Tool zur schnellen Datenabfrage SMARTQRY

Karel Štípek

Bei der Entwicklung einer Datenbankapplikation muss man oft viele einfache Datenabfragen durchführen. Für die meisten dieser Aufgaben lohnt es nicht, sie explizit auszuprogrammieren, d.h. sie als benannte Abfragen zu speichern, bzw. sie von der Benutzeroberfläche abrufbar zu machen.

Auch wenn die Applikation fertig ist, ist es oft nicht anders. Der Anwender will plötzlich eine andere Auswertung, die im Programmkonzept nicht berücksichtigt wurde – und diese möglichst sofort und ohne einen besonderen Aufwand. Meistens geht es um ein paar Summenzahlen, nicht um schön formatierte Berichte. Auch wenn die Arbeit mit dem Abfragengenerator durchaus einfach und anschaulich ist, ist eine minimale Eischulung doch notwendig und ohne die schaffen das viele Menschen nicht.

Es ist auch noch ein anderes Problem nicht so ganz unwesentlich. Viele PC-Anwender, die sich selbst Access-Applikatonen erstellen, bilden nach einiger Zeit auf ihrer Festplatte, bzw. auf den gemeinsamen Servern, eine komplexe Verzeichnisstruktur, wo die wiederholte Suche nach der richtigen Datenbank einige Zeit in Anspruch nehmen kann.

Gespeicherte Datenselektion

Das vorgelegte Programm kann eine Vereinfachung in schnelle Datenabfragen bringen. Das Grundprinzip ist, dass eine Datenselektion benannt, gespeichert und nachträglich wieder einfach aufgerufen werden kann. Es stehen dann folgende Parameter sofort zur Verfügung:

- Pfad und Dateiname der Datenbank
- Name der analysierten Tabelle
- Feldnamen, bzw. Ausdrücke für die Gruppierung und Summierung
- Filterausdruck
- Titel für die Berichtsausgabe der Selektion

Bedienung des Programms

Komfortable Benutzeroberfläche

Alle Bedienungselemente sind in einem Formular untergebracht. Die Struktur und Aufschriften sind übersichtlich und sprechend, außerdem können Sie sich mit der Schaltfläche mit Fragezeichen eine Kurzanleitung ansehen.

Das Formular mit dem Beispiel einer Selektion ist im folgenden Bild dargestellt. Als Testdatenquelle wird in diesem Artikel die Tabelle Artikel aus der Beispieldatenbank Nordwind.mdb, die standardmäßig mit Access installiert werden kann.

Selektion auswählen

Wenn das Programm bereits mindestens eine gespeicherte Selektion enthält, können Sie sie mit der Combobox ganz oben auswählen. Unmittelbar nach der Auswahl wird die Existenz der angegebenen Datenbank und Tabelle überprüft und falls sie nicht mehr vorhanden sind (wenn sich die Datenstruktur seit der Speicherung der Selektion geändert hat), wird eine Fehlermeldung angezeigt. Eine Selektion kann mit der Schaltfläche rechts von der Combobox nach Abfrage gelöscht werden.

