'之前在界面中的其它条件自动清空

End Sub

Private Sub ocx分类树\_GotFocus() '树控件“获得焦点”事件过程

'树控件获得焦点时，要取消强制节点高亮，否则高节节点会和焦点节点分离

Set Me.ocx分类树.DropHighlight = Nothing

End Sub

Private Sub ocx分类树\_LostFocus() '树控件“失去焦点”事件过程

'树控件失去焦点时让当前选中的节点仍然保持高亮显示，以便于知道当前选中的是哪个节点

Set Me.ocx分类树.DropHighlight = Me.ocx分类树.SelectedItem

End Sub

## 21. ClearControlValues 函数

清空所有输入控件的值，以初始化输入界面便于重新录入数据。

Sub Test()

ClearControlValues Me '清空当前窗体中所有控件的值

End Sub

## 22. 禁止修改“最近采购日期”的相关设置

|  |  |  |
| --- | --- | --- |
| **属性** | **属性值** | **说明** |
| **锁定** | 是 |  |
| **背景色** | 15592941 | 和文本框所在的“主体”节背景色一致。 |

短文本和日期居中对齐、长文本左对齐、数字右对齐

## 23. Dcount

返回数据库或数据清单的列中满足指定条件并且包含数字的单元格个数。

23.1统计表中非空值的记录数方法如下:  
=DCount("\*", "表名称", "[字段名] Is Not Null")

|  |  |
| --- | --- |
| 23.2  If Me![已停用] = True Then | |
|  | If DCount("\*", "商品信息表", "商品ID=" & Me!商品ID & " AND 库存数量<>0") > 0 Then |
|  | MsgBox "不允许停用，目前该商品尚有库存。", vbExclamation, "提示" |
|  | Exit Sub |
|  | End If  23.3 |

If DCount("\*", "表名", "字段名 Is Null or 字段名 = 0") > 0 Then

              MsgBox "提示内容", vbInformation, "提示"  
        Exit Sub  
    End If

23.4在子窗体为数据表时，为了统计某条记录在另一个表中出现的记录，使用了DCount函数。开始使用的是下面的表达式：

=DCount("[ID]","TestContent","[JOB\_NO] =" & me.Job\_No)

运行下来，出错，最后花了近一个小时才找到解决方法.

解决方法一：

表达式应写为：

=DCount("[ID]","TestContent","[JOB\_NO] =" & Forms!主窗体名称!子窗体控件名称!Job\_No)

注意：子窗体控件名称不是子窗体的在窗体设计视图显示的名称，而是在主窗体中子窗体控件的名称。

解决方法二:

=DCount("[ID]","TestContent","[JOB\_NO] =" & Job\_No

23.5统计为空的记录数**=DCount("\*","表名称","IsNull([字段名])")**

**23.6**查询商品编码出现序次

在查询视图中的字段

商品编码出现序次: DCount("商品编码","表1","商品编码='" & [商品编码] & "' and 编号 < " & [编号] & "")+1

## 24.子窗体数据源改变

Me.sfrList.Form.RecordSource = "SELECT \* FROM 商品信息表" & strWhere

RecordSource  是指窗体记录源

SourceObject   是指子窗体源对象，用于链接的[未绑定对象框](javascript:AppendPopup(this,'defUnboundObjectFrame_1_1'))

Forms!usysfrmMain!frmChild.SourceObject = "frmyg\_child"

 ControlSource  是指控件来源

## 25.窗体打开模式、 OpenArgs传递

  '以新增模式打开分类编辑窗体

DoCmd.OpenForm Me.Name & "\_分类编辑", , , , **acFormAdd**, acDialog

'以修改模式打开分类编辑窗体，acDialog参数必须要有

DoCmd.OpenForm Me.Name & "\_分类编辑", , , , **acFormEdit**, acDialog

|  |  |
| --- | --- |
| 1. **acFormReadOnly** 即只读模式，允许编辑：否，数据录入：否   '将当前选中的分类编号保存到全局变量，传递给编辑窗体作为默认值 | |
|  | DoCmd.OpenForm FormName:=Me.Name & "\_Edit", \_ |
|  | DataMode:=acFormAdd, \_ |
|  | OpenArgs:=Mid(Me.ocx分类树.SelectedItem.Key, 2) ‘窗体间参数传递  C:\Users\fxh\Desktop\3112831767460276925.jpg |

## 26. vba代码太长，换行

**对于有sql语句，必须在句末+双引号+空格+下划线，下一句前面+&+空格+双引号**

**对于非sql，只需要+空格+\_**

## 27.数据刷新

1.Requery  重新查询 语法为：对象.Requery

Me.cbo上级分类编号.Requery

\* 重新运行基于指定窗体或控件的查询。  
    \* 显示任何新的或已更改的记录，从窗体中移除已删除的记录。  
    \* 根据对窗体的Filter属性所作的任何更改，来更新显示记录。

有些刷新子窗体数据是通过设置子窗体控件的“源对象”属性，来重新加载子窗体实现刷新，这个说到底其实也是用的Requery方法，因为窗体加载时会自动调用Requery方法。Requery是最稳定有效的数据刷新方法。

2. Refresh  刷新 窗体.Requery  
    3. Recalc   更新计算控件 窗体.Recalc

主要是用于刷新计算控件（即控件来源是以等号（=）开头的控件）

## 28. 快速书写SQL语句

偷梁换柱，用 Replace 函数替换的方式，代替引号、连接号的作用

原语句：

strSQL = "Insert INTO 凭证记录表 (状态,日期,制单人,制单时间,关联单号)" \_

& "VALUE ('未审核', " & Format(Me.日期,"\#yyyy-mm-dd\#") & ",'" \_

& Forms!SysFrmMain!Nickname & "',Now(),'" & Me.单号 & "')"

更改后：

strSQL = " Insert INTO 凭证记录表 (状态,日期,制单人,制单时间,关联单号)" \_

& " VALUE ('未审核', #日期#,'@制单人',Now(),'@关联单号')"

strSQL = Repalce(strSQL, "@日期", Format(Me.日期, "yyyy-mm-dd")

strSQL = Repalce(strSQL, "@制单人", Forms!SysFrmMain!Nickname)

strSQL = Repalce(strSQL, "@关联单号", Me.单号)

|  |  |
| --- | --- |
| strSQL = "UPDATE 商品分类表 SET 分类名称 = '@新分类名称' WHERE 分类编号 = '@分类编号'" | |
|  | strSQL = Replace(strSQL, "@新分类名称", Me.txt分类名称) |
|  | strSQL = Replace(strSQL, "@分类编号", Me.txt分类编号) |
|  | DAORunSQL strSQL |

## 29. CurrentDb.Execute执行sql语句

CurrentDb.Execute "DELETE FROM TMP\_采购订单明细表" '删除该表全部数据，无系统提示

Docmd.runsql 有系统提示

DoCmd.SetWarnings False

DoCmd.RunSQL "UPDATE 供应商信息表 SET 最近采购日期 = #" & Me.采购日期 & "# WHERE 供应商ID =" & Me.供应商ID

'审核时更新商品信息表中的最新进价

DoCmd.RunSQL "UPDATE 商品信息表 INNER JOIN TMP\_采购订单明细表 ON (商品信息表.单位=TMP\_采购订单明细表.单位) " \_

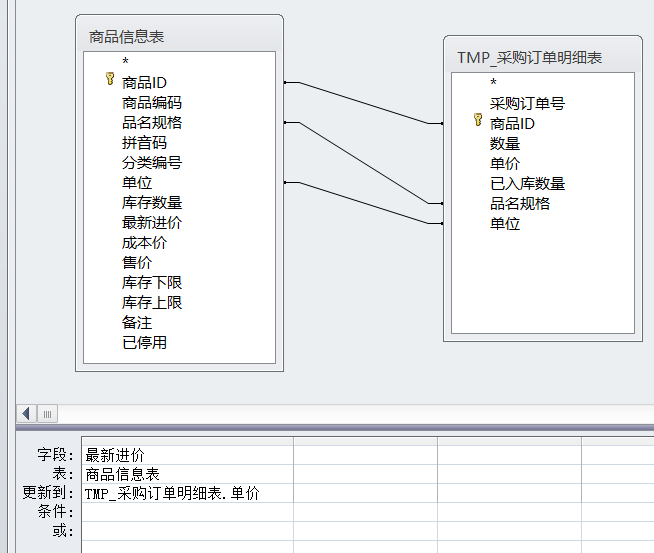
& "AND (商品信息表.品名规格=TMP\_采购订单明细表.品名规格) AND (商品信息表.商品ID=TMP\_采购订单明细表.商品ID) " \_

