

目录

[SQL语句教程] SQL语言教程 - 序 -----	[2]
[SQL语句教程] Workbooks对象的Open方法的帮助说明 -----	[3]
[SQL语句教程] Workbooks对象与Sheets和Worksheets对象的Add方法 -	[5]
[SQL语句教程] Worksheet对象的Delete方法 -----	[6]
[SQL语句教程] 利用ADO的Connection对象来连接数据库 -----	[7]
[SQL语句教程] CREATE TABLE - 创建数据表的语句 -----	[9]
[SQL语句教程] DROP TABLE - 删除数据表的语句 -----	[11]
[SQL语句教程] Alter Table - 修改数据表的语句 -----	[13]
[SQL语句教程] INSERT INTO - 向数据库中添加数据 -----	[15]
[SQL语句教程] UPDATE - 修改数据库中已有的数据 -----	[17]
[SQL语句教程] DELETE FROM - 删除数据库的数据 -----	[19]
[SQL语句教程] CopyFromRecordset方法的使用说明 -----	[20]
[SQL语句教程] Select - 从数据库中检索数据 -----	[21]
[SQL语句教程] Where - 筛选与限制检索的数据 -----	[23]
[SQL语句教程] SQL函数 - 预处理检索值的命令 -----	[26]
[SQL语句教程] Group By - 分类汇总检索的数据 -----	[28]
[SQL语句教程] Having - 筛选汇总后的数据 -----	[30]
[SQL语句教程] Order By - 排序检索的数据 -----	[32]
[SQL语句教程] *(星号) - 数据表中所有的列 -----	[34]
[SQL语句教程] Insert Into Select - 批量添加数据到数据库中 -----	[36]
[SQL语句教程] Select Into - 把检索的数据添加到新的数据表中 -----	[38]
[SQL语句教程] Top - 限制检索结果的数量 -----	[40]
[SQL语句教程] Distinct - 筛选出不重复的数据 -----	[42]

SQL语言教程 - 序

特此声明本教程作者：bengdeng，来源：Excel吧

我作为一学习者，整理了一下，去除一些无用的字句&广告性质的网址。

对于不少的朋友来说，Excel可能还是在用最基本的功能，但就是最基本的功能，已能帮助我们完成很多的工作，一般的Excel网站，把Excel知识分成——基础知识，公式与函数，图表与数据透视表和VBA程序开发。

其它方面在这不谈，Excel就是因为用VBA，这个平民化的编程平台，而让广大的使用者开发出了相当多的实用小程序，甚至是仅仅用录制宏和稍稍的修改，就可以做成一段非常有用的小程序。就是因为这个吧，连WPS从2005版都不惜重写代码来支持VBA，背着模仿者的骂名。

而VBA的强大，还不仅仅如此，Excelhome的VBA程序板块中的这句话，应该更能让你体会到其中的意义——认识宏，学习从Excel VBA基础入门到设计制作模板、加载宏，交流各种VBA方法、VBA属性、VBA事件、窗体控件、Excel对象的使用技巧，以及借助API、ADO、SQL构建完整软件系统。

把Excel当成是数据库软件是不正确的，也是不公平的。它有它相对于数据库易上手，易处理，可视化，简单易用的优点，但也有数据库没有的对大量数据处理，与数据与数据间关系系统和引用不便的缺点。虽然Excel2007版的单个工作表文件已增加了行与列，但对于大量的数据处理，还是远远不如数据库的。

因此，由于最近的工作需要，接触与了解了一点SQL与数据库方面的知识，所以就以Excel为程序的前台，用来录入与输出数据或报表，而利用ACCESS数据库来保存数据，最后用VBA+SQL与ADO来处理数据与联接Excel与数据库，解决了以前对大量数据，多人同时处理数据，数据间统计，对比，引用，关联等诸多以前没办法解决的难题。因此就把自己使用的心得与大家一起分享，一起学习进步。

千里之行，始于足下，接下来的时间，我用不是很正规，也不是很专业，也不是很严肃，更不是很深奥的语言，来试着带领大家进入这个看似与Excel无关的世界，而这第一步，我们要认识的，就是要认识SQL语言中各个SQL语法，SQL函数与SQL指令。而下一篇文章，我们要先介绍下，怎么用SQL接连数据库，而同样的方法也可以接连Excel文件。

Workbooks对象的Open方法的帮助说明

下面的这些内容，在Excel的VBA帮助中可以找到，写在这的目的，是为了在说明SQL连接Excel文件时，让来这的网友可以方便的找到，最近在Excel吧的VBA教程栏目里，可能会多了这些文章，目的就是让有一样相同或类似的东西放在一起，让大家横向对比。

Workbooks 对象的 **Open** 方法：作用是打开一个工作簿。其语法为：

expression.Open(FileName, UpdateLinks, ReadOnly, Format, Password, WriteResPassword, IgnoreReadOnlyRecommended, Origin, Delimiter, Editable, Notify, Converter, AddToMru, Local, CorruptLoad)

其中：

expression 必选。该表达式返回一个 **Workbooks** 对象。

FileName **String** 类型，必需。要打开的工作簿的文件名。

UpdateLinks **Variant** 类型，可选。指定文件中链接的更新方式。如果省略本参数，则提示用户选择链接的更新方式。否则，该参数的取值应为下表中的某个值。

值 含义

- 0 不更新任何引用。
- 1 更新外部引用，但不更新远程引用。
- 2 更新远程引用，但不更新外部引用。
- 3 同时更新远程引用和外部引用。

ReadOnly **Variant** 类型，可选。如果该值为 **True**，则以只读模式打开工作簿。

Format **Variant** 类型，可选。如果 **Microsoft Excel** 正在打开一个文本文件，则该参数用于指定分隔字符，如下表所示。如果省略本参数，则使用当前的分隔符。

值 分隔符

- 1 制表符
- 2 逗号
- 3 空格
- 4 分号
- 5 没有分隔符
- 6 自定义字符（请参阅 **Delimiter** 参数）

Password **Variant** 类型，可选。该字符串指定打开一个受保护工作簿的密码。如果省略该参数并且指定工作簿已设置密码，则提示用户输入密码。

WriteResPassword **Variant** 类型，可选。该字符串为一个写保护工作簿的写入权密码。如果省略该参数并且指定工作簿已设置密码，则提示用户输入密码。

IgnoreReadOnlyRecommended **Variant** 类型，可选。如果该值为 **True**，则设置 **Microsoft Excel** 不显示建议只读消息（如果该工作簿以“建议只读”选项保存）。

Origin **Variant** 类型，可选。如果该文件为文本文件，则该参数用于指示该文件来源于何种操作系统（以便正确映射代码页和回车/换行 (CR/LF)）。可为以下 **XIPlatform** 常量之一：**xlMacintosh**、**xlWindows** 或 **xlMSDOS**。如果省略本参数，则使用当前操作系统。

Delimiter **Variant** 类型，可选。如果该文件为文本文件并且 **Format** 参数为 **6**，则此参数用于指定用作分隔符的字符。例如，可使用 **Chr(9)** 代表制表符，使用 “,” 代表逗号，使用 “;” 代表分号或者使用自定义字符。如果该参数为字符串，则只使用该字符串的第一个字符。

Editable **Variant** 类型，可选。如果该文件为 **Microsoft Excel 4.0** 加载宏，则该参数的值为 **True** 时可打开该加载宏以便在窗口中看到。如果该参数的值为 **False** 或者省略该参数，则该加载宏以隐藏方式打开，并且无法设为可见。本选项不能应用于由 **Microsoft Excel 5.0** 或更高版本的 **Microsoft Excel** 创建的加载宏。如果该文件是 **Excel** 模板，则参数的值为 **True** 时，会打开指定模板用于编辑。参数为 **False** 时，可

根据指定模板打开新的工作簿。默认值为 **False**。

Notify Variant 类型，可选。当该文件不能以可读写模式打开时，如果该参数的值为 **True**，则可将该文件添加到文件通知列表。**Microsoft Excel** 将以只读模式打开该文件并轮询文件通知列表，当文件通知列表中的该文件可用时通知用户。如果该参数的值为 **False** 或省略该参数，则不请求任何通知，并且不能打开任何不可用的文件。

Converter Variant 类型，可选。打开文件时试用的第一个文件转换器的索引号。首先使用的是指定的文件转换器：如果该转换器不能识别此文件，则试用所有的转换器。转换器索引号由 **FileConverters** 属性返回的转换器行号组成。

AddToMru Variant 类型，可选。如果该值为 **True**，则将该工作簿添加到最近使用的文件列表中。默认值为 **False**。

Local Variant 类型，可选。如果该值为 **True**，则以 **Microsoft Excel**（包括控制面板设置）的语言保存文件。如果该值为 **False**（默认值），则以 **Visual Basic for Applications (VBA)** 的语言保存文件，其中 **Visual Basic for Applications (VBA)** 为典型安装的美国英语版本，除非 VBA 项目的 **Workbooks.Open** 来自旧的国际化的 **XL5/95 VBA** 项目。

CorruptLoad Variant 类型，可选。可为以下常量之一：**xlNormalLoad**、**xlRepairFile** 和 **xlExtractData**。如果未指定任何值，则默认值通常为普通状态，但如果 **Excel** 已尝试打开该文件，则可以是安全加载或数据恢复状态。首选值为普通状态。如果 **Excel** 在打开文件时停止操作，则为安全加载状态。如果 **Excel** 再次停止操作，则为数据恢复状态。

示例

本示例打开 **Analysis.xls** 工作簿，然后运行 **Auto_Open** 宏。

```
Workbooks.Open "ANALYSIS.XLS"
```

看起来 **Open** 方法的参数很多，其实经常用的不多，除了必须的 **FileName** 外，最常用的就是 **Password**，用来打开包含有打开权限密码的 **Excel** 文件，下面的例子就是打开密码为“123”的 **ANALYSIS.XLS** 文件：

```
Workbooks.Open "ANALYSIS.XLS", Password:="123"
```

Workbooks对象与Sheets和Worksheets对象的Add方法

上个VBA教程中介绍了Workbooks对象的Open方法，目的是在介绍如果用SQL连接Excel带密码中时，需要用到这个方法，而今天也是因为下一篇SQL语言教程的文章需要而介绍Add方法，以便大家对比一下。下面的这些内容，同样也可以在Excel的VBA帮助中找到。

一、Workbooks 对象的 Add 方法：新建工作簿，新建的工作簿将成为活动工作簿。语法是：

expression.Add(Template)

expression : 必需。该表达式返回一个 Workbooks 对象。

Template : Variant 类型，可选。确定如何创建工作簿。

1、如果本参数为指定一现有 Microsoft Excel 文件名的字符串，那么创建新工作簿将以该指定的文件作为模板。

2、如果本参数为常量，新工作簿将包含指定类型的单张工作表。可为以下 XIWBATemplate 常量之一：xlWBATChart、xlWBATExcel4IntlMacroSheet、xlWBATExcel4MacroSheet 或 xlWBATWorksheet。