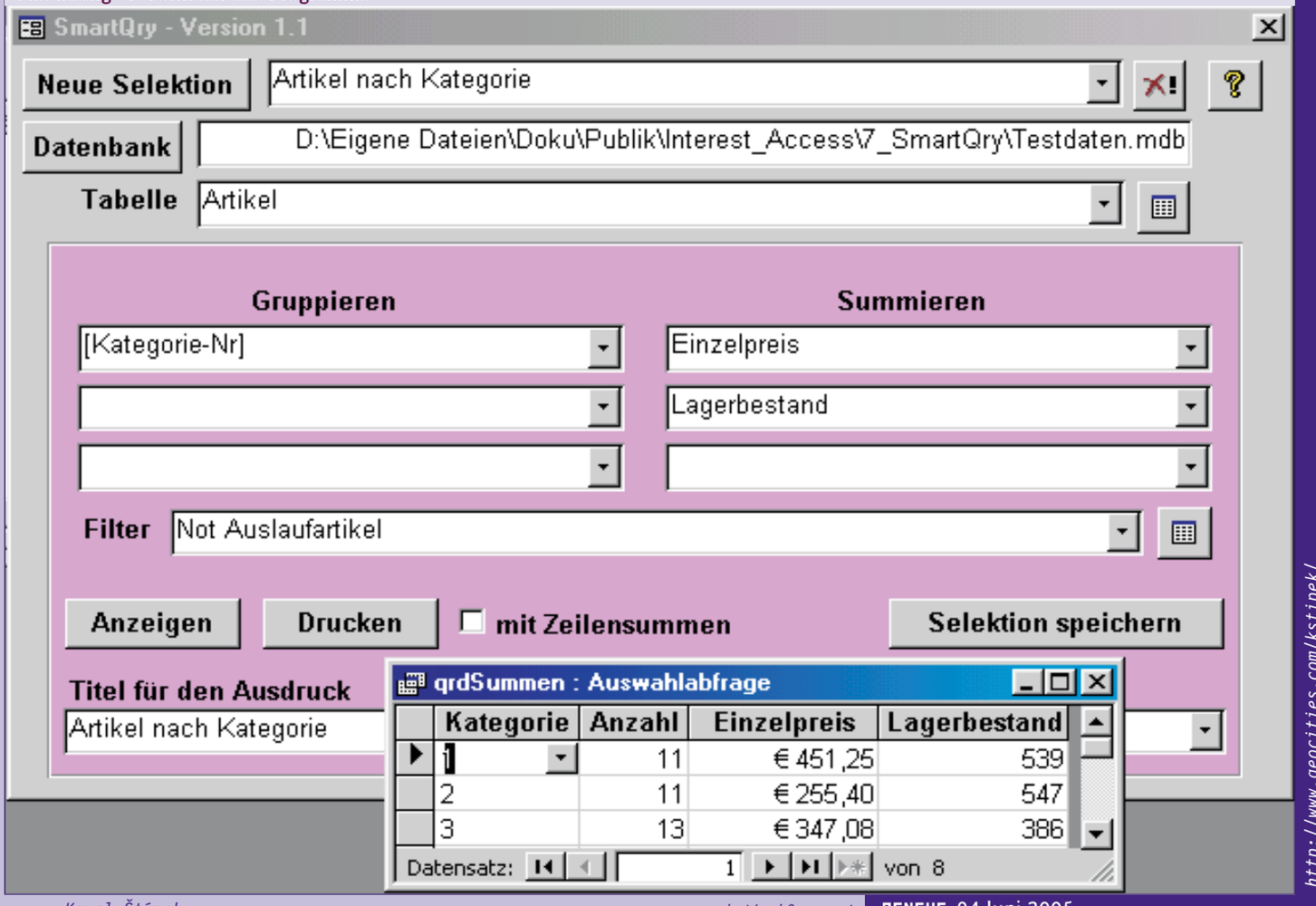
Eine neue Selektion erstellen

Die Schaltfläche `Neue Selektion` links oben löscht alle Felder im Formular und bereitet sie für die neue Eingabe. Allerdings ist es meistens effizienter, eine ähnliche Selektion auszuwählen und sie nach der Modifikation einiger Parameter unter einem neuen Namen zu speichern.

Auswahl einer Datenbank und Tabelle

Die Schaltfläche Datenbank ermöglicht die Auswahl einer Datenbank mit Hilfe des Standarddialogs. Der Pfad und Name der ausgewählten Datenbank wird im daneben liegenden Textfeld angezeigt. Hier können Sie auch direkt Änderungen vornehmen, ohne den Dialog aufzurufen, falls es Ihnen einfacher vor-

Abbildung -1: Formular des Programms



kommt (wenn z.B. die neue Datenbank im gleichen Verzeichnis liegt und ihr Name sich wenig unterscheidet).

Nach der Auswahl einer Datenbank wird die Combobox `Tabelle` automatisch mit allen vorhandenen Tabellennamen befüllt. Nachdem Sie eine Tabelle ausgewählt haben, können Sie sie mit der Schaltfläche rechts in der Datensicht öffnen.

Gruppierung

Sie können bis zu drei Gruppierungsfelder aus allen Feldnamen der jeweiligen Tabelle auswählen. Das Programm setzt voraus, dass Sie die Auswahl von oben nach unten durchführen.

Sie müssen sich dabei nicht nur auf die Tabellenfeldnamen einschränken. In die Combobox kann auch ein Ausdruck eingegeben werden, wie z.B. `year(Bestelldatum)` für die Gruppierung nach dem Jahr der Bestellung. Der Wermuthstropfen dabei ist, dass Sie in dem Fall nicht den Ausdruck im Spaltentitel der Abfrage (im Bericht schon) angezeigt bekommen, weil hier ein Defaultname wie z.B. `Expr1000` von Access eingesetzt wird.