& "SET 商品信息表.最新进价 = TMP\_采购订单明细表.单价;"

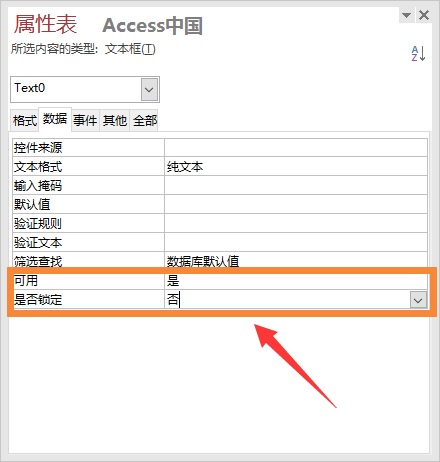
DoCmd.SetWarnings True

'注意，这时还要对窗体 frm供应商信息\_Edit 的加载事件[Form\_Load()]中的 Me![最近采购日期] = rst![最近采购日期] 前面的注释去掉，

'不然会发生打开供应商详细信息时，表中有数据，而窗体上没显示。



## 30.Enabled属性与Locked属性的区别



**1. Enabled 是禁用，变灰，光标也不能移上去**

**2. Locked是锁定，不能修改，但光标可移上去，也不变灰**

**3.另禁用或锁定，2个设置是不同的，一个设置为False(Enabled), 一个设置为True(Locked)**

所以如何希望控件变灰，不想让用户点进去，就使用.Enabled=False

如果只是控件用户不能修改，，但光标可定位进去，则应该使用.Locked=True

## 31. SelectAllListBoxItems 函数

对一个允许多选的列表框，进行全选/全不选操作。

Private Sub btn全选\_Click()

DoCmd.Hourglass True

SelectAllListBoxItems Me.lst商品ID, True

DoCmd.Hourglass False

End Sub

Private Sub btn全不选\_Click()

DoCmd.Hourglass True

SelectAllListBoxItems Me.lst商品ID, False

DoCmd.Hourglass False

End Sub

## 32.LTrim，RTrim，及 Trim 函数示例

本示例使用 **LTrim** 及 **RTrim** 函数将某字符串的开头及结尾的空格全部去除。事实上只使用 **Trim** 函数也可以做到将两头空格全部去除。

Dim MyString, TrimString

MyString = " <-Trim-> " ' 设置字符串初值。

TrimString = LTrim(MyString) ' TrimString = "<-Trim-> "。

TrimString = RTrim(MyString) ' TrimString = " <-Trim->"。

TrimString = LTrim(RTrim(MyString)) ' TrimString = "<-Trim->"。

' 只使用 Trim 函数也同样将两头空格去除。

TrimString = Trim(MyString) ' TrimString = "<-Trim->"。

## 33. 在窗体上显示当前记录和总记录数

在窗体上，我们需要看到有多少条记录和当前记录是第几条，这样有利于我们总体把握记录的变化，更利于维护数据。  看似很难的问题，其实我们可以用CurrentRecord和RecordsetClone两个属性，零代码可以实现显示多少条记录和当前记录。

    CurrentRecord 属性：该属性值对应于窗体左下角的记录编号框中所显示的值。在 Microsoft Access 中，该属性为 Long Integer 型，其值代表窗体上显示的当前记录的记录编号。  
     RecordsetClone 属性：RecordsetClone 属性设置是由窗体的 RecordSource 属性指定的基础查询或基础表的一个副本。例如，如果窗体基于一个查询，那么对 RecordsetClone 属性的引用与使用相同查询来复制 Recordset 对象是等效的。可以使用 RecordCount 属性来计算 Recordset 对象中的记录数

**只需要在窗体上放两个文本框，分别命名为：txtRecNo和txtRecCount，对应于当前记录号和记录数。**

**将txtRecNo的控件来源属性设为：= CurrentRecord**

**将txtRecCount的控件来源属性设为：= RecordsetClone.RecordCount**

## 34. For Each in ... next 详解用法

把当前窗体内所有的控件都逐个引用一次（子窗体控件内的控件不算在内）。这种用“For Each [对象或属性] In [对象集合或属性集合]”的遍历方法在ACCESS编程中属于比较常见的用法。

Function IsLoaded(strFrmName As String) As Boolean

    '  确定一个窗体是否已被装载。

        Const conFormDesign = 0

    Dim objForm As Form

        IsLoaded = False

        For Each objForm In Forms

        If objForm.FormName = strFrmName Then

            If objForm.CurrentView <> conFormDesign Then  '窗体不是设计模式时

                IsLoaded = True

                Exit Function  ' 一旦找到了此窗体，就退出本函数。

            End If

        End If

    Next

 End Function

For Each ctl In Me.Controls

            '根据ctl的控件类型来选择

    Select Case ctl.ControlType

      Case acTextBox '是文本框，要清空(注意，子窗体下面还有两个锁定的文本框不能赋值)

                If ctl.Locked = False Then ctl.Value = Null

            Case acComboBox '是组合框，也要清空

                ctl.Value = Null

            '其它类型的控件不处理

        End Select

Next

Dim ctl as Access.control

        For Each ctl In Me.Controls

            If ctl.ControlType = acTextBox Then

                ctl.Value = Null

            End If

        Next

## 35. DAO使用书签进行搜索并同步窗体的方法

一般来说，我们在窗体上搜索 可以通过设置

    me.filter="条件"  
    me.filteron=true     来实现

另外一种方法

    With Me.RecordsetClone  
    .FindFirst "[字段]=" & Me!搜索的值  
    If Not .NoMatch Then  
       If Me.Dirty Then  
          Me.Dirty = False  
       End If  
       Me.Bookmark = .Bookmark  
    End If  
   End With

通过 recordset 变量,则用下面方法:  
  Dim rs As DAO.Recordset  
  Set rs = Me.RecordsetClone  
  rs.FindFirst "[字段]=" & Me!搜索的值  
  If Not rs.NoMatch Then  
     If Me.Dirty Then  
        Me.Dirty = False  
     End If  
     Me.Bookmark = rs.Bookmark  
  End If  
  Set rs = Nothing

如果是设置父窗体  
     With Me.Parent.RecordsetClone '取当前窗体的记录集克隆   
          '查找需要的数据  
          .FindFirst "[字段]=" & Me!搜索的值  
          '如果找到了,就将当前记录集的书签同步给当前窗体  
          If Not .NoMatch Then  
             If Me.Parent.Dirty Then  
                Me.Parent.Dirty = False  
             End If  
             Me.Parent.Bookmark = .Bookmark  
          End If  
        End With

## 36. SelectAllListBoxItems 函数

对一个允许多选的列表框，进行全选/全不选操作。

DoCmd.Hourglass True

SelectAllListBoxItems Me.lst商品ID, True

DoCmd.Hourglass False

## 37. Dirty 属性

Dirty 属性来确定当前记录自上次保存之后是否修改过。可读/写 Boolean 类型。



Private Sub 商品ID\_AfterUpdate()

Me.品名规格 = Me.商品ID.Column(2) '2代表第3列，下同

Me.单位 = Me.商品ID.Column(3)

Me.库存数量 = Me!商品ID.Column(4)

Me.成本价 = Me!商品ID.Column(5)

Me.售价 = Me!商品ID.Column(6)

Me.单价 = Nz(Me.售价, Me.单价)

Me.数量 = Nz(Me.数量, 1)