3、如果省略本参数，Microsoft Excel 将创建包含一定数目的空白工作表的工作簿（该数目由 SheetsInNewWorkbook 属性设置）。

给出一个最常用也最常见的例子就是，本示例新建一个工作簿。

Workbooks.Add

二、Sheets 和 Worksheets 对象的 Add 方法：新建工作表、图表或宏表。新建的工作表将成为活动工作表。语法是：

expression.Add(Before, After, Count, Type)

expression : 必需。该表达式返回上面的对象之一。

Before : Variant 类型，可选。指定工作表对象，新建的工作表将置于此工作表之前。

After : Variant 类型，可选。指定工作表对象，新建的工作表将置于此工作表之后。

Count : Variant 类型，可选。要新建的工作表的数目。默认值为 1。

Type : Variant 类型，可选。指定工作表类型。

1、Type 可为以下 XISheetType 常量之一：xlWorksheet、xlChart、xlExcel4MacroSheet 或 xlExcel4IntlMacroSheet。

2、如果要基于现有模板插入工作表，则指定该模板的路径。

3、默认值为 xlWorksheet。

下面的示例是活动工作簿的最后一张工作表之前插入一个新的工作表。

ActiveWorkbook.Sheets.Add Before:=Worksheets(Worksheets.Count)

Worksheet对象的Delete方法

上篇上个VBA教程中介绍了Workbooks对象与Sheets和Worksheets对象的Add方法，这一篇就要来说Delete，相对于Add方法，Delete方法是很简单了，也没有参数，其语法是：

expression.Delete(Shift)

expression ： 必需。该表达式返回一个 **Worksheet** 对象。

唯一要说明的是，当要删除一个非空的工作簿或工作表时，此方法将显示提示用户确认删除的对话框。此对话框将按默认方式显示。当调用 **Workbook** 或 **Worksheet** 对象时，**Delete** 会返回一个 **Boolean** 值，如果用户单击“取消”，则该值将返回“**False**”；如果用户单击“删除”，则该值将返回“**True**”。

如果我们要让它不出现这个提示，可以把**Application**对象的**DisplayAlerts**属性，设定为**False**即可，下面这段代码，是删除当前工作表，且不出现提示！运行时请注意，当前工作表不要有需要的数据存在！！！！

```
Sub Excelba()  
Application.DisplayAlerts = False  
ActiveSheet.Delete  
Application.DisplayAlerts = True  
End Sub
```

关于Delete的说明就是这么简单，应该不会很难理解吧*^_^*

利用ADO的Connection对象来连接数据库

在SQL语言教程 - 序中我们说到这一篇要介绍利用ADO的Connection对象来连接数据库，目地就是让大家了解后，就可以便于以后的其它SQL语言教程里的代码，大家可以动手运行一下，看一下效果，之后的大部份例子中，这篇文章介绍的都是要用到的，就像我们要使用Excel文件时，要先用Workbooks对象的Open方法打开Excel文件一样。

下面进入正题。首先要说的是ADO是什么？ADO的全称是：Microsoft ActiveX Data Objects，它使您的客户端应用程序能够通过 OLE DB 提供者访问和操作数据库服务器中的数据。它的主要优点是易于使用、速度快、内存支出低和占用磁盘空间少。ADO 支持用于建立客户端/服务器和基于 Web 的应用程序的主要功能。

第二个问题是如何使用ADO？ADO有不少版本，不同的电脑里可能就有不同的版本，在VBA中使用ADO的方法是，在VBE编辑器中工具菜单的引用项里，引用Microsoft activex date objects x.x，其中x.x为版本号，可能会因为你安装的office的版本不同而不同，我用的例子，大多引用了2.5版。

认识了ADO，那开始说Connection 对象，Connection 对象表示数据源的唯一会话。ADO提供这个对象，来让我们连接数据库。

而Connection 对象的ConnectionString 属性，就是用来指示用于建立到数据源的连接的信息。ConnectionString 属性有五个参数：

参数 | 说明

Provider= : 指定用于连接的提供者的名称。

File Name= : 指定提供者特有的文件（例如，持久保留的数据源对象）的名称，这些文件中包含预置的连接信息。

Remote Provider= : 指定当打开客户端连接时使用的提供者的名称。（仅限于远程数据服务。）

Remote Server= : 指定当打开客户端连接时使用的服务器的路径名称。（仅限于远程数据服务。）

URL= : 指定连接字符串为标识资源（如文件或目录）的绝对 URL。

设置 ConnectionString 属性后，就可以用 Connection 对象Open方法来接连数据库了。

能看到这，可能你已有点晕了，不过不要紧，下面的几个例子，就能让你更好地理解上面这样比较无味的文字，首先是一段连接ACCESS数据库的程序。

Sub 连接进销存表数据库()

'作者: bengdeng

'功能: 连接同一目录下的进销存表数据库文件

'注意: 要在工具/引用中引用microsoft activex date objects x.x

' 其中x.x为版本号，可能会因为你安装的office的版本不同而不同，本例引用了2.5版

Dim conn As ADODB.Connection

Dim WN As String

Dim sSql As String

WN = "进销存表.mdb"

Set conn = New ADODB.Connection

conn.ConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;" & _

"Data Source=" & ThisWorkbook.Path & "\" & WN _

'& ";Jet OLEDB:Database Password=" & "123"

conn.Open

If conn.State = adStateOpen Then

MsgBox "连接成功!"

conn.Close

End If

Set conn = Nothing

End Sub

上面程序的红色部分，是为了打开带密码的Access的数据库文件时，需要指定密码的代码，下面再来一段程序，来连接一个Excel文件！

Sub 连接进销存表()

'作者: bengdeng

'功能: 连接同一目录下的进销存表文件

'注意: 要在工具/引用中引用microsoft activex date objects x.x

' 其中x.x为版本号, 可能会因为你安装的office的版本不同而不同, 本例引用了2.5版

Dim conn As ADODB.Connection

Dim WN As String

Dim sSql As String

WN = "进销存表.xls"

Set conn = New ADODB.Connection

conn.Open "Provider=Microsoft.Jet.Oledb.4.0;" & _

"Extended Properties=Excel 8.0;" & _

"Data Source=" & ThisWorkbook.Path & "\" & WN

If conn.State = adStateOpen Then

MsgBox "连接成功!"

conn.Close

End If

Set conn = Nothing

End Sub

上面的与第一段程序对比, 需要多设定一个参数——“Extended Properties”, 这是指定Excel文件的版本, 现在已经有的版本为5.0、7.0、8.0等, 分别对应的是95版, 97版与2000~2003版的XLS文件格式, 我用的是2003版, 所以设定为=Excel 8.0。

最后需要说明的是, 第二段程序是不能像第一段程序中, 用Jet OLEDB:Database Password="密码"来打开与连接带有密码的Excel文件的, 如果需要处理这样的文件, 就要用Workbooks对象的Open方法先打开这个Excel文件后再处理。下面的这个程序就是这样的例子:

Sub 连接带密码进销存表()

'作者: bengdeng

'功能: 连接同一目录下的打开文件密码为“123”的进销存表文件

'注意: 要在工具/引用中引用microsoft activex date objects x.x

' 其中x.x为版本号, 可能会因为你安装的office的版本不同而不同, 本例引用了2.5版

Dim conn As ADODB.Connection

Dim WN As String

Dim sSql As String

WN = "进销存表.xls"

Workbooks.Open ThisWorkbook.Path & "\" & WN, Password:="123"

Set conn = New ADODB.Connection

conn.Open "Provider=Microsoft.Jet.Oledb.4.0;" & _

"Extended Properties=Excel 8.0;" & _

"Data Source=" & WN

If conn.State = adStateOpen Then

MsgBox "连接成功!"

conn.Close

End If

Workbooks(WN).Close False

Set conn = Nothing

End Sub

这篇文章到这就完了, 有点长, 有三个程序, 大家可以复制到VBA编辑器中, 运行一下代码, 好好的对比与理解一下, 相信, 很快你就会踏这第一步了, 而后面的世界还很多精彩, 看到这的你, 应该可以看出来一点点SQL与ADO与Excel的联系了, 那就让我们一起继续向行吧!!

CREATE TABLE - 创建数据表的语句

数据表是数据库的基本架构，就像Excel文件中的工作表一样，在Excel中我们可以用ADD方法来创建新的工作表，而SQL语言里，CREATE TABLE语句就是用来创建数据表的。

在说明CREATE TABLE语句的语法之前，我们再了解一些相关的知识。Excel工作表中对应有列与行，而在数据库中，对应称为Column与Row，对于这两个单词，应该用过VBA的人都不会陌生，这也是Excel中VBA里列与行的写法。不同的是下面，在数据库中多少列是在创建表时就有设定的，虽然以后还有可能增加，而且在设定时还要规定整列的数据类型，同时也意味着整列的数据类型都是一样的；这个在Excel中是没有这样的规则，而且Excel的最大行与列是由Excel本身决定的，这也是数据库与电子表格对数据约束最大的不同。

而数据库具体有哪些数据类型呢？以ACCESS为例，有存贮日期类型的DATETIME；有存贮数值类型的FLOAT，SMALLINT，INTEGER等，有字符串型的CHAR等等。不同的数据库可能支持不同的数据类型，因此在使用时应该参考一下数据库在这方面的说明。

了解了上面的信息，下面开始说下CREATE TABLE语句的语法：

CREATE TABLE 表格名(列名1 列名1的数据类型,列名2 列名2的数据类型,...)

注意在列名与数据类型中间有一个空格，在VBA中，我们可以利用ADO的Execute方法，来运行SQL语句，下面我们就用CREATE TABLE 来创建一个进销存表数据库的三个数据表——明细表，进仓表与出仓表；其中明细表有5列，分别为物品名称（字符串型），结余日期（日期型），结余数量（双精度型），进仓数量（双精度型），出仓数量（双精度型）；进仓表有3列，分别为进仓日期（日期型），物品名称（字符串型），进仓数量（双精度型）；出仓表有3列，分别为出仓日期（日期型），物品名称（字符串型），出仓数量（双精度型）。代码如下：

Sub 创建进销存表数据库()

'作者: bengdeng

'功能: 在程序文件同一目录创建进销表一个进销存表数据库

'注意: 要在工具/引用中引用microsoft activex date objects x.x

' 其中x.x为版本号，可能会因为你安装的office的版本不同而不同，本例引用了2.5版

' 由于要创建一个新的数据库，还要引用microsoft ado ext.2.x for ddl and security

' 利用ADOX的Create方法来创建，其中2.X为版本号，本例引用了2.8版

Dim MyCat As ADOX.Catalog

Dim conn As ADODB.Connection

Dim WN As String

Dim sSql As String

WN = "进销存表.mdb"

Set MyCat = New ADOX.Catalog

MyCat.Create "Provider=Microsoft.Jet.Oledb.4.0; _

Data Source=" & ThisWorkbook.Path & "\" & WN

Set conn = New ADODB.Connection

conn.ConnectionString = MyCat.ActiveConnection

conn.Open

If conn.State = adStateOpen Then

 sSql = "CREATE TABLE 明细表 (物品名称 Char(255)," & _
 " 结余日期 Date, 结余数量 Float," & " 进仓数量 Float, 出仓数量 Float)"

 conn.Execute sSql

 sSql = "CREATE TABLE 进仓表 (进仓日期 Date, 物品名称 Char(255), 进仓数量 Float)"

 conn.Execute sSql

 sSql = "CREATE TABLE 出仓表 (出仓日期 Date, 物品名称 Char(255), 出仓数量 Float)"

 conn.Execute sSql

 MsgBox "创建数据库成功!" & vbCrLf & "数据库文件名为: " & WN & vbCrLf & _

 "保存位置: " & ThisWorkbook.Path

 conn.Close

End If

Set conn = Nothing

End Sub

因为要创表一个新的数据库，所以使用ADOX的Create 方法创建一个新的数据库。

下面再给出一段创建类似于上面内容的Excel文件。

Sub 创建进销存表文件()