Die Feldnamen, die Leerzeichen oder Bindestriche enthalten, müssen in eckige Klammern eingeschlossen werden. Man könnte zwar die Klammern vom Programm automatisch einfügen lassen, in dem Fall wäre es aber nicht möglich, die oben erwähnten Ausdrücke einzugeben.

Summierung

Genauso wie die Felder für die Gruppierung, können Sie auch bis zu drei Feldern auswählen, für welche die Gruppensummen gebildet werden. Sie dürfen natürlich in den Comboboxen für die Summierung nur die Felder mit numerischen Datentypen auswählen, sonst meldet die Abfrage eine Syntaxfehler.

Ohne die Angabe der Summierung wird nur die Anzahl der Datensätze in den Gruppen ausgewertet.

Tabelle gesamt

Wenn Sie überhaupt keine Gruppierungsfelder oder –ausdrücke angegeben haben, werden die Gesamtanzahl der Datensätze, bzw. die Gesamtsummen der summierten Felder angezeigt. Die Abfrage liefert eine einzige Zeile, die in diesem Fall in Form eines Berichtes nicht dargestellt werden kann.

Daten filtern

In die Combobox Filter können Sie einen beliebigen Filterausdruck in der SQL-WHERE-Syntax (identisch mit den Formeln, die Sie in der Kriterienzeile des Abfragengenerators verwenden können) eingeben. Jede abgeschlossene Eingabe in der Combobox wird unabhängig von der Selektion automatisch gespeichert. Damit sparen Sie sich Tipparbeit bei der Definition des gleichen oder ähnlichen Kriteriums. Mit der Schaltfläche rechts neben dieser Combobox können Sie sich die gefilterten Daten im Detail (ohne die Gruppierung) anschauen.

In der beigelegten Datenbank sind einige gültige Filterausdrücke gespeichert, damit Sie sich ein Bild über die Syntax machen können, falls Sie mit SQL-Sprache nicht vertraut sind. Die gespeicherten Ausdrücke können unverändert natürlich nur dann eingesetzt werden, wenn die analysierte Tabelle die verwendeten Feldnamen auch enthält.

Selektion anzeigen

Wenn Sie auf die Schaltfläche `Anzeigen` klicken, wird aufgrund der angegebenen Selektionsparameter eine dynamische Abfrage erstellt und in der Datensicht geöffnet.

Selektion drucken

Die angezeigte Datenselektion liefert die Gruppensummen mit der Berücksichtigung aller Gruppierungsfelder. Damit ist aber die Selektion noch nicht aussagekräftig genug. Man braucht auch die Zwischensummen der höheren Ebenen und die Gesamtsummen über die ganze Tabelle. Zu dem Zweck kann mit der Schaltfläche `Drucken` ein dynamisch modifizierter Bericht in der Seitenansicht angezeigt oder ausgedruckt werden.

Berichtstitel

Sie können den Ausdruck der Selektion auch mit einem aussagekräftigen Titel versehen, das am Anfang des Berichts gedruckt wird. Den Text können Sie in der Combobox `Titel` für den Ausdruck eingeben. Jede Eingabe wird wie bei den Filterausdrücken unabhängig von der Selektion automatisch gespeichert, damit Sie sie später auswählen und eventuell anpassen können.

Filterausdruck im Bericht

Unter dem Berichtstitel wird auch der Filterausdruck ausgegeben. Wenn für die jeweilige Selektion kein Filter definiert ist, wird an der Stelle der Text `Alle Datensätze` angezeigt.

Summenzeilen im Bericht

Im Bericht werden auch die Summen für höhere Gruppierungsebenen und die Gesamtsummen berechnet. Die einzelnen Ebenen unterscheiden sich in der Schriftart (normal – fett – fett unterstrichen). **Abbildung 2** zeigt den Bericht, wobei alle drei Gruppierungsmöglichkeiten ausgenutzt sind.

Wenn mehrere Felder summiert werden, kann im Bericht auch eine Zeilensumme berechnet werden, falls die sinnvoll ist. Diese Möglichkeit ist im Formular über die Checkbox `Mit Zeilensummen` auswählbar.

