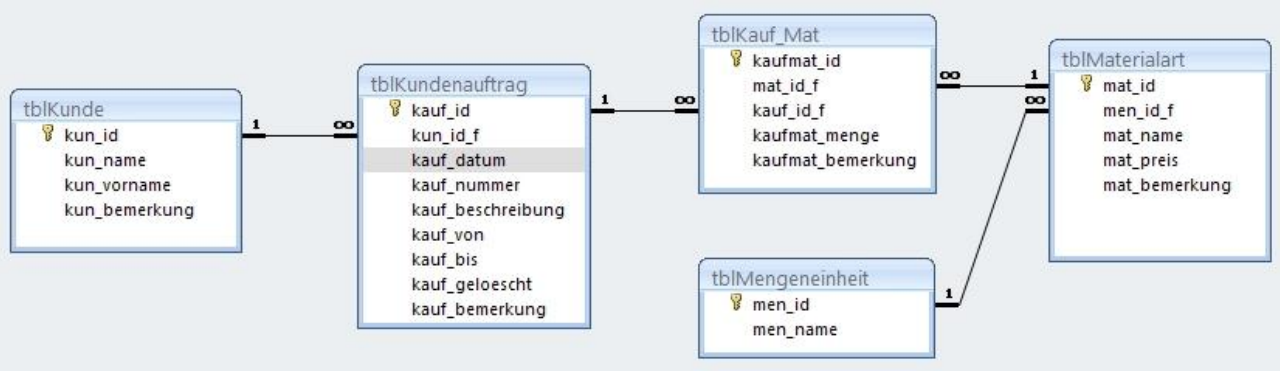
Tutorial „Formularentwicklung“

Ich wundere bzw. ärgere mich immer wieder, wenn ich ungenügend durchdachte und damit benutzerunfreundliche Formulare sehe. Da ich mir selber schon vor längerer Zeit eine Art Style-Guide für meine Formulare überlegt habe, möchte ich diesen hier einmal vorstellen. Ich gehe dabei von folgendem Modell aus:

EIN Kunde erteilt MEHRERE Kundenaufträge.
EIN Kundenauftrag ist von EINEM Kunden.
Also besteht eine 1:n-Beziehung zwischen tblKunde und tblKundenauftrag.

Zu EINEM Kundenauftrag gehören MEHRERE Materialarten.
EINE Materialart gehört zu MEHREREN Kundenaufträgen.
Also besteht eine m:n-Beziehung zwischen tblKundenauftrag und tblMaterialart.
Also brauchen wir eine Zwischentabelle tblKauf_Mat.



Es soll jetzt ein Formular zur Auftragsbearbeitung entwickelt werden. Den ganzen Entwicklungsprozess kann ich hier aus Platzgründen nicht beschreiben. Dazu gibt es in der Accessdatei, die ich unten zum Download anbiete, genügend Material. Ich habe dort nämlich jeden einzelnen Zwischenschritt vom allerersten Entwurf bis zum fertigen Formular dokumentiert. Außerdem finden sich dort auch zahlreiche Kommentare im VBA-Quelltext. Ich möchte hier jetzt vielmehr meine Design- und Funktionsprinzipien erläutern (Die Nummern beziehen sich auf das Bild unten!).

(1) Fenstertitel und Formulartitel bilden eine Einheit und sind zusammen zu lesen.

(2) Alle Formularelemente sind bündig und möglichst gleich breit ausgerichtet. Das braucht man nicht von Hand zurechtzufummeln - dafür gibt es in der Entwurfsansicht des Formulars entsprechende Befehle. Die Farben sind dezent gehalten - also nicht blau, rot, grün in einem Formular, sondern Abstufungen einer Farbe – z.B. dunkelblau / mittelblau / hellblau + schwarz und weiß.

(3) Auf dem Formular befinden sich kleine Hinweise zur Bedienung, damit die Software quasi selbsterklärend wird (Wer liest schon noch Handbücher?). Diese Hinweise erscheinen nach einem Click des Nutzers auf den Button mit dem Fragezeichen.