Me.已出库数量 = 0

'Me.Dirty = False

End Sub

**me.dirty=true ，已修改未保存----是  ，不保存。**

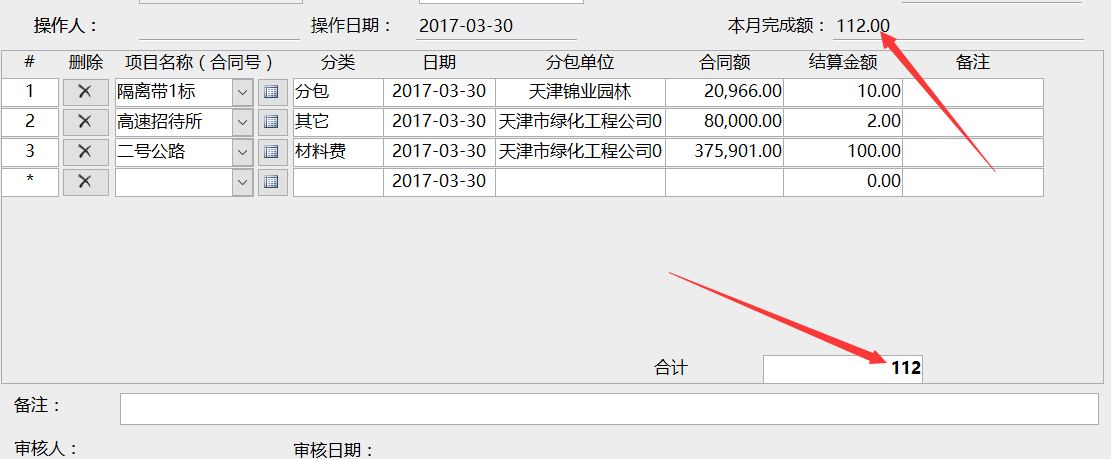
**me.dirty=false , 已修改未保存----否，说明让他保存。**

**Me.Dirty = False 的作用相当于 RunCommand acCmdSaveRecord**

**但  RunCommand acCmdSaveRecord 有焦点要求，如果没有焦点，则保存会失败出错。**

**而 Me.Dirty = False 没有焦点要求。**

## 38.主子窗体计算



**子窗体内**

Sub 本月金额合计()

Dim rst As Object

Dim curSum As Currency

If Me.Dirty Then Me.Dirty = False '如果当前处理未保存状态，先进行保存

Set rst = Me.RecordsetClone '打开订单明细窗体的记录集副本，这样不对当前记录的显示等造成影响

Debug.Print rst.RecordCount

If rst.RecordCount <> 0 Then

rst.MoveFirst '如果订单明细记录不为空，先移到第一条记录

Do Until rst.EOF '循环所有记录至到最后一行记录

curSum = curSum + rst!结算金额 '对每一行记录中的金额进行累加

rst.MoveNext

Loop

End If

Me.Parent!本月完成额 = curSum '将父窗体（即订单窗体）中的“订单金额”的值设为最后累加的值

Set rst = Nothing

End Sub

然后在“更新后”、“确认删除后”，事件中引用函数

Private Sub Form\_Load()

'必须要选中此选项，否则删除确认事件不会发生

If Not GetOption("Confirm Record Changes") Then

SetOption "Confirm Record Changes", True

DoCmd.SetWarnings False

End If

End Sub

## 39. VBA文件对话框（Application.FileDialog属性）的常用属性与方法的解释

用法如下

Application.FileDialog(fileDialogType)

fileDialogType      MsoFileDialogType 类型，必需。文件对话框的类型。

|  |
| --- |
| MsoFileDialogType 可为以下 MsoFileDialogType 常量之一。 |
| msoFileDialogFilePicker  允许用户选择文件。 |
| msoFileDialogFolderPicker  允许用户选择一个文件夹。 |
| msoFileDialogOpen  允许用户打开文件。 |
| msoFileDialogSaveAs  允许用户保存一个文件。 |

分别举例如下：

1、msoFileDialogFilePicker

1）选择单个文件

代码:

Sub SelectFile()

    '选择单一文件

    With Application.FileDialog(msoFileDialogFilePicker)

        .AllowMultiSelect = False

        '单选择

        .Filters.Clear

        '清除文件过滤器

        .Filters.Add "Excel Files", "\*.xls;\*.xlw"

        .Filters.Add "All Files", "\*.\*"

        '设置两个文件过滤器

        If .Show = -1 Then

            'FileDialog 对象的 Show 方法显示对话框，并且返回 -1（如果您按 OK）和 0（如果您按 Cancel）。

            MsgBox "您选择的文件是：" & .SelectedItems(1), vbOKOnly + vbInformation, "提示"

        End If

    End With

End Sub

2）选择多个文件

代码:

Sub SelectFile()

    '选择多个文件

    Dim l As Long

    With Application.FileDialog(msoFileDialogFilePicker)

        .AllowMultiSelect = True

        '单选择

        .Filters.Clear

        '清除文件过滤器

        .Filters.Add "Excel Files", "\*.xls;\*.xlw"

        .Filters.Add "All Files", "\*.\*"

        '设置两个文件过滤器

        .Show

        'FileDialog 对象的 Show 方法显示对话框，并且返回 -1（如果您按 OK）和 0（如果您按 Cancel）。

        For l = 1 To .SelectedItems.Count

            MsgBox "您选择的文件是：" & .SelectedItems(l), vbOKOnly + vbInformation, "提示"

        Next

    End With

End Sub

2、msoFileDialogFolderPicker

代码:

Sub SelectFolder()

    '选择单一文件

    With Application.FileDialog(msoFileDialogFolderPicker)

        If .Show = -1 Then

        'FileDialog 对象的 Show 方法显示对话框，并且返回 -1（如果您按 OK）和 0（如果您按 Cancel）。

            MsgBox "您选择的文件夹是：" & .SelectedItems(1), vbOKOnly + vbInformation, "提示"

        End If

    End With

End Sub

文件夹仅能选择一个

3、msoFileDialogOpen

4、msoFileDialogSaveAs

使用方法与前两种相同

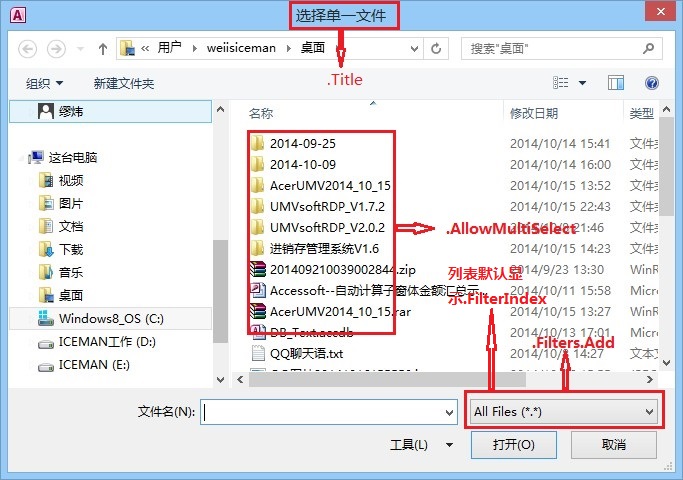
只是在.show

可以用.Execute方法来实际打开或者保存文件。

FileDialog常用属性

|  |  |
| --- | --- |
| Title | 标题 |
| Filter | 设置过滤字符 |
| FilterIndex | 指定列表框中的默认的选项 |
| AllowMultiSelect | 是否多选 |