'作者: bengdeng

'功能: 在程序文件同一目录创建进销表一个进销存表文件

'注意: 要在工具/引用中引用microsoft activex date objects x.x

' 其中x.x为版本号，可能会因为你安装的office的版本不同而不同，本例引用了2.5版

Dim conn As ADODB.Connection

Dim WN As String

Dim sSql As String

WN = "进销存表.xls"

Set conn = New ADODB.Connection

conn.Open "Provider=Microsoft.Jet.Oledb.4.0;" & _

"Extended Properties=Excel 8.0;" & _

"Data Source=" & ThisWorkbook.Path & "\" & WN

If conn.State = adStateOpen Then

sSql = "CREATE TABLE 明细表 (物品名称 Char(255)," & _

" 结余日期 Date, 结余数量 Float," & _

" 进仓数量 Float, 出仓数量 Float)"

conn.Execute sSql

sSql = "CREATE TABLE 进仓表 (进仓日期 Date, 物品名称 Char(255), 进仓数量 Float)"

conn.Execute sSql

sSql = "CREATE TABLE 出仓表 (出仓日期 Date, 物品名称 Char(255), 出仓数量 Float)"

conn.Execute sSql

MsgBox "创建文件成功!" & vbCrLf & "数据库文件名为: " & WN & vbCrLf & _

"保存位置: " & ThisWorkbook.Path

conn.Close

End If

Set conn = Nothing

End Sub

有意思的是，创建Excel文件，不需要理会这个文件是否存在，如果存在就会在旧的文件中增加工作表，而不存在则会自动创建，不需要像ACCESS还用利用Create 方法。

CREATE TABLE 就介绍说明到这，现在大家动手复制一下上面的两段代码，运行一下，再对比一下，最后再理解一下，相信很快就会明白的哦*~*~*。

DROP TABLE - 删除数据表的语句

上一篇SQL语言教程是说创建数据表的语句CREATE TABLE，而这篇，是要介绍删除数据表的语言DROP TABLE。在Excel中，我们删除工作表的方法是Delete，而DROP TABLE的语法，也比CREATE TABLE 的语法简单多了：

DROP TABLE 表格名

简单得不需要再说明的吧*^_^*，下面我们就举个删除由上篇文章生成的进销存表中的出仓表，感受一下DROP TABLE的使用方法：

Sub 删除数据库中的出仓表()

'作者: bengdeng

'功能: 在程序文件同一目录下进销存表数据库里的出仓表

'注意: 要在工具/引用中引用microsoft activex date objects x.x

' 其中x.x为版本号，可能会因为你安装的office的版本不同而不同，本例引用了2.5版

Dim conn As ADODB.Connection

Dim WN As String

Dim TableName As String

Dim sSql As String

WN = "进销存表.mdb"

TableName = "出仓表"

Set conn = New ADODB.Connection

conn.ConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;" & _
"Data Source=" & ThisWorkbook.Path & "\" & WN

conn.Open

If conn.State = adStateOpen Then

sSql = "DROP TABLE " & TableName

conn.Execute sSql

MsgBox "成功删除了" & WN & "中的" & TableName

conn.Close

End If

Set conn = Nothing

End Sub

同样，再给出一段删除上篇文章中生成的进销存表的Excel文件中的出仓工作表的代码：

Sub 删除出仓表()

'作者: bengdeng

'功能: 在程序文件同一目录下进销存表文件里的出仓表

'注意: 要在工具/引用中引用microsoft activex date objects x.x

' 其中x.x为版本号，可能会因为你安装的office的版本不同而不同，本例引用了2.5版

Dim conn As ADODB.Connection

Dim WN As String

Dim TableName As String

Dim sSql As String

WN = "进销存表.xls"

TableName = "出仓表"

Set conn = New ADODB.Connection

conn.ConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;" & _
"Extended Properties=Excel 8.0;" & _
"Data Source=" & ThisWorkbook.Path & "\" & WN

conn.Open

If conn.State = adStateOpen Then

sSql = "DROP TABLE " & TableName

```
conn.Execute sSql
MsgBox "成功删除了" & WN & "中的" & TableName
conn.Close
End If
Set conn = Nothing
End Sub
```

对于第二段代码，运行后的结果与第一段代码不太一样，我们用Excel打开文件后，会发现出仓表还是存在的，不过里面的数据已被删除，这时，如果再运行第二段代码的话，会发现有一——表'出仓表'不存在的提示！！看来SQL处理Excel文件时，把没有任何数据的空表会认为是不存在的。

最后，动一下手，运行一下这两段代码吧。

Alter Table - 修改数据表的语句

通过这几天在SQL语言教程 中介绍SQL的文章，相信对SQL感兴趣的朋友也就会慢慢增加，这两天一直困惑在上篇文章介绍的DROP TABLE 应用中，为什么不能删除Excel工作表，终于在网上的内容中看出一些头绪。

DROP 命令并不是物理上把字段删除，而只是简单地把它标记为 SQL 操作中不可见的。随后对该表的插入和更新将在该字段存储一个 NULL。因此，删除一个字段是很快，但是它不会立即缩减你的表在磁盘上的大小，因为被删除了的字段占据的空间还没有回收。这些空间将随着现有的行的更新而得到回收。

看来关于DROP相关的命令还有待继续学习，而今天的Alter Table中也包含DROP，在Excel中也存在与上面类似的现象。

Alter Table语言，是用来修改已创建数据表中表格的结构，包括增加列，修改已有列的数据类型和删除已有的列，其语法是：

增加列：

Alter Table 表格名 Add Column 列名 列名的数据类型

修改数据类型：

Alter Table 表格名 Alter Column 列名 列名的数据类型

删除列：

Alter Table 表格名 Drop Column 列名

SQL没有直接修改列名的SQL语言，可以用建立新列，再把数据复制到新列中，然后删除旧列的方法来完成列名的更改，还可以利用ADO的Field对象的Name属性来修改，具体大家可以参考一下ADO的帮助，这就不详细介绍了。

下面还是给出一段程序来让我们更了解上面说的内容：

Sub 修改进销存表数据库中明细表的结构()

'作者：bengdeng

'功能：在程序文件同一目录下进销存表数据库中修改明细表的结构

'注意：要在工具/引用中引用microsoft activex date objects x.x

' 其中x.x为版本号，可能会因为你安装的office的版本不同而不同，本例引用了2.5版

Dim conn As ADODB.Connection

Dim WN As String

Dim TableName As String

Dim ColumnName As String

Dim sSql As String

WN = "进销存表.mdb"

TableName = "明细表"

ColumnName = "示例栏"

Set conn = New ADODB.Connection

conn.ConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;" & _
"Data Source=" & ThisWorkbook.Path & "\" & WN

conn.Open

If conn.State = adStateOpen Then

MsgBox "首先在 " & TableName & " 中增加一列“文本类型”的 " & ColumnName & " " "

sSql = "Alter Table " & TableName & " Add Column " & ColumnName & " Char(50)"

conn.Execute sSql

MsgBox "成功在 " & TableName & " 中增加一列“文本类型”的 " & ColumnName & _

" " & vbCrLf & "您可以打开数据库示例文件查看一下效果" & vbCrLf & _

"查看完毕后关闭数据库并按确认继续程序！"

MsgBox "接着把刚才 " & TableName & " 中增加的 " & ColumnName & _

" " 更改为日期类型！"

sSql = "Alter Table " & TableName & " Alter Column " & ColumnName & " Date"

conn.Execute sSql

MsgBox "成功把刚才 " & TableName & " 中增加的 " & ColumnName & _

```

" " 更改为日期类型！" & vbCrLf & _
"您可以打开数据库示例文件查看一下效果" & vbCrLf & _
"查看完毕后关闭数据库并按确认继续程序！"

MsgBox "最后把刚才 " & TableName & " 中增加的 " & ColumnName & " 删除！"
sSql = "Alter Table " & TableName & " Drop Column " & ColumnName
conn.Execute sSql
MsgBox "成功把刚才 " & TableName & " 中增加的 " & ColumnName & _
" " 删除！" & vbCrLf & _
"您可以打开数据库示例文件查看一下效果"
conn.Close
End If
Set conn = Nothing
End Sub

```

同样，我们也给出一段操作**Excel**的代码，不同的是因为**Excel**本身没有列的数据类型的限制，所以修改数据类型的代码不能正确运行，因此就删除，感兴趣的朋友自己可以动手试下：

```

Sub 修改进销存表文件中明细表的结构()
*****
'作者：bengdeng
'功能：在程序文件同一目录下进销存表文件中修改明细表的结构
'注意：要在工具/引用中引用microsoft activex date objects x.x
' 其中x.x为版本号，可能会因为你安装的office的版本不同而不同，本例引用了2.5版
*****

Dim conn As ADODB.Connection
Dim WN As String
Dim TableName As String
Dim ColumnName As String
Dim sSql As String
WN = "进销存表.xls"
TableName = "明细表"
ColumnName = "示例栏"

Set conn = New ADODB.Connection
conn.ConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;" & _
"Extended Properties=Excel 8.0;" & _
"Data Source=" & ThisWorkbook.Path & "\" & WN
conn.Open
If conn.State = adStateOpen Then
MsgBox "首先在 " & TableName & " 中增加一列“文本类型”的 " & ColumnName & " "
sSql = "Alter Table " & TableName & " Add Column " & ColumnName & " Char(50)"
conn.Execute sSql
MsgBox "成功在 " & TableName & " 中增加一列“文本类型”的 " & ColumnName & _
" " & vbCrLf & _
"您可以打开数据库示例文件查看一下效果" & vbCrLf & _
"查看完毕后关闭数据库并按确认继续程序！"

MsgBox "最后把刚才 " & TableName & " 中增加的 " & ColumnName & " 删除！"
sSql = "Alter Table " & TableName & " Drop Column " & ColumnName
conn.Execute sSql
MsgBox "成功把刚才 " & TableName & " 中增加的 " & ColumnName & _
" " 删除！" & vbCrLf & _
"您可以打开数据库示例文件查看一下效果"

conn.Close
End If
Set conn = Nothing
End Sub

```

以之前的**Drop Table**一样，**SQL**并不能删除**Excel**中的数据，但程序确能正确运行！最后大家自己动一下复制上面的代码，在**VBA**里操作一下，就能更好的理解了。

INSERT INTO - 向数据库中添加数据

继续我们的SQL之路，下面的好多语句应该都不能单独说明，因为很多情况下都是组合在一起用的，不过为了一篇文章一个知识点，一个个的来了解，我们就从向数据库添加数据的INSERT INTO开始。

前面几篇 SQL语言教程 中介绍SQL的文章后，我们已可以创建，修改与删除一个数据库，今天介绍的INSERT INTO，我们就可以向上面创建的数据库中添加数据。INSERT INTO添加数据基本有两种方法：一种是一次性输入一笔数据，另一种是一次性输入多笔数据。因为后者还要用到SQL语句中最核心，最重要，最常用，也是最难全懂的Select语句，因此在这一篇文章中，我们先介绍第一种，语法如下：

INSERT INTO 表格名 (列名1, 列名2, ...) VALUES (数值1, 数值2, ...)