Programminterne Datenstrukturen

Die analysierte Tabelle

Jede Tabelle, die mit dem Programm analysiert wird, wird nach der Auswahl verknüpft und die Verknüpfung auf den Namen "T" umbenannt. So präsentiert sich intern jede Tabelle für die dynamisch generierte Abfrage und Bericht gleich. Die Verknüpfung wird bei jedem Öffnen des Formulars gelöscht, damit es nicht zu Fehlermeldungen kommt, wenn die vorigesmal verknüpfte Datenbank oder Tabelle eventuell nicht mehr vorhanden sind.

Speicherung der Selektion

Die Selektion wird nicht als eine übliche Access-Abfrage, sondern als ein Datensatz mit den Inhalten der Eingabefelder im Formular gespeichert. Aus diesen Werten wird erst bei der Datenanzeige der notwendige SQL-Ausdruck dynamisch zusammengestellt und in einer temporären Abfrage gespeichert.

Bei der Berichtsausgabe wird direkt auf die Tabelle zugegriffen. Die Gruppierung und Summierung wird im Bericht selbst nach der Modifikation durch die Inhalte der Formularfelder durchgeführt.

Tabelle tblSelection

Ein Satz der Tabelle `tblSelection` enthält eine gespeicherte Datenselektion. Die Struktur besteht aus dem primären Schlüssel, dem Namen der Selektion und einer Menge von Feldern, welche die Inhalte der Steuerelemente des Formulars enthalten. (Das Formular ist aber nicht an die Tabelle gebunden, wie später erklärt wird.)

Tabellen tblWhere und tblRepTitle

Beim Erstellen einer Selektion kann der Filterausdruck und der Berichtstitel aus allen bisher eingegebenen Werten ausgewählt werden. Die Texte werden in den Tabellen `tblWhere`, bzw. `tblRepTitle` gespeichert und mit der Selektion über die fremden Schlüssel in der Tabelle `tblSelection` verknüpft.

Die Strukturen aller Tabellen und ihre Beziehungen sind am besten im Beziehungsfenster ersichtlich.

Abbildung -2: Beispiel eines Berichts

Artikel nach Kategorie

[Kategorie-Nr]=4 and [Lieferanten-Nr]>14

Kategorie-Nr	Lieferanten-Nr	Liefereinheit	Anzahl	Lagerbestand
4	15	10 x 500-g-Packungen	1	26,00
4	15	10-kg-Paket	1	26,00
4	15	500-g-Packung	1	112,00
4	**15**		**3**	**164,00**
4	28	15 x 300-g-Stücke	1	19,00
4	28	5-kg-Packung	1	79,00
4	**28**		**2**	**98,00**
4			**5**	**262,00**
GESAMTSUMME			**5**	**262,00**

Eingabe (Auswahl) der Selektionsparameter

Formular ist ungebunden

Auch wenn ein Datensatz der Tabelle `tblSelection` praktisch ein Abbild des Formulars ist, darf das Formular nicht gebunden sein. Der Grund dafür ist der, dass wir meistens

- eine gespeicherte Selektion auswählen,
- sie eventuell ändern
- die geänderte unter einem anderen Namen speichern.

Wären die Formularfelder an die Tabellenfelder gebunden, würde mit jeder Änderung im Formular sofort auch der ausgewählte Datensatz geändert.