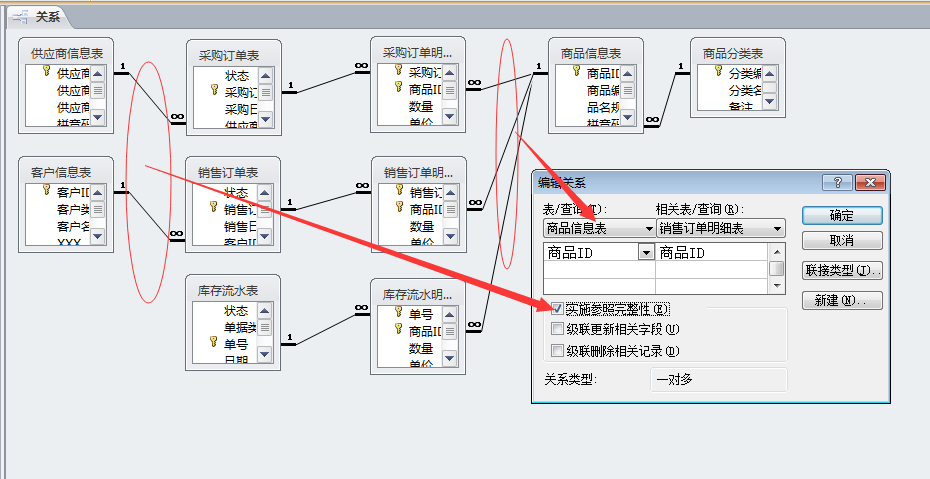
附上图片

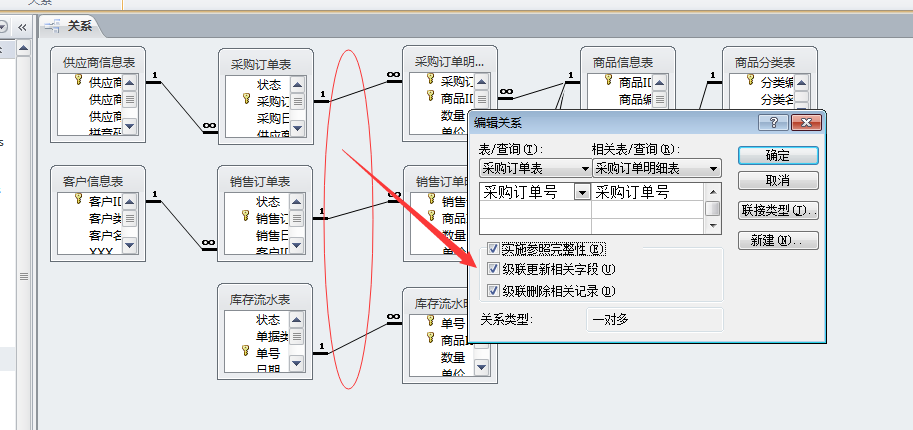


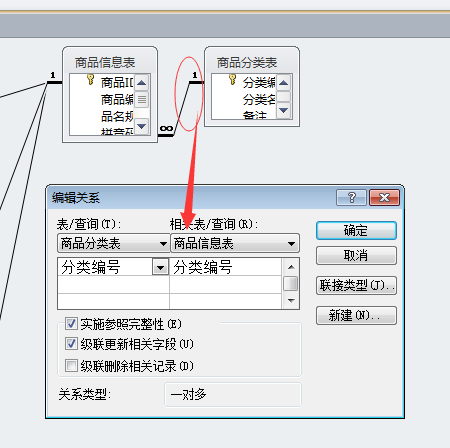
## 40. LoadRecord 新平台

LoadRecord "TMP\_采购订单明细表", "SELECT \* FROM [采购订单查询\_明细] WHERE [采购订单号]=" & SQLText(Me![采购订单号]), CurrentProject.Connection

## 41.表关系







**“实施参照完整性”**，可以对两个表中的数据进行强制约束，阻止无效数据的保存。可以生成表关系图，通过关系图直观明了的了解各表之间的关系，便于数据库维护。

**“级联更新”**，当主表主键字段的数据被修改，可以自动更新从表的外键字段数据，而无需额外编程。

**“级联删除”**，当删除主表中的数据时，可以自动删除从表中的相关数据，而无需额外编程。

是否建立严格的表关系，具体要视你开发的系统的实际情况而定：严格的关系代表着严格的控制，这样数据的规范性和准确性有足够的保障，但同时也意味着会牺牲灵活性。如果你的数据并不是通过具有严格验证的接口（如数据录入窗体）存入数据库的，而是希望通过导入的方式，先导入再加以处理之类。那么严格的表关系就很可能会造成使用上的不便，比如导入时不停向用户弹出一些警告、错误消息框，这样就会因为使用上的不友好，而使得软件的实用价值打了折扣。所以，这里需要**根据实际情况进行权衡**。

## 42.选择并添加商品

**1.模块1添加：Public gfrmTemp As Form**

**2.使用此按钮的窗体**

btnAdd添加按钮 单价事件

Private Sub btnAdd\_Click()

Set gfrmTemp = Me

DoCmd.OpenForm "frm商品选择"

End Sub

Public Function AddProduct(SelectionList As ListBox)

Dim rst As DAO.Recordset

Dim varItem As Variant

For Each varItem In SelectionList.ItemsSelected

Set rst = Me.RecordsetClone

rst.FindFirst "商品ID=" & SelectionList.Column(0, varItem)

If rst.NoMatch Then

Me.Recordset.AddNew

Me!商品ID = SelectionList.Column(0, varItem)

Call 商品ID\_AfterUpdate

Else

Me.Recordset.Bookmark = rst.Bookmark

Me!数量 = Me!数量 + 1

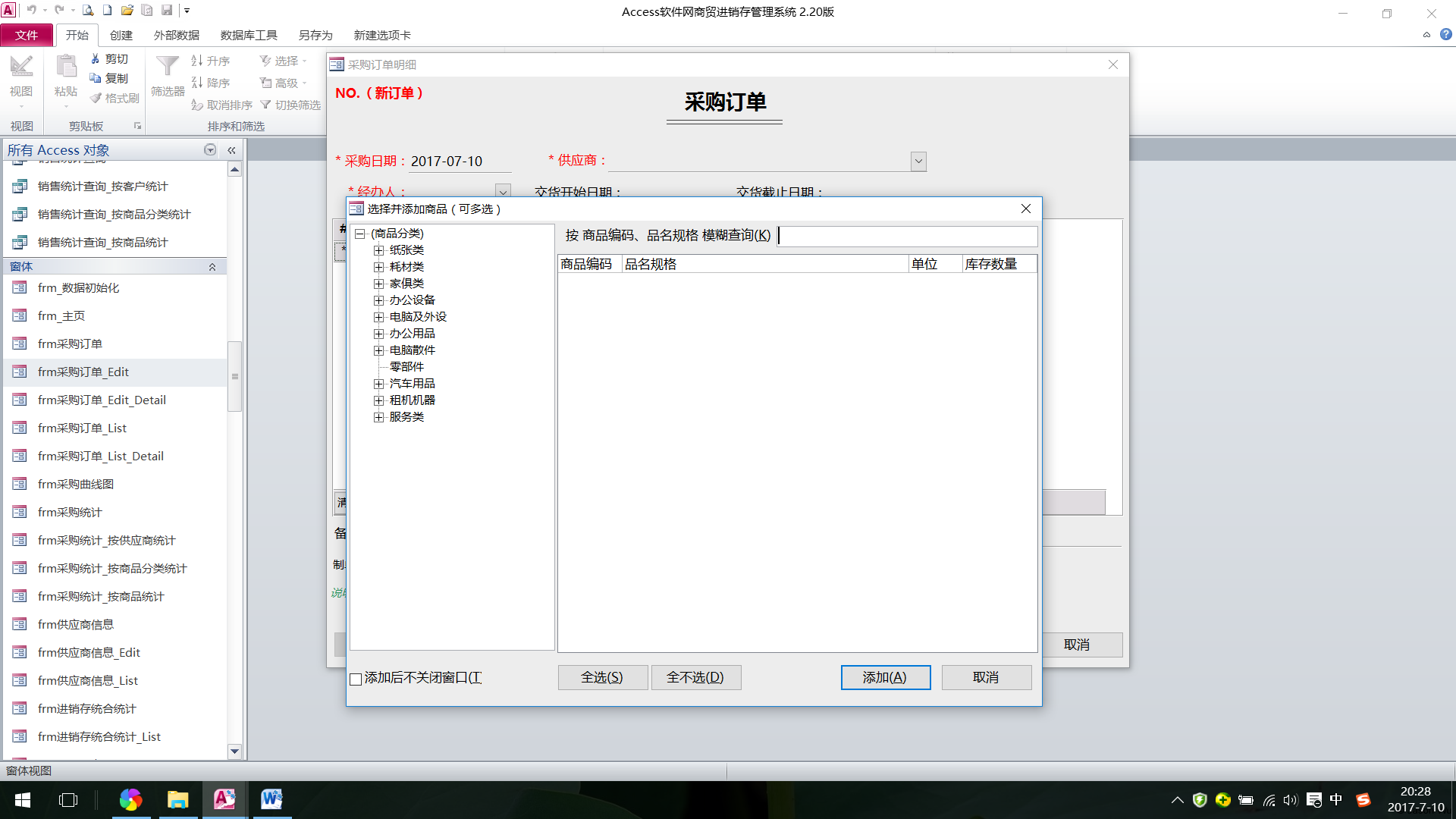
End If

Next

Me.Dirty = False

Set rst = Nothing

End Function



**frm商品选择**

txt条件

btnOk

ocx分类树

btnCancel

btn全不选

btn全选

chk添加后不关闭窗口

lst商品ID

**3.frm商品选择 窗体**

Private mstrCategoryWhere As String

Private mstrDataSourceSQL As String

Private Sub Form\_Load()