上面的语法就是向表格名中指定的表格里添加一行新的数据，在列名1，列名2……中添加对应的值，需要说明的一点是，不一定表格中的每一个列名都指定值，如果没有指定，该列的值就是设定数据库时默认的值。如果列的数值没有，而设定数据库时该列设定的是必填字段，而程序会提示出错。

按照惯例，我们还是给出一段程序来让我们更了解上面说的内容：

Sub 向销存表数据库录入数据()

'作者: bengdeng

'功能: 在程序文件同一目录下进销存表数据库中录入数据

'注意: 要在工具/引用中引用microsoft activex date objects x.x

' 其中x.x为版本号，可能会因为你安装的office的版本不同而不同，本例引用了2.5版

Dim conn As ADODB.Connection

Dim WN As String

Dim TableName As String

Dim sSql As String

WN = "进销存表.mdb"

TableName = "明细表"

Set conn = New ADODB.Connection

conn.ConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;" & _
"Data Source=" & ThisWorkbook.Path & "\" & WN

conn.Open

If conn.State = adStateOpen Then

sSql = "INSERT INTO " & TableName & " (物品名称, 结余日期, 结余数量, 进仓数量, 出仓数量)" & _
"VALUES ('铅笔', # & Date & '#,0,0,0')"

conn.Execute sSql

MsgBox "成功在 " & TableName & " 中增加了一条记录！"

conn.Close

End If

Set conn = Nothing

End Sub

上面的程序有两点特别说明一下：一个是字符串可以用单引号来特别确认，这个在字符串与SQL的一些保留关键词一样时，特别需要注意；另一个是日期数值的表示方式为——#日期#，否则不正确。同样，我们也给出一段操作Excel的代码：

Sub 向销存表文件录入数据()

'作者: bengdeng

'功能: 在程序文件同一目录下进销存表文件中录入数据

'注意: 要在工具/引用中引用microsoft activex date objects x.x

' 其中x.x为版本号，可能会因为你安装的office的版本不同而不同，本例引用了2.5版

Dim conn As ADODB.Connection

Dim WN As String

Dim TableName As String

Dim sSql As String

```
WN = "进销存表.xls"
TableName = "明细表"

Set conn = New ADODB.Connection
conn.ConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;" & _
    "Extended Properties=Excel 8.0;" & _
    "Data Source=" & ThisWorkbook.Path & "\" & WN
conn.Open
If conn.State = adStateOpen Then
sSql = "INSERT INTO [" & TableName & "$] (物品名称, 结余日期,结余数量,进仓数量,出仓数量)" _
    & "VALUES ('一笔',#" & Date & "#,0,0,0)"
    conn.Execute sSql
    MsgBox "成功在 " & TableName & " 中增加了一条记录！"
    conn.Close
End If
Set conn = Nothing
End Sub
```

最后大家自己动一下复制上面的代码，在VBA里操作一下，就能更好的理解了。希望通过这些文章，能让你开始对SQL的应用感兴趣，并应用到工作与生活中。

UPDATE - 修改数据库中已有的数据

前面一篇 **SQL语言教程** 介绍了 **INSERT INTO**，我们已可以向数据库中添加新的数据，这一篇我们接着介绍**UPDATE**，用来修改数据库中已有的数据。**UPDATE** 的语法为：

UPDATE 表格名 **SET** 列名1 = 数值1, 列名2 = 数值2,…… **WHERE** {条件}

UPDATE如果不指定条件，则会修改指定数据表中所有的数据，而指定条件后，则后修改数据表中所有满足条件的数据，下面，我们还是来写段程序来理解一个**UPDATE**的用法。

这段程序的功能是在进仓表中录入一条新记录——铅笔，10，然后更新明细表中对应铅笔项下的进仓数量，在原来的基础上增加10，程序如下：

Sub 更新销存表数据库中的数据()

'作者: bengdeng

'功能: 在程序文件同一目录下进销存表数据库中更新数据

'注意: 要在工具/引用中引用microsoft activex date objects x.x

'其中x.x为版本号，可能会因为你安装的office的版本不同而不同，本例引用了2.5版

Dim conn As ADODB.Connection

Dim WN As String

Dim TableName1 As String

Dim TableName2 As String

Dim sSql As String

Dim SI As Long

WN = "进销存表.mdb"

TableName1 = "明细表"

TableName2 = "进仓表"

SI = 10

Set conn = New ADODB.Connection

conn.ConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;" & _

"Data Source=" & ThisWorkbook.Path & "\" & WN

conn.Open

If conn.State = adStateOpen Then

sSql = "INSERT INTO " & TableName2 & _

" (进仓日期,物品名称, 进仓数量) VALUES (# & Date & #,'铅笔'," & SI & ")"

conn.Execute sSql

sSql = "UPDATE " & TableName1 & " SET 进仓数量 = 进仓数量 + " & SI & _

" WHERE 物品名称 = '铅笔'"

conn.Execute sSql

MsgBox "成功在 " & TableName2 & " 中增加了一条记录！"

conn.Close

End If

Set conn = Nothing

End Sub

之后，也给出一段操作**Excel**的代码，从这几篇文章的程序，大家对比一下应该可以发现，其实这两段代码主要的区别就在于**conn.ConnectionString**的不同和在引用**Excel**工作表是，应该在**Excel**工作表名称后多加一个\$。

Sub 更新进销存表文件中的数据()

'作者: bengdeng

'功能: 在程序文件同一目录下进销存表文件中更新数据

'注意: 要在工具/引用中引用microsoft activex date objects x.x

'其中x.x为版本号，可能会因为你安装的office的版本不同而不同，本例引用了2.5版

Dim conn As ADODB.Connection

```
Dim WN As String
Dim TableName1 As String
Dim TableName2 As String
Dim sSql As String
Dim SI As Long

WN = "进销存表.xls"
TableName1 = "明细表"
TableName2 = "进仓表"
SI = 10

Set conn = New ADODB.Connection
conn.ConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;" & _
    "Extended Properties=Excel 8.0;" & _
    "Data Source=" & ThisWorkbook.Path & "\" & WN
conn.Open
If conn.State = adStateOpen Then

    sSql = "INSERT INTO [" & TableName2 & "$] _
        (进仓日期,物品名称, 进仓数量) VALUES (#" & Date & "#,'铅笔'," & SI & ")"
    conn.Execute sSql

    sSql = "UPDATE [" & TableName1 & "$] SET 进仓数量 = 进仓数量 + " & SI & _
        " WHERE [物品名称] = '铅笔'"
    conn.Execute sSql

    MsgBox "成功在 “" & TableName2 & """ 中增加了一条记录！"
    conn.Close
End If
Set conn = Nothing
End Sub
```

最后大家还是动下手，复制上面的代码，运行一下看看效果吧。
一天进步一点，慢慢地，我们就会更强大*^_^*

DELETE FROM - 删除数据库的数据

通过前两篇 SQL语言教程 的介绍，我们已可以向数据库中添加数据和修改数据库中已有的数据。下面要介绍的 **DELETE FROM**，就能让我们删除数据库中的数据，其语法是：

DELETE FROM 表格名 WHERE {条件}

DELETE FROM 如果不指定条件，将清除指定表格中的所有数据，所以使用**DELETE FROM**时应该特别小心，因此，很多时候我们经常会在数据中增加一个列来标识数据是否向用户展示，这样对于数据就比较安全，不过这就是后话了，不在此继续讨论了。下面还是给出一个例子，功能是删除“明细表”中所有“物品名称”为“铅笔”的数据。

Sub 删除销存表数据库中的数据()

'作者: bengdeng

'功能: 在程序文件同一目录下进销存表数据库中删除数据

'注意: 要在工具/引用中引用microsoft activex date objects x.x

' 其中x.x为版本号，可能会因为你安装的office的版本不同而不同，本例引用了2.5版

Dim conn As ADODB.Connection

Dim WN As String

Dim TableName As String

Dim sSql As String

Dim TStr As String

WN = "进销存表.mdb"

TableName = "明细表"

TStr = "铅笔"

Set conn = New ADODB.Connection

conn.ConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;" & _
"Data Source=" & ThisWorkbook.Path & "\" & WN

conn.Open

If conn.State = adStateOpen Then

sSql = "DELETE FROM " & TableName & " WHERE 物品名称 = " & TStr & ""

conn.Execute sSql

MsgBox "成功删除 " & TableName & " 中 " & TStr & " 记录！"

conn.Close

End If

Set conn = Nothing

End Sub

因为Excel文件不支持用**DELETE FROM** 删除数据，因此这篇文章就没有相关的对Excel文件操作的程序了。如果要删除Excel文件中的行，应该在VBA中用**Delect**来完成，有兴趣的朋友也可以动手改一下上面的程序，来看一下错误的提示！

最后大家还是动手复制一下上面的程序，来看一下效果吧！

今天的内容，你学会了吗*^_^*

CopyFromRecordset方法的使用说明

介绍CopyFromRecordset方法的目的，是因为在以后的 SQL语言教程 中，我们大部份都需要它，来完成数据库向Excel传递内容的任务，因此特别在这里列出来说明一下。下面的大部份内容，可以在Excel的VBA帮助中找到。

CopyFromRecordset 方法的功能是将一个 ADO 或 DAO Recordset 对象的内容复制到工作表中，复制的起始位置在指定区域的左上角。其语法为：

expression.CopyFromRecordset(Data, MaxRows, MaxColumns)

expression : 必需。该表达式返回一个 Range 对象。

Data : Variant 类型，必需。复制到指定区域的 Recordset 对象。

MaxRows : Variant 类型，可选。复制到工作表的记录个数上限。如果省略该参数，将复制 Recordset 对象的所有记录。

MaxColumns : Variant 类型，可选。复制到工作表的字段个数上限。如果省略该参数，将复制 Recordset 对象的所有字段。

说明：复制从 Recordset 对象的当前行开始的内容。复制完成之后，Recordset 对象的 EOF 属性值为 True。

在以后的 SQL语言教程 中会有很多应用到此方法的例子，因此在这里就不再举例了，如果要看实例，那就在 SQL语言教程 中查看吧*^_^*

Select - 从数据库中检索数据

通过前面几篇 SQL 语言教程 的介绍，我们已可以创建，修改，删除数据库表与数据库里的数据，今天我们要接触的是 SQL 语言中，最根本，最常用，但也最难全懂的——Select 语句。在说明 Select 的语法时，我一直想它的语法还是像以前说明的那样，写成中文的形式，不过最后还是决定，先把原来的英文语法写在这里吧：

```
SELECT select_list
[ INTO new_table ]
FROM table_source
[ WHERE search_condition ]
[ GROUP BY group_by_expression ]
[ HAVING search_condition ]
[ ORDER BY order_expression [ ASC | DESC ] ]
```

如果要改为之前我的那种说明方式，可以理解为：

```
Select 列名1,列名2,……
[Into 新表格名]
From 表格名
[Where {条件}]
[Group By 组合列名1,组合列名2,……]
[Having {组合条件}]
[Order By 排序列名1,排序列名2,…… [Asc|Desc]]
```

其中[]是非必需的，而今天我们第一步先来给出一个最最简单的应用例子，这个例子不包含所有的[]的项目，其功能是在 Excel 表格中，列出数据库里明细表的所有内容。其中要用到的 CopyFromRecordset。

Sub 进销存表数据库中读取明细表的数据()