(4) Der Standardbutton zum Schließen des Fensters wird durch einen selbst programmierten Button ergänzt. Beide haben dieselbe Funktion.

(5) Wir Mitteleuropäer lesen von links nach rechts und von oben nach unten. In dieser Weise sollten daher auch die einzelnen Formularelemente angeordnet sein - in der Reihenfolge, in der sie der Benutzer höchstwahrscheinlich brauchen wird. Hier also: Links in der Liste einen Auftrag auswählen, dann rechts daneben die Auftragsdaten bearbeiten und schließlich unten das dazugehörige Material hinzufügen.

Die Liste ist nach Datumswerten absteigend sortiert, d.h. die neuesten Aufträge stehen oben. Das muss in den Listenfeld-Eigenschaften unter Daten / Datensatzherkunft eingestellt werden.

(6) Die Suche wird hier so realisiert: Es gibt Eingabefelder für die Suchkriterien („Datum“, „Nummer“ und „Beschreibung“). Der Nutzer clickt auf „Suchen“ und daraufhin werden in der Liste nur diejenigen Aufträge angezeigt, auf die die Suchkriterien passen. Auch Kombinationen von Suchkriterien sind möglich. Ein Click auf „Alle“ lässt in der Liste wieder alle Aufträge erscheinen.

(7) Fast jedes Formular braucht die drei Standardaktionen "Neu", "Speichern" und "Löschen". Die entsprechenden Buttons
- werden entweder beschriftet oder erhalten ein aussagekräftiges Bildchen
- befinden sich im farblich abgesetzten Fußbereich des Formulars
- werden von links nach rechts in der Reihenfolge angeordnet, in der sie gebraucht werden: Erst "Neu", dann "Speichern" und dann "Löschen".
- befinden sich auf jedem Formular immer in derselben Reihenfolge an der gleichen Stelle ("erwartungskonform"!)
- zeigen, wenn der Mauszeiger einige Sekunden darüber verharrt, einen sog. "SteuerelementTip-Text", damit der Benutzer genau weiß, was der Button bewirkt.

Neu:
Betätigt der Benutzer den "Neu"-Button, so werden die Liste und der "Löschen"-Button deaktiviert. Dadurch wird er gezwungen, auch wirklich das zu tun, was er vorhatte.

Speichern:
Nach dem Click auf den "Speichern"-Button werden die Liste und der "Löschen"-Button wieder aktiviert. Vor dem Speichern muss überprüft werden, ob der Benutzer die Pflichtfelder ausgefüllt hat - in diesem Fall Kunde, Beschreibung und Nummer. Wenn nicht, bekommt der Nutzer eine Warnmeldung.
Nach dem Speichern ist der gerade gespeicherte Datensatz in der Liste markiert. Das geschieht nicht automatisch, sondern muss programmiert werden.

Löschen:
Das Löschen erfolgt erst nach der Rückfrage "Wollen Sie wirklich ...?". Nach dem Löschen ist der erste Datensatz in der Liste markiert. Das geschieht auch nicht automatisch, sondern muss programmiert werden.
Achtung! Trick_17: Der „gelöschte“ Datensatz ist gar nicht gelöscht! Er wird nur in der Tabelle tblKundenauftrag durch den Eintrag des aktuellen Datums in der Spalte kauf_geloescht als „gelöscht markiert“:

```
strSQLtext = "UPDATE tblKundenauftrag SET kauf_geloescht=" & _
             Format(CDate(Date), "\#mm\/dd\/yyyy\#") & _
             " WHERE kauf_id=" & Me!kauf_id
CurrentDb.Execute strSQLtext
```

Im Formular frmAuftragsbearbeitung und in der darin enthaltenen Liste werden entsprechend nur solche Zeilen der Tabelle tblKundenauftrag dargestellt, die in der Spalte kauf_geloescht keinen Eintrag haben. Dadurch sind „gelöschte“ Daten jederzeit wieder herstellbar!