mstrDataSourceSQL = " SELECT 商品ID,商品编码,品名规格,单位,库存数量,最新进价,成本价,售价" \_

& " FROM 商品信息表"

Me.lst商品ID.RowSource = mstrDataSourceSQL & " WHERE 1=2"

LoadProductCategoriesTree Me.ocx分类树.Object

On Error Resume Next

Me.Move GetFormProperty(Me.Name, "窗体左边距", -1), \_

GetFormProperty(Me.Name, "窗体上边距", -1)

Me.chk添加后不关闭窗口 = GetFormProperty(Me.Name, "添加后不关闭窗口", True)

Me.txt条件.SetFocus

End Sub

Private Sub Form\_Close()

SaveFormProperty Me.Name, "窗体左边距", Me.WindowLeft

SaveFormProperty Me.Name, "窗体上边距", Me.WindowTop

End Sub

Private Sub ocx分类树\_LostFocus()

Set Me.ocx分类树.Drophighlight = Me.ocx分类树.SelectedItem

End Sub

Private Sub ocx分类树\_NodeClick(ByVal Node As Object)

If Node.Key <> "K" Then

mstrCategoryWhere = " 分类编号 Like '" & Mid(Node.Key, 2) & "\*'"

Me.lst商品ID.RowSource = mstrDataSourceSQL & " WHERE " & mstrCategoryWhere

Else

mstrCategoryWhere = "" '如果选择的是根节点，则清空类别条件表达式变量

Me.lst商品ID.RowSource = mstrDataSourceSQL '并将“lst商品ID”列表框的行来源设置为不带条件的数据源SQL

End If

Me.lst商品ID = Null '清空“lst商品ID”列表框的值，避免重置行来源之前的无效值造成影响

Set Me.ocx分类树.Drophighlight = Nothing '

End Sub

Private Sub chk添加后不关闭窗口\_AfterUpdate()

SaveFormProperty Me.Name, "添加后不关闭窗口", Me.chk添加后不关闭窗口 ‘保存成功返回**True-1**，否则返回**False 0**

End Sub

Private Sub txt条件\_Change()

Dim strWhere As String

strWhere = Trim(Me.txt条件.Text)

If Len(mstrCategoryWhere) > 0 Then strWhere = " AND " & mstrCategoryWhere

If Len(strWhere) > 0 Then strWhere = " AND [商品编码] & [品名规格] Like " & SQLText("\*" & strWhere & "\*")

strWhere = Mid(strWhere, 6)

If strWhere <> "" Then strWhere = " WHERE " & strWhere

Me.lst商品ID.RowSource = mstrDataSourceSQL & strWhere

End Sub

Private Sub btn全选\_Click()

DoCmd.Hourglass True

SelectAllListBoxItems Me.lst商品ID, True

DoCmd.Hourglass False

End Sub

Private Sub btn全不选\_Click()

DoCmd.Hourglass True

SelectAllListBoxItems Me.lst商品ID, False

DoCmd.Hourglass False

End Sub

Private Sub btnOK\_Click()

On Error GoTo ErrorHandler

Dim varItem As Variant

If Me.lst商品ID.ItemsSelected.Count = 0 Then

MsgBoxEx "当前未选中任何商品。", vbInformation

Exit Sub

End If

Call gfrmTemp.AddProduct(Me.lst商品ID)

If Me.chk添加后不关闭窗口 Then

Call btn全不选\_Click

Else

DoCmd.Close acForm, Me.Name, acSaveNo

End If

ExitHere:

Exit Sub

ErrorHandler:

MsgBoxEx Err.Description, vbCritical

Resume ExitHere

End Sub

Private Sub btnCancel\_Click()

On Error Resume Next

DoCmd.Close acForm, Me.Name, acSaveNo

End Sub

## 43.审核按钮

Public Sub btnAudit\_Click() ‘单击事件

On Error GoTo ErrorHandler

Call AuditPurchaseOrder(Me.sfrList!采购订单号)

ExitHere:

Exit Sub

ErrorHandler:

RDPErrorHandler Me.Name & ": btnAudit\_Click()"

Resume ExitHere

End Sub

Function AuditPurchaseOrder(BillNo As Variant) As Boolean

On Error GoTo ErrorHandler

Dim strSQL As String

Dim strMsg As String

Dim strBillNo As String

If Me.sfrList.Form.CurrentRecord < 1 Then

Exit Function

End If

If Me.sfrList!采购订单号 <> BillNo Then

Me.sfrList.Form.Recordset.MoveFirst

Me.sfrList.Form.Recordset.FindFirst "采购订单号='" & BillNo & "'"

End If

Me.sfrList.SetFocus

RunCommand acCmdSelectRecord

strBillNo = "采购订单 No." & Me.sfrList!采购订单号

strMsg = "审核之后将不能再编辑订单，确定要完成" & strBillNo & " 的审核吗？"

If MsgBox(strMsg, vbExclamation + vbOKCancel, "审核确认") = vbCancel Then

Exit Function

End If

strSQL = " UPDATE 采购订单表 SET 状态='已审核',审核时间=Now(), 审核人='@审核人'" \_

& " WHERE [采购订单号]='@采购订单号'"

strSQL = Replace(strSQL, "@审核人", GetParameter("Current User Nickname", dbText, ""))

strSQL = Replace(strSQL, "@采购订单号", Me.sfrList![采购订单号])

daorunsql strSQL

'Me.RefreshDataList

RequeryDataObject gsfrList

AuditPurchaseOrder = True

ExitHere:

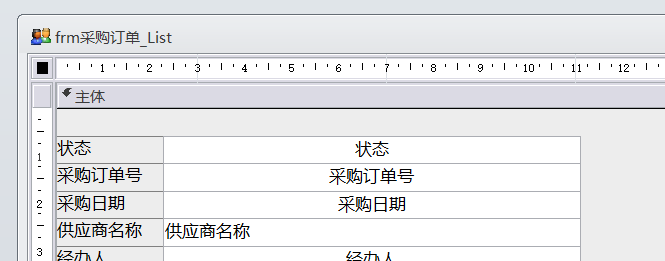
Exit Function

ErrorHandler:

MsgBoxEx Err.Description, vbCritical

Resume ExitHere

End Function



Private Sub 状态\_Click() ‘单击事件或者做成函数在所有字段的单击事件里引用

If Me.状态 = "待审核" Then

Me.Parent.Form.btnAudit.Enabled = True

Else

Me.Parent.Form.btnAudit.Enabled = False

End If

End Sub

Public Sub btnAudit\_Click() '审核

On Error GoTo ErrorHandler

Dim strSQL As String

Dim strMsg As String

Dim strBillNo As String

Dim strUsername As String

Dim strRoleName As String '自定义变量

If Me.sfrList![状态] Like "\*已审核\*" Then

MsgBoxEx "已审核过请勿重复审核！", vbInformation, "提示"

Else

strBillNo = "请购单号：" & Me.sfrList!请购单号

strMsg = "审核之后将不能再编辑单据，确定要完成" & strBillNo & " 的审核吗？"