'作者: bengdeng

'功能: 在程序文件同一目录下进销存表数据库中读取明细表的数据

'注意: 要在工具/引用中引用microsoft activex date objects x.x

' 其中x.x为版本号，可能会因为你安装的office的版本不同而不同，本例引用了2.5版

```
Dim conn As ADODB.Connection
Dim WN As String
Dim TableName As String
Dim sSql As String
Dim xSh As Worksheet
Dim sRan As Range
Dim ColNum As Integer
Dim ii As Integer
```

WN = "进销存表.mdb"

TableName = "明细表"

'需要写入的工作表名称

Set xSh = ThisWorkbook.Worksheets("明细表")

'标题开始单元格

Set sRan = xSh.Range("A1")

'标题列数

ColNum = 5

Set conn = New ADODB.Connection

conn.ConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;" & _
"Data Source=" & ThisWorkbook.Path & "\" & WN

conn.Open

If conn.State = adStateOpen Then

sSql = "Select "

For ii = 1 To ColNum

sSql = sSql & sRan.Offset(0, ii - 1).Value & ", "

```

Next
sSql = Left(sSql, Len(sSql) - 1) & " From " & TableName

sRan.Offset(1, 0).CopyFromRecordset conn.Execute(sSql)

MsgBox "成功读取 “" & TableName & ”” 中的记录！"
conn.Close
End If
Set conn = Nothing
End Sub

```

同样，我们也给一段读取进销存表.xls中明细表里数据的程序，代码如下：

```

Sub 进销存表文件中读取明细表的数据()
*****
'作者: bengdeng
'功能: 在程序文件同一目录下进销存表数据文件中读取明细表的数据
'注意: 要在工具/引用中引用microsoft activex date objects x.x
'      其中x.x为版本号，可能会因为你安装的office的版本不同而不同，本例引用了2.5版
*****

Dim conn As ADODB.Connection
Dim WN As String
Dim TableName As String
Dim sSql As String
Dim xSh As Worksheet
Dim sRan As Range
Dim ColNum As Integer
Dim ii As Integer

WN = "进销存表.xls"
TableName = "明细表"
'需要写入的工作表名称
Set xSh = ThisWorkbook.Worksheets("明细表")
'标题开始单元格
Set sRan = xSh.Range("A1")
'标题列数
ColNum = 5

Set conn = New ADODB.Connection
conn.ConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;" & _
    "Extended Properties=Excel 8.0;" & _
    "Data Source=" & ThisWorkbook.Path & "\" & WN
conn.Open
If conn.State = adStateOpen Then
    sSql = "Select "
    For ii = 1 To ColNum
        sSql = sSql & sRan.Offset(0, ii - 1).Value & ","
    Next
    sSql = Left(sSql, Len(sSql) - 1) & " From [" & TableName & "$]"

    sRan.Offset(1, 0).CopyFromRecordset conn.Execute(sSql)

    MsgBox "成功读取 “" & TableName & ”” 中的记录！"
    conn.Close
End If
Set conn = Nothing
End Sub

```

通过两个程序，我们已可以初步了解**Select**，接下来的几篇文章，我们将一个个介绍[]中的每一个参数，而现在你要做的，就是复制上面的程序，动手学习一下吧*^_^*。SQL与Excel的世界已开始越来越精彩，一起继续前进吧！

Where - 筛选与限制检索的数据

我们已介绍是完整的**Select**的语法，并给出最简单的**Select**的应用例子，这一篇我们要来说明**Select**的第一个子句——**Where**。

Where的功能就是根据后面的条件，限制与筛选**Select**检索的数据，而条件中又可以分为几种：

- 1、运算符。通过比较运算符 —— >（大于），<（小于），=（等于），>=（大于等于），<=（小于等于），<>（不等于）来设定条件，例如：

```
Select 进仓日期,物品名称,进仓数量 From 进仓表 Where 物品名称='铅笔'
```

在进仓表检索出物品名称为铅笔的数据。

- 2、Or 与 And。通Or与And边接多个条件，例如：

```
Select 进仓日期,物品名称,进仓数量 From 进仓表 Where 物品名称='铅笔' And 日期=#2008-10-2#
```

在进仓表检索出2008年10月2日里物品名称为铅笔的数据。

- 3、Between...And。在指定的范围之内检索数据，例如：

```
Select 进仓日期,物品名称,进仓数量 From 进仓表 Where 进仓数量 Between 3 and 6
```

在进仓表检索出进仓数据在3与6之间的数据。

- 4、In。通过In 列出所有可能的值，例如：

```
Select 进仓日期,物品名称,进仓数量 From 进仓表 Where 物品名称 in ('铅笔','毛笔','钢笔')
```

在进仓表检索出物品名称为铅笔、毛笔和钢笔的数据。

- 5、Like。通过Like与通配符，列出符合的条件。通配符有%,_,[]三种，不同的数据库可能支持不一样，在Excel+ADO+ACCESS的环境下，三种都支持，例如：

```
Select 进仓日期,物品名称,进仓数量 From 进仓表 Where 物品名称 Like '铅%'
```

在进仓表检索出物品名称以铅为开头的的数据。

```
Select 进仓日期,物品名称,进仓数量 From 进仓表 Where 物品名称 Like '_笔%'
```

在进仓表检索出物品名称以任一个字符+笔为开头的的数据。实例中包含钢笔、铅笔等，不包含圆珠笔、水彩笔等。

```
Select 进仓日期,物品名称,进仓数量 From 进仓表 Where 物品名称 Like '[钢,圆珠,毛]%'
```

在进仓表检索出物品名称以钢，圆珠，毛为开头的的数据。

到此**Where**条件的说明就介绍完成，大家可以看出，特别是善用最后的**like**，可以更灵活地筛选出我们需要的数据。下面就以最后一个条件为内容，给出一个完整的程序，其它条件，大家只要代替**Where**后面的部分就可以了，代码如下：

```
Sub 进销存表数据库中按条件读取明细表的数据()
```

```
*****
```

```
'作者: bengdeng
```

```
'功能: 在程序文件同一目录下进销存表数据库中按条件读取明细表的数据
```

```
'注意: 要在工具/引用中引用microsoft activex date objects x.x
```

```
' 其中x.x为版本号，可能会因为你安装的office的版本不同而不同，本例引用了2.5版
```

```
*****
```

```
Dim conn As ADODB.Connection
```

```
Dim WN As String
```

```

Dim TableName As String
Dim sSql As String
Dim xSh As Worksheet
Dim sRan As Range
Dim ColNum As Integer
Dim ii As Integer

WN = "进销存表.mdb"
TableName = "进仓表"
'需要写入的工作表名称
Set xSh = ThisWorkbook.Worksheets("进仓表")
'标题开始单元格
Set sRan = xSh.Range("A1")
'标题列数
ColNum = 3

Set conn = New ADODB.Connection
conn.ConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;" & _
    "Data Source=" & ThisWorkbook.Path & "\" & WN
conn.Open
If conn.State = adStateOpen Then
    sSql = "Select "
    For ii = 1 To ColNum
        sSql = sSql & sRan.Offset(0, ii - 1).Value & ", "
    Next
    sSql = Left(sSql, Len(sSql) - 1) & " From " & TableName & _
        " Where 物品名称 Like '[钢,铅,毛]%"

    sRan.Offset(1, 0).CopyFromRecordset conn.Execute(sSql)

    MsgBox "成功读取 “" & TableName & ” 中的记录！ "
    conn.Close
End If
Set conn = Nothing
End Sub

```

同样，我们也是给如一段操作Excel的程序：

Sub 进销存表文件中按条件读取明细表的数据()

'作者：bengdeng

'功能：在程序文件同一目录下进销存表数据文件中按条件读取明细表的数据

'注意：要在工具/引用中引用microsoft activex date objects x.x

' 其中x.x为版本号，可能会因为你安装的office的版本不同而不同，本例引用了2.5版

```

Dim conn As ADODB.Connection
Dim WN As String
Dim TableName As String
Dim sSql As String
Dim xSh As Worksheet
Dim sRan As Range
Dim ColNum As Integer
Dim ii As Integer

```

WN = "进销存表.xls"

TableName = "进仓表"

'需要写入的工作表名称

Set xSh = ThisWorkbook.Worksheets("进仓表")

'标题开始单元格

Set sRan = xSh.Range("A1")

'标题列数

ColNum = 3

```
Set conn = New ADODB.Connection
conn.ConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;" & _
    "Extended Properties=Excel 8.0;" & _
    "Data Source=" & ThisWorkbook.Path & "\" & WN
conn.Open
If conn.State = adStateOpen Then
    sSql = "Select "
    For ii = 1 To ColNum
        sSql = sSql & sRan.Offset(0, ii - 1).Value & ","
    Next
    sSql = Left(sSql, Len(sSql) - 1) & " From [" & TableName & "$] _
        Where [物品名称] Like '[钢,铅,毛]%"

    sRan.Offset(1, 0).CopyFromRecordset conn.Execute(sSql)

    MsgBox "成功读取 “" & TableName & "” 中的记录！ "
    conn.Close
End If
Set conn = Nothing
End Sub
```

大家就可以复制上面的代码，动手运行与体会一下吧*^_^*。

SQL函数 - 预处理检索值的命令

作者: bengdeng | 来源: Excel吧 | 时间: 2008-10-15 | 阅读权限: 游客 | 会员币: 0 | 【大 中 小】上一篇文章 SQL语言教程 中我们介绍了Select的第一个子句Where, 而在了解第二个子句Group By之前, 我们还应该了解一下Sql提供的一些基本的SQL函数, 如果你了解Excel的函数, 你应该也可以从字面上了解下面5个函数的意义:

- 1、**Sum**。对检索的数值求和。
- 2、**Avg**。对检索的数据求平均值。
- 3、**Min**。求检索出数据的最小值。
- 4、**Max**。求检索出数据的最大值。
- 5、**Count**。求检索数据中非空的个数。

通过上面5个函数, 我们就可以查询数据集的进行运算而获得结果, 使用这些SQL函数的语法是:

SELECT 函数名(列名) **FROM** 表格名

下面我们就通过一段程序, 来求出进仓表中所有铅笔的进仓数量的总和, 代码如下:

Sub 利用SQL函数处理进销存表数据库中的数据()

'作者: bengdeng

'功能: 在程序文件同一目录下进销存表数据库中利用SQL函数处理的数据

'注意: 要在工具/引用中引用microsoft activex date objects x.x

' 其中x.x为版本号, 可能会因为你安装的office的版本不同而不同, 本例引用了2.5版

Dim conn As ADODB.Connection

Dim WN As String

Dim TableName As String

Dim sSql As String

Dim tStr As String

WN = "进销存表.mdb"

TableName = "进仓表"

tStr = "铅笔"

Set conn = New ADODB.Connection

conn.ConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;" & _

"Data Source=" & ThisWorkbook.Path & "\" & WN

conn.Open

If conn.State = adStateOpen Then

sSql = "Select Sum(进仓数量) As 进仓总计 From " & TableName & _
" Where 物品名称=" & tStr & ""

MsgBox " " & TableName & " 中 " & tStr & " 总计进仓数量为: " & _

conn.Execute(sSql)("进仓总计")

conn.Close

End If

Set conn = Nothing

End Sub

同样，我们也可以处理Excel文件中的数据：

Sub 利用SQL函数处理进销存表文件中的数据()

'作者: bengdeng

'功能: 在程序文件同一目录下进销存表文件中利用SQL函数处理的数据

'注意: 要在工具/引用中引用microsoft activex date objects x.x

' 其中x.x为版本号，可能会因为你安装的office的版本不同而不同，本例引用了2.5版

Dim conn As ADODB.Connection

Dim WN As String

Dim TableName As String

Dim sSql As String

Dim tStr As String

WN = "进销存表.xls"