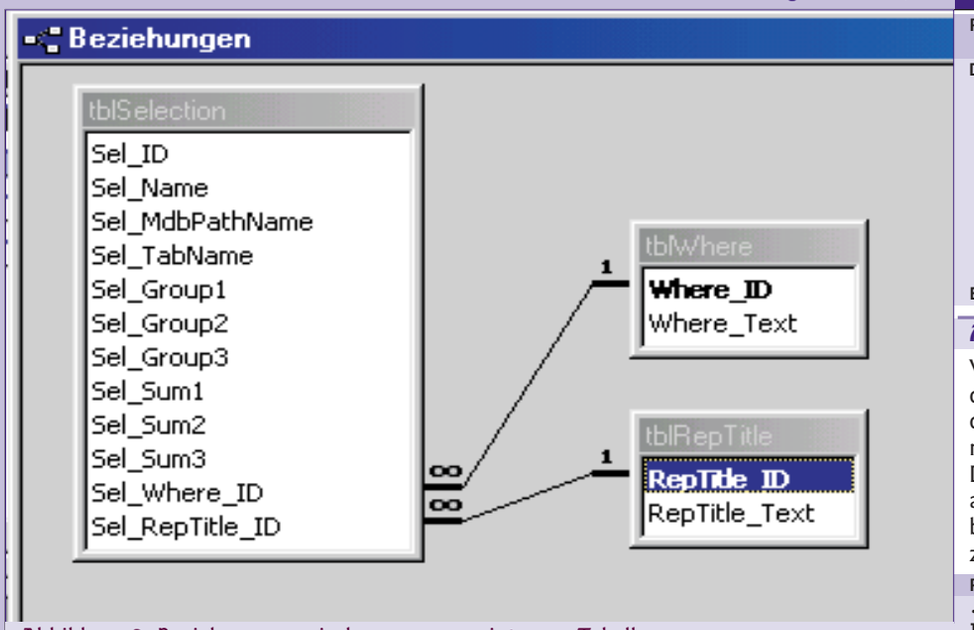


Abbildung-3: Beziehungen zwischen programminternen Tabellen

Tabellenfeldnamen speichern

Damit die Übertragung der Dateninhalt aus der Tabelle ins Formular (bei der Auswahl der Selektion) und zurück (beim Speichern der Selektion) nicht mühsam mit einzelnen Zuweisungen gelöst werden muss, sind die Tabellenfeldnamen in der Eigenschaft `Marke` jedes Steuerelements gespeichert. Diese Eigenschaft (englisch *Tag*) ist im Access seit der Version 97 verfügbar und kann einen Zeichenfolgenausdruck mit bis zu 2048 Zeichen enthalten.

Selektion auswählen

Der notwendige Programmcode wird in der Ereignisprozedur `Nach Aktualisierung` der Combobox `cboQry` aufgerufen. Es handelt sich eigentlich um eine Standardlösung der Synchronisation eines Formulars mit der Auswahl in einer Combobox. Mit der Methode `FindFirst` wird der Datensatz aus der Tabelle `tblSelection` gesucht, der den in der Combobox ausgewählte Schlüssel `Sel_ID` enthält. Der Unterschied besteht darin, dass die Formularfelder nicht automatisch durch die Datenbindung, sondern explizit mit den im Tag angegebenen Tabellenfeldern befüllt werden.

Am Ende der Prozedur werden die Prozeduren `txtMDB_AfterUpdate` und `cboTab_AfterUpdate` aufgerufen. Dadurch werden die gleichen Aktionen durchgeführt wie wenn die Datenbank und Tabelle manuell ausgewählt würde. Zuletzt wird die Verknüpfung mit der Funktion `CheckLink` (aus den Microsoft-Beispiellösungen) überprüft.

```
Private Sub cboQry_AfterUpdate()
...
Set rec =
CurrentDb.OpenRecordset("tblSelection"...
rec.FindFirst "Sel_ID = " & Me.cboQry
For Each ctl In Me.Controls
  fldname = Nz(ctl.Tag, "")
  If Len(fldname) > 0 Then
    ctl.Value = rec.Fields(fldname)
  End If
Next ctl
rec.Close
txtMDB_AfterUpdate
cboTab_AfterUpdate
CheckLink
End Sub
```

Datenbank öffnen

Der Aufruf des Standarddialogs wurde komplett und unverändert als Modul `modOpenFileDialog` aus einer Beispiel-Datenbank übernommen und wird deswegen nicht näher behandelt. Nach der Auswahl wird der Pfad und Dateiname in das Textfeld `txtMDB` übertragen. Da auch direkte Änderungen in diesem Feld möglich sind, steht der Code für die Auswertung der Datenbankwahl erst in der Ereignisprozedur dieses Textfeldes.

- Die alte Tabellenverknüpfung wird gelöscht.
- Alle Tabellennamen der Datenbank werden in die Datensatzherkunft der Combobox `cboTab` (Tabellenauswahl) übernommen

```
Private Sub txtMDB_AfterUpdate()
TableDelete "T"
If Len(Nz(Me.txtMDB, "")) > 0 Then
  cboTab.RowSource = "SELECT name
  FROM [" & Me.txtMDB &
  "].msysobjects
  WHERE (type=1) and not name like 'MSys*'
  ORDER BY name;"
End If
End Sub
```

Tabelle auswählen

Nach der Auswahl einer Tabelle in der Combobox `cboTab` wird die Tabelle mit der Prozedur `TabConnect` verknüpft und die Comboboxe für die Definition der Gruppierung und Summierung aktualisiert, damit sie die aktuellen Tabellenfeldnamen zum Auswählen anbieten.

```
Private Sub cboTab_AfterUpdate()
Dim i%
  TabConnect
  For i = 1 To 3
    Me.Controls("cboGroup" & i).Requery
    Me.Controls("cboSum" & i).Requery
  Next i
End Sub
```