If MsgBox(strMsg, vbExclamation + vbOKCancel, "审核确认") = vbCancel Then

Exit Sub

End If

End If

strUsername = GetParameter("Current User Nickname", dbText, "")

strRoleName = Mid(GetParameter("Current User RoleName", dbText, ""), InStrRev(GetParameter("Current User RoleName", dbText, ""), "\_") + 1)

' If Me.sfrDetail!状态 Like "\*提交审核\*" Then

strSQL = "UPDATE 请购单表 SET 状态= '" & strRoleName & "已审核" & "', 审核时间=date()" \_

& ",审核人=" & SQLText(strUsername) \_

& " WHERE [请购单号]=" & SQLText(Me.sfrList![请购单号])

DAORunSQL strSQL

Me.RefreshDataList '刷新子窗体

' End If

ExitHere:

Exit Sub

ErrorHandler:

Select Case Err.Number

Case Else

RDPErrorHandler Me.Name & ": Sub btnAudit\_Click()"

End Select

Resume ExitHere

End Sub

## 44.完成按钮

Public Sub btnFinish\_Click()

Call FinishPurchaseOrder(Me.sfrList!采购订单号)

End Sub

Function FinishPurchaseOrder(BillNo As Variant) As Boolean

On Error GoTo ErrorHandler

Dim strWhere As String

Dim strSQL As String

Dim strMsg As String

Dim strBillNo As String

Dim lngCount As Long

If Me.sfrList.Form.CurrentRecord < 1 Then

Exit Function

End If

If Me.sfrList!采购订单号 <> BillNo Then

Me.sfrList.Form.Recordset.MoveFirst

Me.sfrList.Form.Recordset.FindFirst "采购订单号='" & BillNo & "'"

End If

Me.sfrList.SetFocus

RunCommand acCmdSelectRecord

strBillNo = "采购订单 No." & Me.sfrList!采购订单号

strWhere = "[采购订单号]=" & SQLText(Me.sfrList![采购订单号]) & " AND 已入库数量 Is Null OR 已入库数量<数量"

lngCount = DCount("\*", "采购订单查询\_明细", strWhere)

If lngCount > 0 Then

strMsg = "目前还有 " & lngCount & " 种商品尚未完成全部入库目标。" & vbCrLf

End If

strMsg = strMsg & "完成订单后，将不能再对其进行入库，确定要完成" & strBillNo & " 吗？"

If MsgBox(strMsg, vbExclamation + vbOKCancel, "完成确认") = vbCancel Then

Exit Function

End If

strSQL = "UPDATE 采购订单表 SET 状态='已完成' WHERE [采购订单号]=" & SQLText(Me.sfrList![采购订单号])

DAORunSQL strSQL

Me.RefreshDataList

FinishPurchaseOrder = True

ExitHere:

Exit Function

ErrorHandler:

MsgBoxEx Err.Description, vbCritical

Resume ExitHere

End Function

## 45. 错误'2448'：您不能将数值指定给这个对象

窗体加载时，写了给文本框赋值代码，文本框还加了控件来源，就会冲突报错

**改为：txt收款金额**



保存按钮中添加：

rst![收款金额] = Me![txt收款金额]

## 46.附件改进型

附件字段——双击事件

Private Sub 附件\_DblClick(Cancel As Integer)

Set gfrmTemp1 = Me

DoCmd.OpenForm FormName:="frm\_附件", \_

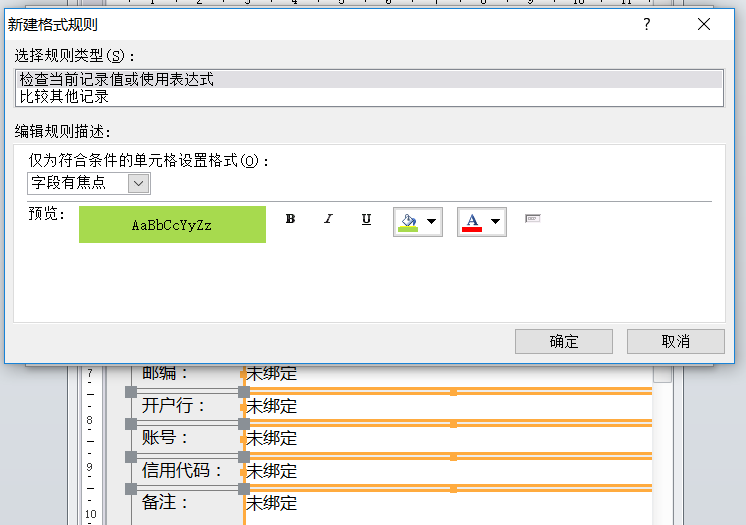
DataMode:=acFormAdd, \_

OpenArgs:=Me.员工ID

End Sub

## 47. 输入框焦点变色

通过条件格式实现，将输入框的条件格式的条件设为“字段有焦点”，然后再设置对应的格式即可，要点：可以选中所有输入框，然后一次性设置条件格式。

****

Private FocusName As Control

Public Function GetFocus()

Form.ActiveControl.BackColor = 967423

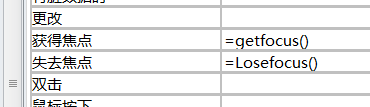
Set FocusName = Form.ActiveControl

End Function

Public Function LoseFocus()

FocusName.BackColor = 16777215

End Function



RGB颜色对照

<http://tool.oschina.net/commons?type=3>

## 48. 撤消审核

撤消审核(&U) btnUndoAudit

其过程与审核相反操作。

## 49. 三种来源的区别

RecordSource  是指窗体记录源

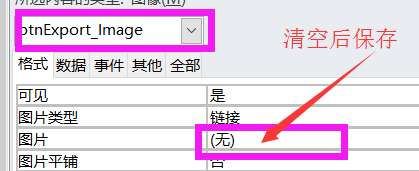
SourceObject   是指子窗体源对象  
 ControlSource  是指控件来源

strFrm = Form\_frmYg\_sg\_Main!frmChild.SourceObject    '将frmChild子窗体的源对象名称保存到strFrm变量

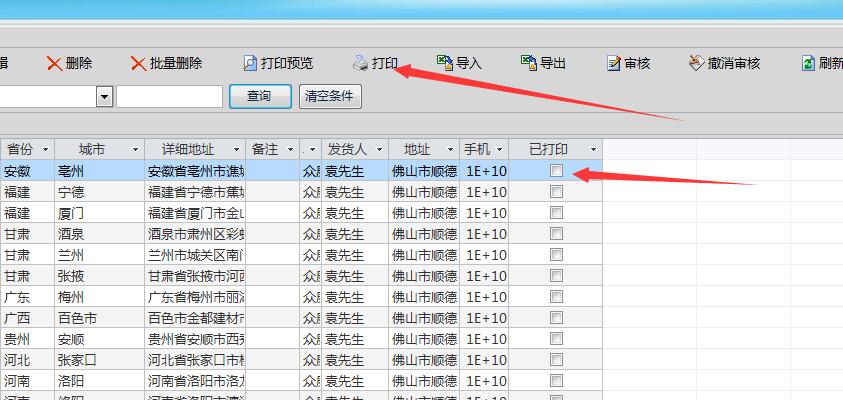
Form\_frmYg\_sg\_Main!frmChild.SourceObject = strFrm    '根据strFrm变量中保存的名称，重新加载子窗体的源对象

这两行代码其实就相当于一个通过重新加载子窗体源对象来实现刷新的功能。一般用在以下情形：  
子窗体的源对象窗体，绑定了一个查询，而现在需要通过代码去修改查询， 这时为了避免该查询的绑定窗体牌打开状态，造成查询被锁定无法成功修改，所以需要先卸载绑定窗体， 修改完查询后，再重新根据变量中记录的源对象名称重新加载绑定窗体到子窗体中

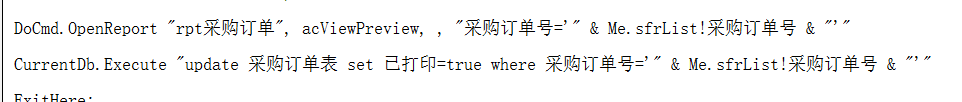
## 50. 打开窗体图标显示错误

设计视图——

## 51. 打印后标记为已打印



**打印后写更新语句**



## 52. 窗体打开后不加载数据

Private Sub Form\_Open(Cancel As Integer)