TableName = "进仓表"

tStr = "铅笔"

Set conn = New ADODB.Connection

conn.ConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;" & _

"Extended Properties=Excel 8.0;" & _

"Data Source=" & ThisWorkbook.Path & "\" & WN

conn.Open

If conn.State = adStateOpen Then

sSql = "Select Sum(进仓数量) As 进仓总计 From [" & TableName & "\$] _
Where 物品名称=" & tStr & ""

MsgBox " " & TableName & " 中 " & tStr & " 总计进仓数量为: " & _
conn.Execute(sSql)("进仓总计")

conn.Close

End If

Set conn = Nothing

End Sub

大家就可以复制上面的代码，然后运行看一下效果，理解一下这篇文章的内容吧。介绍了SQL函数之后，下一节就可以了解Select的第二个子句——Group By。

Group By - 分类汇总检索的数据

上一篇 SQL语言教程 中我们了解SQL自带的函数，现在我们就来介绍一下Select的第二个子句——Group By。

Group By的功能，类似于Excel的分类汇总，它们指定列中相同的数据，按对应的函数合并在一起处理，其使用的语法是：

SELECT 条件列名1, 函数(汇总列名1) **FROM** 表格名 **GROUP BY** 条件列名1

要注意的是，条件列与汇总列都可以有一个或者多个，而汇总列名应该是可以让指定函数的计算类型，如果要在Select中列举出来，一定要在 **GROUP BY** 作为条件列出现，否则就会出错。但在**GROUP BY** 作为条件的列，不一定需要在Select中一一列举出来。

下面举个例子，这个例子，按物品名称汇总所有进仓表中各个物品的进仓数量，并列在一个新的工作表中。实例中还应用到ADO的Recordset对象，我们利用它来获得检索数据的列名，完整代码如下：

Sub 进销存表数据库中按条件分类汇总进仓表的数据()

'作者: bengdeng

'功能: 在程序文件同一目录下进销存表数据库中按条件分类汇总进仓表的数据

'注意: 要在工具/引用中引用microsoft activex date objects x.x

' 其中x.x为版本号，可能会因为你安装的office的版本不同而不同，本例引用了2.5版

Dim conn As ADODB.Connection

Dim rst As ADODB.Recordset

Dim WN As String

Dim TableName As String

Dim sSql As String

Dim xSh As Worksheet

Dim sRan As Range

Dim ii As Integer

WN = "进销存表.mdb"

TableName = "进仓表"

'需要写入的工作表名称

Set xSh = ThisWorkbook.Worksheets.Add

'标题开始单元格

Set sRan = xSh.Range("A1")

Set conn = New ADODB.Connection

conn.ConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;" & _
"Data Source=" & ThisWorkbook.Path & "\" & WN

conn.Open

If conn.State = adStateOpen Then

sSql = "Select 物品名称, Sum(进仓数量) As 进仓汇总 From " & TableName & _
" Group By 物品名称"

Set rst = New ADODB.Recordset

rst.Open sSql, conn, 1, 3

For ii = 0 To rst.Fields.Count - 1

sRan.Offset(0, ii).Value = rst.Fields(ii).Name

Next

sRan.Offset(1, 0).CopyFromRecordset rst

MsgBox "成功汇总 " & TableName & " 中的记录！"

conn.Close

End If

Set conn = Nothing

End Sub

同样，我们也给出一段操作Excel文件的代码，让大家对比一下，如果你从第一篇 SQL语言教程 看到现在的话，应该也能把上面的代码修改为下面的这段代码了吧*~_*

Sub 进销存表文件中按条件分类汇总进仓表的数据()

'作者: bengdeng

'功能: 在程序文件同一目录下进销存表文件中按条件分类汇总进仓表的数据

'注意: 要在工具/引用中引用microsoft activex date objects x.x

' 其中x.x为版本号，可能会因为你安装的office的版本不同而不同，本例引用了2.5版

Dim conn As ADODB.Connection

Dim rst As ADODB.Recordset

Dim WN As String

Dim TableName As String

Dim sSql As String

Dim xSh As Worksheet

Dim sRan As Range

Dim ii As Integer

WN = "进销存表.xls"

TableName = "进仓表"

'需要写入的工作表名称

Set xSh = ThisWorkbook.Worksheets.Add

'标题开始单元格

Set sRan = xSh.Range("A1")

Set conn = New ADODB.Connection

conn.ConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;" & _

"Extended Properties=Excel 8.0;" & _

"Data Source=" & ThisWorkbook.Path & "\" & WN

conn.Open

If conn.State = adStateOpen Then

sSql = "Select 物品名称, Sum(进仓数量) As 进仓汇总 _
From [" & TableName & "\$] Group By 物品名称"

Set rst = New ADODB.Recordset

rst.Open sSql, conn, 1, 3

For ii = 0 To rst.Fields.Count - 1

sRan.Offset(0, ii).Value = rst.Fields(ii).Name

Next

sRan.Offset(1, 0).CopyFromRecordset rst

MsgBox "成功汇总 “" & TableName & "” 中的记录！"

conn.Close

End If

Set conn = Nothing

End Sub

大家就可以复制上面的代码，然后运行看一下效果，理解一下这篇文章的内容吧。

Having - 筛选汇总后的数据

前面一篇 SQL语言教程 里介绍了Select的第二个子句Group By，这一篇要说明的是第三个子句Having。Having子句的功能是筛选Group By汇总后的数据，只有有Group By的情况下，才可以用Having，其作用有点类似于用Where来筛选Select的结果。其语法是：

SELECT 条件列名1, 函数(汇总列名1) **FROM** 表格名 **GROUP BY** 条件列名1 **HAVING** (函数条件)

要说明的是，Having后面出现的函数条件，应该在Selet中有列出来的，否则会出错。

为了更好的理解Having，我们还是举个例子，在上一篇介绍Group By中，我们举的例子是：按物品名称汇总所有进仓表中各个物品的进仓数量，并列在一个新的工作表中。而如果只要列出汇总数大于等于10的所有物品的汇总，这里就需要多加一个Having了。

Sub 进销存表数据库中按筛选分类汇总进仓表后的数据()

'作者: bengdeng

'功能: 在程序文件同一目录下进销存表数据库中筛选分类汇总进仓表后的数据

'注意: 要在工具/引用中引用microsoft activex date objects x.x

' 其中x.x为版本号，可能会因为你安装的office的版本不同而不同，本例引用了2.5版

Dim conn As ADODB.Connection

Dim rst As ADODB.Recordset

Dim WN As String

Dim TableName As String

Dim sSql As String

Dim xSh As Worksheet

Dim sRan As Range

Dim ii As Integer

WN = "进销存表.mdb"

TableName = "进仓表"

'需要写入的工作表名称

Set xSh = ThisWorkbook.Worksheets.Add

'标题开始单元格

Set sRan = xSh.Range("A1")

Set conn = New ADODB.Connection

conn.ConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;" & _
"Data Source=" & ThisWorkbook.Path & "\" & WN

conn.Open

If conn.State = adStateOpen Then

sSql = "Select 物品名称, Sum(进仓数量) As 进仓汇总 From " & TableName & _
" Group By 物品名称 Having Sum(进仓数量) >= 10"

Set rst = New ADODB.Recordset

rst.Open sSql, conn, 1, 3

For ii = 0 To rst.Fields.Count - 1

sRan.Offset(0, ii).Value = rst.Fields(ii).Name

Next

sRan.Offset(1, 0).CopyFromRecordset rst

MsgBox "成功汇总 " & TableName & " 中的记录！"

conn.Close

End If

Set conn = Nothing

End Sub

同样，我们还是给出一段操作Excel文件的例子：

Sub 进销存表文件中按筛选分类汇总进仓表后的数据()

'作者: bengdeng

'功能: 在程序文件同一目录下进销存表文件中筛选分类汇总进仓表后的数据

'注意: 要在工具/引用中引用microsoft activex date objects x.x

' 其中x.x为版本号，可能会因为你安装的office的版本不同而不同，本例引用了2.5版

Dim conn As ADODB.Connection

Dim rst As ADODB.Recordset

Dim WN As String

Dim TableName As String

Dim sSql As String

Dim xSh As Worksheet

Dim sRan As Range

Dim ii As Integer

WN = "进销存表.xls"

TableName = "进仓表"

'需要写入的工作表名称

Set xSh = ThisWorkbook.Worksheets.Add

'标题开始单元格

Set sRan = xSh.Range("A1")

Set conn = New ADODB.Connection

conn.ConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;" & _

"Extended Properties=Excel 8.0;" & _

"Data Source=" & ThisWorkbook.Path & "\" & WN

conn.Open

If conn.State = adStateOpen Then

sSql = "Select 物品名称, Sum(进仓数量) As 进仓汇总 From [" & TableName & "\$] " & _
" Group By 物品名称 Having Sum(进仓数量) >= 10"

Set rst = New ADODB.Recordset

rst.Open sSql, conn, 1, 3

For ii = 0 To rst.Fields.Count - 1

sRan.Offset(0, ii).Value = rst.Fields(ii).Name

Next

sRan.Offset(1, 0).CopyFromRecordset rst

MsgBox "成功汇总 “" & TableName & "” 中的记录！"

conn.Close

End If

Set conn = Nothing

End Sub

从上面的例子，我们应该就可以了解**Having**的用处了，也应该可以了解**Having**与**Where**的区别了吧。之后大家就可以复制上面的代码，然后运行看一下效果，理解一下这篇文章的内容吧。

Order By - 排序检索的数据

在前面介绍了Select的三个子句，这一篇要介绍的是第四个子句子句Order By。

如果说Where对应的Excel的功能是筛选，Group By对应的Excel功能是分类汇总的话，那么Order By对应的就是Excel的排序功能了。有时候我们检索的数据，在列出的时候需要按某些列的数据排列出来，这时就要用到Order By子句，其语法是：

SELECT 列名1,列名2…… **FROM** 表格名 **ORDER BY** 排序列名 [**ASC|DESC**]

语句最后的Acs与Desc分别代表升序与降序，如果没有指定的话，将默认为升序(Acs)，同样排序列名可以是多列，与检索的列名一样，用“,”分开。

下面，还是举个例子来帮助我们理解上面的内容。这段代码的功能是列出进仓表中按物品名称升序且按日期降序排列后的数据：

Sub 列出排序后进销存表数据库中进仓表的数据()

'作者: bengdeng

'功能: 列出排序后在程序文件同一目录下进销存表数据库中明细表的数据

'注意: 要在工具/引用中引用microsoft activex date objects x.x

' 其中x.x为版本号，可能会因为你安装的office的版本不同而不同，本例引用了2.5版

Dim conn As ADODB.Connection

Dim WN As String

Dim TableName As String

Dim sSql As String

Dim xSh As Worksheet

Dim sRan As Range

Dim ColNum As Integer

Dim ii As Integer

WN = "进销存表.mdb"

TableName = "进仓表"

'需要写入的工作表名称

Set xSh = ThisWorkbook.Worksheets("进仓表")

'标题开始单元格

Set sRan = xSh.Range("A1")

'标题列数

ColNum = 3

Set conn = New ADODB.Connection

conn.ConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;" & _

"Data Source=" & ThisWorkbook.Path & "\" & WN

conn.Open

If conn.State = adStateOpen Then

sSql = "Select "