Filterausdruck und Berichtstitel

Die Comboboxe für die Eingabe oder Auswahl der Filterausdrücke und der Berichtstitel speichern automatisch alle eingegeben Texte in den Tabellen `tblWhere`, bzw. `tblRepTitle`. Der Code, der diese Funktionalität realisiert ist in der Ereignisprozedur Bei Nicht in Liste enthalten – der Code für die Combobox `cboWhere` folgt.

```
Private Sub cboWhere_NotInList
     (NewData As String, Response As Integer)
Dim rec As Recordset
  If Len(Trim(NewData)) > 0 Then
    Set rec = CurrentDb.OpenRecordset
      ("SELECT * FROM tblWhere")
    rec.AddNew
    rec.Fields("Where_Text") = NewData
    rec.Update
    rec.Close
    Response = DATA_ERRADDED
  End If
End Sub
```

Anzeige der gefilterten Daten

Wenn Sie auf die Schaltfläche rechts neben der Combobox für die Eingabe der Filterbedingung klicken, wird die Abfrage `qrdDetail` modifiziert und in der Datensicht geöffnet. Diese Abfrage berücksichtigt nur den Filterausdruck, es werden die Datensätze der Tabelle ohne Gruppierung (im Detail) angezeigt.

```
Private Sub cmdDetail_Click()
...
If CheckLink() Then
  s = " SELECT * FROM T "
  If Not IsNull(Me.cboWhere.Column(1)) Then
    s = s & " WHERE (" &
    Me.cboWhere.Column(1) & ")"
  End If
  Set qdf = CurrentDb.QueryDefs("qrdDetail")
  qdf.SQL = s
  qdf.Close
  DoCmd.OpenQuery ("qrdDetail")
  ...
```

Anzeige, Ausdruck und Speichern einer Selektion

Selektion anzeigen

Nachdem Sie auf die Schaltfläche Anzeigen klicken, wird zuerst die Gültigkeit der Verknüpfung überprüft und dann mit der Funktion `Def_qrdSummen()` die Abfrage `qrdSummen` modifiziert und in der Datensicht geöffnet.

```
Private Sub cboSelView_Click()
...
If CheckLink() Then
  If Def_qrdSummen() Then
    DoCmd.OpenQuery ("qrdSummen")
...
```

Funktion Def_qrdSummen()

Die Funktion `Def_qrdSummen()` legt aus den Werten der Formularfelder den SQL-Ausdruck für die Abfrage `qrdSummen` zusammen und weist ihn der Abfrage zu.

Zuerst werden die Angaben in den Comboboxen für die Gruppierung zu einem gemeinsamen Ausdruck in der Variablen `sgroup` gespeichert. Es wird vorausgesetzt, dass die Comboboxe von oben nach unten verwendet werden.

```
...
If Len(Me.cboGroup1) > 0 Then
    sgroup = Me.cboGroup1
    If Len(Me.cboGroup2) > 0 Then
      sgroup = sgroup & "," & Me.cboGroup2
        If Len(Me.cboGroup3) > 0 Then
          sgroup = sgroup & "," &
          Me.cboGroup3
        End If
    End If
End If
```

Der Text in der Variablen `sgroup` bildet den Anfang des SQL-Ausdrucks der gesamten Abfrage, der in der Variablen `s` gespeichert wird. Gleich dahinter wird der Ausdruck für die Anzahl der Datensätze in den Gruppen eingefügt.

```
s = "SELECT "
If Len(sgroup) > 0 Then
```

```
  s = s & sgroup & ","
End If
s = s & " count(*) AS Anzahl "
```

Der Ausdruck für die Summierung wird in der Variablen `ssum` gebildet. Der Code ist ähnlich wie bei der Gruppierung, deswegen ist nur der Anfang aufgelistet. Beachten Sie, dass die Summenfelder den Aliasnamen gleich dem Feldnamen bekommen. Auf die vom Abfragengenerator standardmäßig generierte Formulierung "Summe von " wird mit Rücksicht auf die beschränkte Seitenbreite im Bericht verzichtet.

```
If Len(Me.cboSum1) > 0 Then
  ssum = ",SUM(T.[" & Me.cboSum1 & "]) AS
  [" & Me.cboSum1 & "]"
  If Len(Me.cboSum2) > 0 ...
```