ApplyTheme Me

LoadLocalLanguage Me

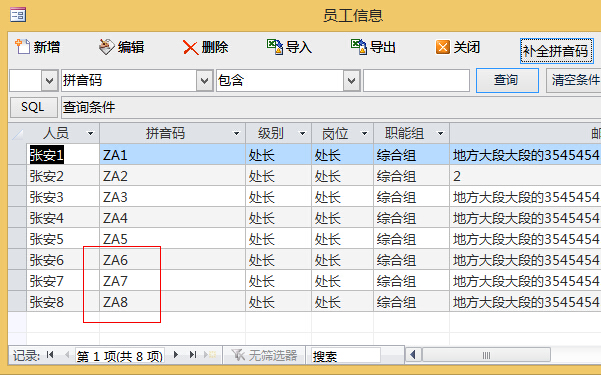
**Me.Filter = "1=2"**

**Me.FilterOn = True**

End Sub

## 53.批量更新拼音码

导入EXCEL员工信息表后，拼音码一般都要一个个再修改上去，很麻烦，但可以写个事件批量更新拼音码



frm员工信息 主窗体里加上这段代码就行了  
Private Sub 补全拼音码\_Click()  
    DoCmd.RunSQL "update tbl员工信息表 set 拼音码=pinyin(人员) where 拼音码 is null"  
    RequeryDataObject Me.sfrList  '刷新窗体  
End Sub

54. 数据查询模块

**开发接口**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **属性名** | **必需/可选** | **数据类型** | **可读写** | **说明** |
| QueryForm | 必需 | Form | 只写 | 通用查询模块所调用的界面窗体，指示一个Form对象。 |
| DataForm | 必需 | Form | 可读写 | 绑定数据的窗体，指示一个Form对象。 |
| DataSource | 可选 | String | 可读写 | 对应数据窗体的“记录源”属性，如果指定了 DataSource 属性，则会在查询时通过重新设置记录源的形式实现查询，否则将采用窗体筛选器实现查询。 |
| 语法形式举例：SELECT \* FROM 表名 @WHERE |
| 其中的 @WHERE 为固定用法，当用户点击“查询”按钮时会将其替换为对象的 WHERE 子句（如果无条件，则替换为空），并重置数据窗体的记录源进行查询。 |
| FieldName | 可选 | String | 可读写 | 主要用于指定窗体打开时显示的默认查询字段，缺省情况下默认为列表中第 1 个字段。 |
| Operator | 可选 | String | 可读写 | 主要用于指定窗体打开时显示的默认比较方式。由于更改了 FieldName 属性之后，默认会根据字段的数据类型自动指定默认比较方式，是否类型默认为“选中/是”、数字类型和日期类型默认为“等于”、文本类型和备注类型默认为“包含”，所以仅当需要设置的默认比较方式不是这些的时候，才需要设置该属性。 |
| WhereSQL | 可选 | String | 可读写 | 返回或指定要应用的查询条件对应的 SQL 语句。写入 WhereSQL 参数后将会自动执行查询。主要用于在窗体打开时加载默认的动态查询条件，例如每次打开时即默认查询当月数据。 |
| QueryConditions | 可选 | String | 可读写 | 返回或指定要应用的查询条件对应的说明字串，即显示在界面中的查询条件。和 WhereSQL 属性配合使用，应用于窗体打开时加载默认的动态查询条件，并显示相应的说明文字。 |
| FixedWhereSQL | 可选 | String | 可读写 | 固定条件表达式的 SQL 语句，设置后该条件不能被清空而始终有效。 |
| FixedQueryConditions | 可选 | String | 可读写 | 同 QueryConditions 属性，FixedWhereSQL 属性配合使用。 |
| InstantSearch | 可选 | Boolean | 可读写 | 是否启用即时搜索功能，默认情况下使用“我的设置”中的配置，也可以在调用时另行指定。 |

|  |  |
| --- | --- |
| **方法名** | **说明** |
| AddAllField | 添加数据窗体中所有绑定了控件的字段，如果记录源中有此字段，但是未绑定到任何控件，则该字段不会被添加。控件来源是以等号“=”开头的计算表达式，因为不是字段，所以也不会被添加。 |
| AddField | 自定义指定要用于查询的字段，一次添加一个字段，如果要添加多个字段，请多次调用AddField方法。 |
| 参数：DataType、Name、Caption DataType 字段的数据类型（fdtYesNo：是否型字段，fdtNumeric：数字型字段，fdtDate日期型字段， fdtText：短文本型字段，fdtMemo：长文本型（备注型）字段 Name 字段名称 Caption 字段标题，向用户显示的字段名称，因为字段名称很可能并不是以自然语言命名的，所以可以 指定标题来向用户显示更友好的名称，缺省情况下自动默认为和 Name 属性一致。 |
| **RunQuery** | 执行查询，相当于在界面中手动点击“查询”按钮。 |
| ClearFields | 清空字段列表，用于在不重新打开窗体，但变更了数据源时重新设定查询字段，需要先清空字段列表，再使用 AddAllFields 或 AddField 方法重新添加字段。 |
| ClearConditions | 清空查询条件，相当于在界面中手动点击“清空条件”按钮。 |

**回调功能**

**提示** 根据大量的实际使用反馈，初学者在“回调”这一概念上不容易理解，因此在 2.0.0 版本引入了 FixedWhereSQL 和 FixedQueryConditions 属性来指定固定查询条件，代替回调功能。回调功能的用法仍然保留用于兼容旧的版本，但不建议再使用。

所谓回调，即点击“查询”按钮后，不自动进行查询，而是仅通过查询模块取得生成的查询条件语句，再通过回调函数，由开发者自行决定要如何对其进行应用。这在一些需要始终附加额外的查询条件的场景中特别有用，例如下面的示例中，普通用户只能查看他自己录入的订单，而管理员可以查看所有订单：

Private Sub Form\_Load()

Set mclsQuery = New GeneralQuery

With mclsQuery

.QueryForm = Me.sfrQuickQuery.Form

.DataForm = Me.sfrList.Form

.AddField fdtText, "订单号"

.AddField fdtText, "客户名称"

.AddField fdtDate, "销售时间"

End With

End Sub

'回调函数的名称必需是RunQuery，必须且只能有一个String型的参数。

Function RunQuery(ByVal WhereSQL As String)

If WhereSQL <> "" And Forms!SysFrmMain!txtRoleName <> "管理员" Then

WhereSQL = WhereSQL & " AND 销售员='" & Forms!SysFrmMain!txtRoleName & "'"

End If

If WhereSQL <> "" Then

WhereSQL = " WHERE " & WhereSQL

End If

Me.sfrList.Form.RecordSource = "SELECT \* FROM 订单表" & WhereSQL

End Function

**示例**

1）创建空白主窗体 frmXXX、数据表窗体 frmXXX\_List；  
2）在frmXXX窗体中创建2个子窗体，分别命名为sfrQuickQuery 、sfrList；  
3）将sfrQuickQuery子窗体的“源对象”属性设为“SysFrmGeneralQuickQuery”，sfrList子窗体的“源对象”属性设为“frmXXX\_List”；  
4）在frmXXX窗体中添加如下代码：

Private Sub Form\_Load()