For ii = 1 To ColNum

sSql = sSql & sRan.Offset(0, ii - 1).Value & ", "

Next

sSql = Left(sSql, Len(sSql) - 1) & " From " & TableName & _

" Order By 物品名称 Asc,进仓日期 Desc"

sRan.Offset(1, 0).CopyFromRecordset conn.Execute(sSql)

MsgBox "成功读取 "" & TableName & "" 中的记录! "

conn.Close

End If

Set conn = Nothing

End Sub

按照惯例，我们也给出一段处理Excel文件的代码，其实如果从第一篇教程看起的话，大家应该都可以自己修改成下面的代码了。

```
Sub 列出排序后进销存表文件中进仓表的数据()
*****
'作者: bengdeng
'功能: 列出排序后在程序文件同一目录下进销存表文件中明细表的数据
'注意: 要在工具/引用中引用microsoft activex date objects x.x
'      其中x.x为版本号，可能会因为你安装的office的版本不同而不同，本例引用了2.5版
*****

Dim conn As ADODB.Connection
Dim WN As String
Dim TableName As String
Dim sSql As String
Dim xSh As Worksheet
Dim sRan As Range
Dim ColNum As Integer
Dim ii As Integer

WN = "进销存表.xls"
TableName = "进仓表"
'需要写入的工作表名称
Set xSh = ThisWorkbook.Worksheets("进仓表")
'标题开始单元格
Set sRan = xSh.Range("A1")
'标题列数
ColNum = 3

Set conn = New ADODB.Connection
conn.ConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;" & _
    "Extended Properties=Excel 8.0;" & _
    "Data Source=" & ThisWorkbook.Path & "\" & WN
conn.Open
If conn.State = adStateOpen Then
    sSql = "Select "
    For ii = 1 To ColNum
        sSql = sSql & sRan.Offset(0, ii - 1).Value & ","
    Next
    sSql = Left(sSql, Len(sSql) - 1) & " From [" & TableName & "$] Order By 物品名称 Asc,进仓日期 Desc"

    sRan.Offset(1, 0).CopyFromRecordset conn.Execute(sSql)

    MsgBox "成功读取 “" & TableName & ” 中的记录！"
    conn.Close
End If
Set conn = Nothing
End Sub
```

到此Order By就介绍完了，现在动一下手，复制上面的代码，然后运行看一下效果，学习一下吧！

*(星号) - 数据表中所有的列

作者: bengdeng | 来源: Excel吧 | 时间: 2008-10-19 | 阅读权限: 游客 | 会员币: 0 | 【大 中 小】
 通过前面几篇 SQL语言教程 的文章, 我们已了解了四个Select的子句, 在了解下一个子句Into之前, 我们先来了解一下*(星号)在SQL的作用。

*(星号), 在程序一般代表所有的东西, 在SQL里也不例外, 它的作用就是代表数据表中所有的列。在我们检索某些特定的数据时, 我们不需要知道列的信息, *(星号)就能起到它的作用了。*(星号)主要用到两个方面:

1、列出指定数据表中的所有列的数据。其语法是:

```
SELECT * FROM 表格名
```

2、在Count中, 获得数据的行数。其语法是:

```
SELECT COUNT(*) FROM 表格名
```

第二种应用的情况是很常用到的, 而第一种应用, 不应该在任何时候都使用Select *, 因为当数据的列变动后, Select *的结果也会变动, 可能有不可预见的情况出现, 所以一般情况下, 还是列出列名清单。

下面就举一个例子, 来让我们理解一下Select *的用途。这段程序是在一个新的工作表中列出明细表里的所有数据, 像这样的情况, 用Select *是最理想的了。

Sub 列出进销存表数据库中明细表里的数据()

'作者: bengdeng

'功能: 列出在程序文件同一目录下进销存表数据库中明细表里的数据

'注意: 要在工具/引用中引用microsoft activex date objects x.x

' 其中x.x为版本号, 可能会因为你安装的office的版本不同而不同, 本例引用了2.5版

```
Dim conn As ADODB.Connection
```

```
Dim rst As ADODB.Recordset
```

```
Dim WN As String
```

```
Dim TableName As String
```

```
Dim sSql As String
```

```
Dim xSh As Worksheet
```

```
Dim sRan As Range
```

```
Dim ii As Integer
```

```
WN = "进销存表.mdb"
```

```
TableName = "出仓表"
```

'需要写入的工作表名称

```
Set xSh = ThisWorkbook.Worksheets.Add
```

'标题开始单元格

```
Set sRan = xSh.Range("A1")
```

```
Set conn = New ADODB.Connection
```

```
conn.ConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;" & _  
"Data Source=" & ThisWorkbook.Path & "\" & WN
```

```
conn.Open
```

```
If conn.State = adStateOpen Then
```

```
    sSql = "Select * From " & TableName
```

```
    Set rst = New ADODB.Recordset
```

```
    rst.Open sSql, conn, 1, 3
```

```
    For ii = 0 To rst.Fields.Count - 1
```

```
        sRan.Offset(0, ii).Value = rst.Fields(ii).Name
```

```
        If rst.Fields(ii).Name = "出仓日期" Then
```

```
            sRan.Offset(0, ii).EntireColumn.NumberFormatLocal = "yyyy-m-d"
```

```
        End If
```

```
    Next
```

```

sRan.Offset(1, 0).CopyFromRecordset rst

MsgBox "成功列出 “" & TableName & "” 中的所有记录！"
conn.Close
End If
Set conn = Nothing
End Sub

```

对于上面的代码，大家可以自己动手改成对Excel文件操作的例子，下面对Excel文件操作的例子，我们来举一个应用Count(*)的实例。这段程序的功能是获得明细表中所有数据的行数，代码如下：

```

Sub 获得进销存表文件中明细表的数据行数()
*****
'时间: 2008-10-19
'作者: bengdeng
'功能: 获得在程序文件同一目录下进销存表文件中明细表的数据行数
'注意: 要在工具/引用中引用microsoft activex date objects x.x
'      其中x.x为版本号，可能会因为你安装的office的版本不同而不同，本例引用了2.5版

*****

Dim conn As ADODB.Connection
Dim rst As ADODB.Recordset
Dim WN As String
Dim TableName As String
Dim sSql As String

WN = "进销存表.xls"
TableName = "明细表"

Set conn = New ADODB.Connection
conn.ConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;" & _
    "Extended Properties=Excel 8.0;" & _
    "Data Source=" & ThisWorkbook.Path & "\" & WN
conn.Open
If conn.State = adStateOpen Then
    sSql = "Select Count(*) From [" & TableName & "]"
    MsgBox " “" & TableName & "” 中总计有 “" & conn.Execute(sSql)(0) & "” 条的记录！"
    conn.Close
End If
Set conn = Nothing
End Sub

```

*在SQL中的应用就介绍完了，在之后介绍Into子句时，我们还会用到它，现在你可以动一下手，复制上面的代码，然后运行看一下效果，学习一下吧！

Select Into - 把检索的数据添加到新的数据表中

通过前面几篇 SQL语言教程，我们已了解Select 的三个子句与一些相关的内容，这篇要介绍的是第四个
子句Into。

标题中，为什么不是只写Into而写的是Select Into，是要与下面一篇中Insert Into 第二种输入方法区别开来，我们曾经说过Insert Into 还有一种是一次性输入多笔数据的方法，没有介绍，在我们了解了本篇的内容之后，下一篇会来了解与对比。

通过之前对Insert Into 的认识，我们可以看出Into的功能是向数据表中增加记录，其语法是：

SELECT 列名1,列名2…… **INTO** 新表格名 **FROM** 表格名

要注意的一点是，新表格名在数据库中应该是不存在的，否则就会出错。

下面举个例子，来看看一下Select Into 的应用。这个例子是吧出仓表中所有的铅笔的记录，添加到一个新的表名为“临时表”的数据表中。

Sub 进销存表数据库中出仓表里检索的数据生成新表()

'时间: 2008-10-20

'作者: bengdeng

'功能: 在程序文件同一目录下进销存表数据库中出仓表里检索的数据生成新表

'注意: 要在工具/引用中引用microsoft activex date objects x.x

' 其中x.x为版本号，可能会因为你安装的office的版本不同而不同，本例引用了2.5版

Dim conn As ADODB.Connection

Dim WN As String

Dim OldTableName As String

Dim NewTableName As String

Dim sSql As String

Dim tStr As String

WN = "进销存表.mdb"

OldTableName = "出仓表"

NewTableName = "临时表"

tStr = "铅笔"

Set conn = New ADODB.Connection

conn.ConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;" & _
"Data Source=" & ThisWorkbook.Path & "\" & WN

conn.Open

If conn.State = adStateOpen Then

sSql = "Select * Into " & NewTableName & _
" From " & OldTableName & " Where 物品名称=" & tStr & ""

conn.Execute sSql

MsgBox "成功把 “" & OldTableName & "” 中的所有的 “" & tStr & _
"” 记录汇总到 “" & NewTableName & "” 中！"

conn.Close

End If

Set conn = Nothing

End Sub

运行上面的结果，我们可以看到，生成了一个结构与出仓表一样的临时表，同时把出仓表数据中所有物品名称为铅笔的数据添加在其中。因此，Select Into也常用来创建一个数据结构与已存在的数据表一样的数据表，只要把Where的条件设定为一个不可能存在的情况就可以了。

同样，我们也能出一段操作Excel文件的例子，代码如下：

Sub 进销存表文件中出仓表里检索的数据生成新表()

'时间: 2008-10-20

'作者: bengdeng

'功能: 在程序文件同一目录下进销存表文件中出仓表里检索的数据生成新表

'注意: 要在工具/引用中引用microsoft activex date objects x.x

' 其中x.x为版本号，可能会因为你安装的office的版本不同而不同，本例引用了2.5版

Dim conn As ADODB.Connection

Dim WN As String

Dim OldTableName As String

Dim NewTableName As String

Dim sSql As String

Dim tStr As String

WN = "进销存表.xls"

OldTableName = "出仓表"

NewTableName = "临时表"

tStr = "铅笔"

Set conn = New ADODB.Connection

conn.ConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;" & _

"Extended Properties=Excel 8.0;" & _

"Data Source=" & ThisWorkbook.Path & "\" & WN

conn.Open

If conn.State = adStateOpen Then

sSql = "Select * Into [" & NewTableName & "] From [" & OldTableName & "\$] _

Where 物品名称=" & tStr & ""

conn.Execute sSql

MsgBox "成功把 “ & OldTableName & ” 中的所有的 “ & tStr & _

” 记录汇总到 “ & NewTableName & ” 中！ ”

conn.Close

End If

Set conn = Nothing

End Sub

要注意的是新表中的表名不需要加\$，而旧表中应该加上\$，到这一篇文章至，Select的语法与其子句就介绍完了，大家可以复制一下上面的代码，自己体会一下吧。

Insert Into Select - 批量添加数据到数据库中

通过前面几篇 SQL语言教程 中，我们已说到这一篇要介绍的是Insert Into的另一种用法，也就是把检索出来的数据，批量添加到数据库中，其语法是：

INSERT INTO 表格名1 (列名11,列名12, ...) SELECT 列名21, 列名22, ... FROM 表格名2

以上的语法是最基本的语法。在Select后，还可能含有 WHERE、GROUP BY、及 HAVING 等子句，以及表格连接及别名等等。与Select Into对比，Select Into中目标表名应该在数据库中是不存在的，而Insert Into Select正好相反，其目标表名在数据库中一定要存在的。