Zuletzt wird dem Selektionsausdruck noch der Name der Tabellenverknüpfung in der Klausel `FROM` und der Filterausdruck angefügt und der Inhalt der Variablen `sgroup` in der Klausel `GROUP BY` noch einmal wiederholt.

```
s = s & " FROM T "
If Not IsNull(Me.cboWhere.Column(1)) Then
  s = s & _
     " WHERE (" & Me.cboWhere.Column(1) & ")"
End If
If Len(sgroup) > 0 Then
  s = s & " GROUP BY " & sgroup
End If
```

Damit ist der SQL-Ausdruck fertig und wird der Abfrage `qrdSummen` zugewiesen.

Selektion ausdrucken

Bei der Ausgabe der Ergebnisse der Selektion im Bericht wird keine dynamische Abfrage erstellt, sondern direkt auf die Tabelle unter dem Verknüpfungsnamen "T" zugegriffen. Die notwendige Gruppierung und Summierung wird im Bericht selbst implementiert. Diese Lösung hat den Vorteil, dass in den Fußbereichen der Gruppen auch die Zwischensummen automatisch berechnet werden.

Es ist nur dann sinnvoll, die Selektion zu drucken, wenn mindestens ein Gruppierungsfeld (oder Ausdruck) definiert ist. Beim Klick auf die Schaltfläche `Drucken` wird also nach der Überprüfung der Verknüpfung der Bericht nur dann geöffnet, wenn der Inhalt der Combobox `cboGroup1` nicht leer ist.

```
Private Sub cmdSelPrint_Click()
...
If CheckLink() Then
  If IsNull(Me.cboGroup1) Then
    MsgBox "Für den Bericht muss ...
  Else
    DoCmd.OpenReport "rptSummen", acPreview
...
```

Berichtsentwurf

Im Bericht `rptSummen` werden in den Textfeldern für Berichtstitel, Filterausdruck und Spaltenaufschriften die Werte direkt aus dem Formular angezeigt. Deswegen darf der Bericht ohne das Formular `frmMain` nicht geöffnet werden. Das wird am Anfang der Ereignisprozedur `Beim Öffnen` überprüft.

Es werden drei Gruppierungsebenen definiert, allerdings ohne den Ausdruck. In den Fußbereichen des Berichts werden die notwendigen Textfelder erstellt, sie bleiben aber alle ungebunden.

Prozedur PropertyByTag

Beim Öffnen des Berichts werden die Eigenschaften der Steuerelemente je nach ihrer Position im Bericht und je nach den Werten der Selektionsparameter dynamisch geändert. Damit nicht jedes einzelne Element namentlich angesprochen werden muss, werden alle, die logisch zusammengehören, mit einem bestimmten Eintrag in der Eigenschaft `Marke` (Tag) gekennzeichnet. Die Prozedur `PropertyByTag` kann dann eine bestimmte Eigenschaft in allen gleich gekennzeichneten Steuerelementen ändern.

Sie wird mit vier Parametern aufgerufen:

`frm`	Verweis auf das Objekt (Formular oder Bericht)
`ptag`	gesuchter Text in der Eigenschaft `Marke`
`prop`	Name der geänderten Eigenschaft
`propval`	der neue Wert der Eigenschaft

```
Public Sub PropertyByTag (frm As Object, _
   ptag$, prop$, propval As Variant)
...
For Each ctl In frm.Controls
  If InStr(ctl.Tag, ptag) > 0 Then
    ctl.Properties(prop) = propval
  End If
Next ctl
End Sub
```

Bericht öffnen

In der Ereignisprozedur `Beim Öffnen` wird zuerst überprüft (mit der Funktion `IsLoaded()` aus der Access-Beispieldatenbank), ob das Formular offen ist. Wenn das nicht der Fall ist, wird eine Meldung ausgegeben und das Öffnen des Berichts unterbrochen. Sonst wird die Referenz auf das Formular in der lokalen Variablen `frm` gespeichert. Der Filterausdruck wird in die Berichtseigenschaft Filter übernommen.

```
Private Sub Report_Open(Cancel As Integer)
If Not IsLoaded("frmMain") Then
  MsgBox "Dieser Bericht kann nur über das
          Hauptformular geöffnet werden"
  Cancel = True
Else
  Set frm = Forms("frmMain")
  Me.Filter =
  Nz(frm.Controls("cboWhere").Column(1), "")
```