Set mclsQuery = New GeneralQuery

With mclsQuery

.QueryForm = Me.sfrQuickQuery.Form '指定查询窗体，必需的，且必须放在最前面

.DataForm = Me.sfrList.Form '指定数据窗体，必需的，且必须放在最前面

' .AddAllFields '如果不需要特别指定字段，则可以用AddAllFields方法自动添加所有绑定字段

.AddField fdtText, "公司名称" '只想让一部分字段可以用于查询时用AddField方法

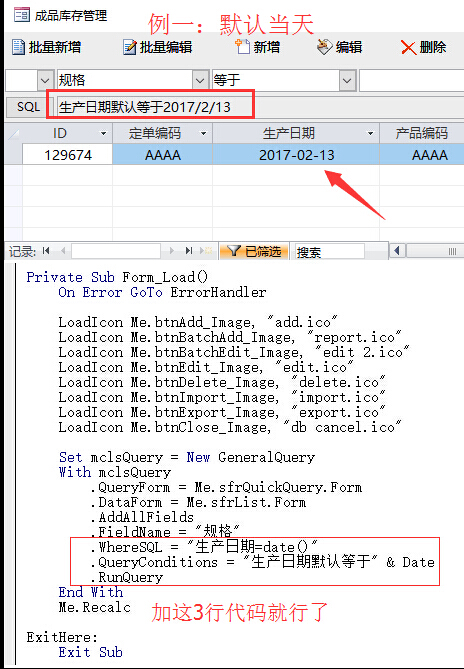
.AddField fdtNumeric, "联系人"

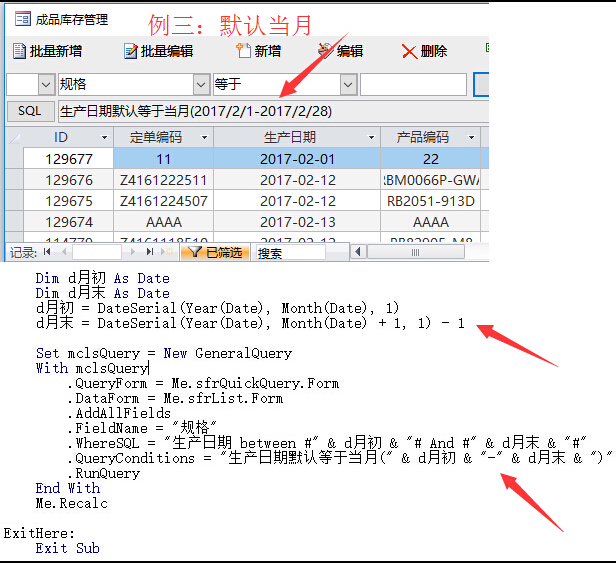
.FieldName = "公司名称" '指定自定义的默认查询字段

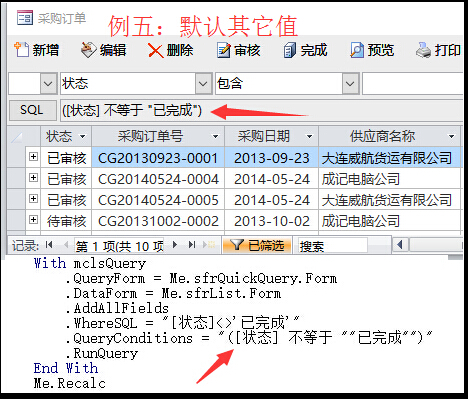
End With

End Sub

## 55. 通用查询如何实现默认当天或当月/本月或默认其它指定值







代码整理如下：

**例一**

 .WhereSQL = "生产日期=date()"  
 .QueryConditions = "生产日期默认等于" & Date  
 .RunQuery  
 **例二**  
 .WhereSQL = "生产日期=#2017-02-13#"  
 .QueryConditions = "生产日期默认等于2017-02-13"  
 .RunQuery

**例三**  
    Dim d月初 As Date  
    Dim d月末 As Date  
    d月初 = DateSerial(Year(Date), Month(Date), 1)  
    d月末 = DateSerial(Year(Date), Month(Date) + 1, 1) - 1  
      
    Set mclsQuery = New GeneralQuery  
    With mclsQuery  
        .QueryForm = Me.sfrQuickQuery.Form  
        .DataForm = Me.sfrList.Form  
        .AddAllFields  
        .FieldName = "规格"  
        .WhereSQL = "生产日期 between #" & d月初 & "# And #" & d月末 & "#"  
        .QueryConditions = "生产日期默认等于当月(" & d月初 & "-" & d月末 & ")"  
        .RunQuery  
    End With

**例四**  
 .FixedWhereSQL = "[bxrq]>=DateAdd('d',-60,Date()) "  
 .FixedQueryConditions = "([报销日期]在60天内记录)"  
 .RunQuery

**例五**  
 .WhereSQL = "[状态]<>'已完成'"  
 .QueryConditions = "([状态] 不等于 ""已完成"")"  
 .RunQuery

## 56. 指定列表框或组合框中默认值的所在行

在“默认值”属性框中，键入 [组合框名称].ItemData(n)，其中 n 是要作为默认值的所在行号。ItemData 属性是从零开始的，所以如果要使用列表中第一行为默认值，请键入：ItemData(0)。例如，如果要指定“SupplierCombo”组合框的第一列为组合框的默认值，请在“默认值”属性框中键入 [SupplierCombo].ItemData(0)。

## 57. 智能筛选条件使用变量

Private Sub 供应商\_AfterUpdate()

Dim str As String

str = Me.供应商 ‘同是智能筛选

mclsSC负责人.Init Combo:=Me.负责人, \_

SearchField:="员工信息表.员工姓名 & 员工信息表.YG拼音码", \_

SQLSELECT:="员工信息表.员工ID, 员工信息表.员工姓名, 单位信息表.单位ID", \_

SQLFROM:="单位信息表 RIGHT JOIN 员工信息表 ON 单位信息表.单位ID = 员工信息表.单位名称 ", \_

SQLWHERE:="(员工信息表.离职=False) and [单位信息表].[单位ID] = '" & str & " ' ", \_

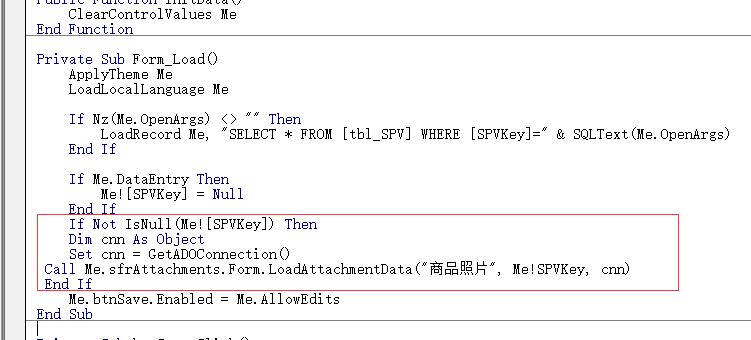
SQLORDERBY:="单位信息表.单位ID"

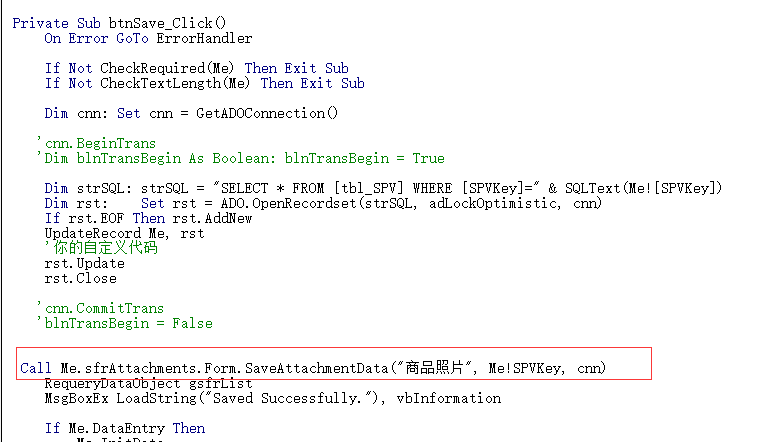
End Sub

## 58. 通用附件模块使用

编辑窗体中添加一个子窗体控件，控件名修改为“sfrAttachments”，其“源对象”属性设为“SysFrmAttachments”，然后在加载和保存数据时，添加一行调用代码。

在对应的编辑窗体加载和保存按钮添加的代码如图所示





## 59. 树控件报错

一个很奇怪的错误，树控件不停的提示鼠标事件出错，好多方法都没用，可以试着在所在窗体添加上下边的代码

    Private Sub Form\_Error(DataErr As Integer, Response As Integer)  
        Response = -1  
    End Sub