下面的这个例子，是把出仓表中的所有的钢笔数据，添加到上一篇文章生成的临时表的数据库中，如果你没有看到上一篇文章，请运行上一篇文章的程序生成临时表后，再运行下面这段程序。

Sub 进销存表数据库中出仓表里检索的数据添加到已有的表()

'作者: bengdeng

'功能: 在程序文件同一目录下进销存表数据库中出仓表里检索的数据生成新表

'注意: 要在工具/引用中引用microsoft activex date objects x.x

' 其中x.x为版本号，可能会因为你安装的office的版本不同而不同，本例引用了2.5版

Dim conn As ADODB.Connection

Dim WN As String

Dim OldTableName As String

Dim NewTableName As String

Dim sSql As String

Dim tStr As String

WN = "进销存表.mdb"

OldTableName = "出仓表"

NewTableName = "临时表"

tStr = "钢笔"

Set conn = New ADODB.Connection

conn.ConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;" & _
"Data Source=" & ThisWorkbook.Path & "\" & WN

conn.Open

If conn.State = adStateOpen Then

sSql = "Insert Into " & NewTableName & " Select * From " & _
OldTableName & " Where 物品名称=" & tStr & ""

conn.Execute sSql

MsgBox "成功把 “" & OldTableName & "” 中的所有的 “" & tStr & _
"” 记录汇总到 “" & NewTableName & "” 中！"

conn.Close

End If

Set conn = Nothing

End Sub

因为上一篇文章生成的临时表与出仓表的结构是完全相同的，因此正如上面的例子，我们不需要列出NewTableName中列明细资料，而OldTableName的列的资料，我们也可以用*来表示！

那么，如果列名不一样，能不能导入呢？当列的数据类型一致时，还是可以导入的，下面的这段操作Excel文件的程序，功能是把进仓表的毛笔数据，导入到临时表中，其中进仓表的进仓日期对应着临时表的出仓日期，进仓数量对应着出仓数量，代码如下：

Sub 进销存表文件中进仓表里检索的数据添加到已有的表()

'作者: bengdeng

'功能: 在程序文件同一目录下进销存表文件中进仓表里检索的数据生成新表

'注意: 要在工具/引用中引用microsoft activex date objects x.x

' 其中x.x为版本号，可能会因为你安装的office的版本不同而不同，本例引用了2.5版

Dim conn As ADODB.Connection

Dim WN As String

Dim OldTableName As String

Dim NewTableName As String

Dim sSql As String

Dim tStr As String

WN = "进销存表.xls"

OldTableName = "进仓表"

NewTableName = "临时表"

tStr = "毛笔"

Set conn = New ADODB.Connection

conn.ConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;" & _

"Extended Properties=Excel 8.0;" & _

"Data Source=" & ThisWorkbook.Path & "\" & WN

conn.Open

If conn.State = adStateOpen Then

sSql = "Insert Into [" & NewTableName & "\$] (出仓日期,物品名称,出仓数量) " & _

"Select 进仓日期 As 出仓日期,物品名称,进仓数量 As 出仓数量 From [" & _

OldTableName & "\$] Where 物品名称=" & tStr & ""

conn.Execute sSql

MsgBox "成功把 " & OldTableName & " 中的所有的 " & _

tStr & " 记录汇总到 " & NewTableName & " 中! "

conn.Close

End If

Set conn = Nothing

End Sub

Insert Into Select就介绍完了，现在可以动手复制一下上面的代码，运行一下看一下效果吧*^_^*。

Top - 限制检索结果的数量

继续在 SQL语言教程 中介绍余下的几个SQL语句关键词。

在前面的 SQL语言教程 中我们已了解了Select，前面所讲的Select中，我们获得的检索数据是符合条件的所有的数据，而有时候我们并不需要所有的，而只是需要列出最前面的几条数据时，这时我们就需要用到Top。

Top的作用就是限制检索结果的数量，其使用方法是：

Select Top 数量 列名1,列名2,…… From 表格名 [Order By 排序列名1,排序列名2,…… [Asc|Desc]]

需要注意的是，Top限制的数量在一般情况下就是检索出来的数量，但如果正好其排序(Order By)的最后一条数据中，出现并列的情况时，检索的数结果就会大于数量，因此，如果我们需要准确的结果时，排序的条件就要是唯一的，以防止上面的情况发生。

下面同样给出一个例子，这个例子是检索出10月3日中，出仓数量最大的三条数据。

Sub 进销存表数据库中出仓表指定日期最大三条出仓记录()

'作者: bengdeng

' 功能: 在程序文件同一目录下进销存表数据库中出仓表里指定日期的最大三条出仓记录

'注意: 要在工 具/引用中引用microsoft activex date objects x.x

' 其中x.x为 版本号，可能会因为你安装的office的版本不同而不同，本例引用了2.5版

Dim conn As ADODB.Connection

Dim WN As String

Dim TableName As String

Dim sSql As String

Dim xSh As Worksheet

Dim sRan As Range

Dim tNum As Integer

Dim tDate As Date

WN = "进销存表.mdb"

TableName = "出仓表"

Set xSh = ThisWorkbook.Worksheets(TableName)

Set sRan = xSh.Range("A1")

tNum = 3

tDate = "2008-10-3"

Set conn = New ADODB.Connection

conn.ConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;" & _

 & "Data Source=" & ThisWorkbook.Path & "\" & WN

conn.Open

If conn.State = adStateOpen Then

 sSql = "Select Top " & tNum & " * From " & TableName & _

 " Where 出仓日期=#" & tDate & "# Order By 出仓数量 Desc"

 sRan.Offset(1, 0).CopyFromRecordset conn.Execute(sSql)

 MsgBox "成功导入 "" & TableName & "" 中 "" & tDate & "" 的最大" & tNum & "条出仓记录! "

 conn.Close

End If

Set conn = Nothing

End Sub

而下面这个操作Excel的例子，功能和上面的差不多，不过数量改成前4条。

Sub 进销存表文件中出仓表指定日期最大三条出仓记录()

'作者: bengdeng

'功能: 在程序文件同一目录下进销存表文件中出仓表里指定日期的最大三条出仓记录

'注意: 要在工具 /引用中引用microsoft activex date objects x.x

' 其中x.x为版本号，可能会因为你安装的office的版本不同而不同，本例引用了2.5版

Dim conn As ADODB.Connection

Dim WN As String

Dim TableName As String

Dim sSql As String

Dim xSh As Worksheet

Dim sRan As Range

Dim tNum As Integer

Dim tDate As Date

WN = "进销存表.xls"

TableName = "出仓表"

Set xSh = ThisWorkbook.Worksheets(TableName)

Set sRan = xSh.Range("A1")

tNum = 4

tDate = "2008-10-3"

Set conn = New ADODB.Connection

conn.ConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;" & _

Extended Properties=Excel 8.0;" & _

"Data Source=" & ThisWorkbook.Path & "\ " & WN

conn.Open

If conn.State = adStateOpen Then

sSql = "Select Top " & tNum & " * From [" & TableName & "\$] _
Where 出仓日期=#" & tDate & "# Order By 出仓数量 Desc"

sRan.Offset(1, 0).CopyFromRecordset conn.Execute(sSql)

MsgBox "成功导入 “" & TableName & ” 中 “" & tDate & ” 的最大" & tNum & "条出仓记录！"

conn.Close

End If

Set conn = Nothing

End Sub

第二个例子的结果是5行！！可以看到，因为排序的数量，并不是唯一的，导致有可能出现并列的结果！为了更好的理解上面的文字，现在可以动手复制上面的代码，运行一下看一下效果吧*^_^*。

Distinct - 筛选出不重复的数据

在前一篇 **SQL语言教程** 中，我们介绍了用**Top**来限制结果的数量，而这一篇要介绍的**Distinct**，其功能是用来使结果是不重复的数据。

在**Excel**中我们可以利用高级筛选来筛选出不重复的值，在最新版的**Excel2007**中，还新增了删除重复项的功能，这些都类似于**Distinct**的作用，其语法是：

Select Distinct 列名 **From** 表格名

一般来说，因为**Distinct** 会按列名对整个数据表进行额外的排序，如果此列不是索引列的话，在大量数据时运行的效率不是很好，因此除非万不得已，一般都不用**Distinct** 而用以后在介绍的**Exists**来代替**Distinct** 的一些用法，不过对于数据量较小时，运算的速度还是可以接受的。

下面这个例子，是列出单条记录中，有出仓数量大于8的物品名称清单。

Sub 进销存表数据库中出仓表指定条件的清单()

'作者: bengdeng

'功能: 在程序文件同一目录下进销存表数据库中出仓表指定条件的清单

'注意: 要在工具/引用中引用microsoft activex date objects x.x

' 其中x.x为版本号，可能会因为你安装的office的版本不同而不同，本例引用了2.5版

Dim conn As ADODB.Connection

Dim WN As String

Dim TableName As String

Dim sSql As String

Dim xSh As Worksheet

Dim sRan As Range

Dim tNum As Integer

WN = "进销存表.mdb"

TableName = "出仓表"

Set xSh = ThisWorkbook.Worksheets.Add

Set sRan = xSh.Range("A1")

tNum = 8

Set conn = New ADODB.Connection

conn.ConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;" & _

"Data Source=" & ThisWorkbook.Path & "\ " & WN

conn.Open

If conn.State = adStateOpen Then

sSql = "Select Distinct 物品名称 From " & TableName & " Where 出仓数量 > " & tNum

sRan.Value = "下面为 " & TableName & " 中出仓数量大于" & tNum & "的物品清单! "

sRan.Offset(1, 0).CopyFromRecordset conn.Execute(sSql)

MsgBox "成功导入 " & TableName & " 中出仓数量大于" & tNum & "的物品清单! "

conn.Close

End If

Set conn = Nothing

End Sub

照例，我们也给出一段相同功能的操作Excel文件的程序：

Sub 进销存表文件中出仓表指定条件的清单()

'作者: bengdeng

'功能: 在程序文件同一目录下进销存表文件中出仓表指定条件的清单

'注意: 要在工具/引用中引用microsoft activex date objects x.x

' 其中x.x为版本号，可能会因为你安装的office的版本不同而不同，本例引用了2.5版

Dim conn As ADODB.Connection

Dim WN As String

Dim TableName As String

Dim sSql As String

Dim xSh As Worksheet

Dim sRan As Range

Dim tNum As Integer

WN = "进销存表.xls"

TableName = "出仓表"

Set xSh = ThisWorkbook.Worksheets.Add

Set sRan = xSh.Range("A1")

tNum = 8

Set conn = New ADODB.Connection

conn.ConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;" & _

"Extended Properties=Excel 8.0;" & _

"Data Source=" & ThisWorkbook.Path & "\" & WN

conn.Open

If conn.State = adStateOpen Then

sSql = "Select Distinct 物品名称 From [" & TableName & "\$] Where 出仓数量 > " & tNum

sRan.Value = "下面为 “" & TableName & "” 中出仓数量大于" & tNum & "的物品清单! "

sRan.Offset(1, 0).CopyFromRecordset conn.Execute(sSql)

MsgBox "成功导入 “" & TableName & "” 中出仓数量大于" & tNum & "的物品清单! "

conn.Close

End If

Set conn = Nothing

End Sub

为了更好的理解上面的文字，你可以动手复制上面的代码，运行一下看一下效果吧*^_^*。