Gruppierung im Bericht

Die bis zu drei Gruppierungen (numeriert mit 0, 1 und 2) im Bericht müssen um die entsprechenden Ausdrücke (Inhalte der Comboboxen `cboGroup...` im Formular) ergänzt werden. Die Gruppendefinition darf in keinem Fall leer bleiben, das würde der Bericht als einen Syntaxfehler auswerten. Auch wenn die niedrigeren Comboboxen im Formular leer bleiben und die entsprechenden Fußbereiche ausgeblendet werden, müssen sie einen gültigen Ausdruck (hier wird der gleiche wie auf der darüberliegenden Ebene verwendet) enthalten.

Nach der Definition des Gruppenausdrucks wird auch die Datenherkunft der Textfelder der gleichen Ebene auf das im Formular ausgewählte Feld (oder eingegebene Ausdruck) mit der Prozedur `PropertyByTag` gesetzt.

```
Me.GroupLevel(0).ControlSource =
  "=" & frm.Controls("cboGroup1")
Me.GroupLevel(1).ControlSource = ...
Me.GroupLevel(2).ControlSource = ...
PropertyByTag Me, "group1", _
  "ControlSource", "=" &
  frm.Controls("cboGroup1")
If Len(frm.cboGroup2) > 0 Then
    ...
```

Summenfelder im Bericht

Bei der Definition der Summenfelder ist es nur notwendig, die Datenherkunft der in allen Fußbereichen vorbereiteten ungebundenen Feldern (mit der Prozedur `PropertyByTag`) zu definieren.

```
If Len(frm.cboSum1) > 0 Then
  PropertyByTag Me, "sumfield1",
    "ControlSource",
    "=SUM([" & frm.Controls("cboSum1") & "])"
  If Len(frm.cboSum2) > 0
```

Die Felder für Zeilensummen enthalten beim Berichtsentwurf erstellte Ausdrücke, die sich auf die Berichtsfelder beziehen. Sie werden nur dann eingeblendet, wenn mehr als ein Summenfeld existiert und die Checkbox für die Zeilensumme im Formular aktiviert wurde.

```
If (Not frm.Controls("chkRowSum"))
Or IsNull(frm.cboSum1) Then
  PropertyByTag Me, "rowsum", "Visible", False
End If
```

Fußbereiche ausblenden

Wenn in der Selektion nur ein oder zwei Gruppierungsausdrücke definiert sind, werden die ungenutzten Fußbereiche des Berichts ausgeblendet. Das wird über den Parameter `Cancel` der Ereignisprozedur `Beim Formatieren` in den Bereichen `Gruppenfuß 2` und `Gruppenfuß3` realisiert.

```
Private Sub Gruppenfuß2_Format _
  (Cancel As Integer, FormatCount As Integer)
  Cancel = IsNull(frm.Controls("cboGroup2"))
End Sub
```

Selektion speichern

Wenn die Selektion getestet wurde und setzen voraus, dass sie in der gleichen oder modifizierten Form auch in der Zukunft verwendet werden kann, können Sie alle eingegebenen Parameter unter einem Namen speichern.

Nach der Eingabe des Namens in einer InputBox wird der umgekehrte Vorgang durchgeführt, als bei der Auswahl der Selektion – die Inhalte der ungebundenen Formularfelder werden in einem neuen Datensatz der Tabelle `tblSelection` gespeichert.

Am Ende der Prozedur wird die Combobox `cboQry` aktualisiert, damit die bereits gespeicherte Selektion sofort aufrufbar ist.

```
Private Sub cmdSelSave_Click()
...
If CheckLink() Then
  qryname = InputBox("Geben Sie den Namen ...
...
  Set rec = CurrentDb.OpenRecordset(
      "tblSelection", dbOpenDynaset)
  With rec
    .AddNew
    !sel_name = qryname
    On Error Resume Next
    For Each ctl In Me.Controls
      fldname = Nz(ctl.Tag, "")
      If Len(fldname) > 0 Then
        .Fields(fldname) = ctl.Value
      End If
    Next ctl
    .Update
    .Close
  End With
  Me.cboQry.Requery
...
```

Fazit

Das vorgestellte Programm hat sich trotz der einfachen Konzeption sowohl unter den Entwicklern als auch unter den Endusern gut bewährt und spart bei den sich wiederholenden Datenabfragen eine Menge Zeit.