(8) Laut Datenmodell ist EIN Kundenauftrag von EINEM Kunden. Dieser wird also über ein Kombinationsfeld ausgewählt. Durch die Auswahl eines Kunden im Kombinationsfeld wird dessen Primärschlüssel als Fremdschlüssel in die Tabelle tblKundenauftrag eingetragen. Man beachte die Anzeige im Kombinationsfeld: Normalerweise kann man dort nur den Inhalte EINER Tabellenspalte anzeigen - also z.B. nur den Nachnamen, aber nicht Vorname UND Nachname. Mit einem kleinen Trick gelingt das aber trotzdem. Dazu habe ich die Datensatzherkunft des Kombinationsfeldes so definiert:

```
SELECT kun_id, kun_name & ", " & kun_vorname AS kundenname
FROM tblKunde
ORDER BY kun_name
```

Noch ein wichtiger Trick: Wenn bei offenem Auftragsbearbeitungs-Formular ein neuer Kunde angelegt wird, so erscheint dieser NICHT automatisch in dem Kombinationsfeld! Dafür muss man durch entsprechende Programmierung selber sorgen. Der folgende Code bewirkt, dass das Kombinationsfeld vor jeder Benutzung zuerst aktualisiert wird:

```
Private Sub cboKunde_GotFocus()
cboKunde.Requery
End Sub
```

Das Kombinationsfeld wird hier für zweierlei Zwecke genutzt: Zum einen zur ANZEIGE des Kunden, von dem der ausgewählte Auftrag ist; zum anderen aber auch zur EINGABE des Kundennamens bei einem neuen Auftrag. Das hat den Nachteil, dass durch einen unvorsichtigen Click in das Kombinationsfeld ein Auftrag von Meier in einen Auftrag von Schulze verwandelt wird. Dem kann man mit folgendem Code vorbeugen:

```
Private Sub cboKunde_Dirty(Cancel As Integer)
If IsNull(cboKunde.OldValue) Then Exit Sub
If MsgBox("Wollen Sie den Auftraggeber wirklich ändern?", vbYesNo + vbDefault-
Button2) = vbNo Then Cancel = True
End Sub
```

Beim erstmaligen Eintrag von Daten (OldValue=Null) wird nicht nachgefragt.

(9) Es gibt zumindest zwei Gelegenheiten, bei denen der Benutzer vom Auftragsbearbeitungs-Formular zum Kundenformular wechseln möchte:
Erstens: Er möchte sich einfach die Kundendaten noch einmal ansehen.
Zweitens: Der Auftrag ist von einem Kunden, den er noch nicht in seiner DB hat und er möchte ihn jetzt anlegen (Achtung: siehe "Requery-Trick" unter Punkt (8)!)
Damit der Benutzer auf einfache Weise zum Kundenformular kommt, ist die Beschriftung des Kombinationsfeldes als Button ausgelegt. Ein Click auf den Button "Kunde" öffnet das Kundenformular und zeigt auch gleich denjenigen Kunden an, der im Auftragsbearbeitungs-Formular angezeigt wurde. Auch das geschieht nicht automatisch, sondern muss programmiert werden. Der folgende Code übergibt dem Kundenformular den Primärschlüssel des im Kombinationsfeld angezeigten Kunden (=gebundene Spalte des Kombinationsfeldes):

```
Private Sub cmdKunde_Click()
DoCmd.OpenForm "frmKunden", , , , , , Str(cboKunde)
End Sub
```

Das Öffnungsargument Str(cboKunde) wird dann in der Form_Open-Prozedur des Kundenformulars benutzt, um den Kunden aus dem Kombinationsfeld des Auftragsbearbeitungs-Formulars im Kundenformular anzuzeigen:

```
Me!lstKunden = OpenArgs
```

(10) Die Gestaltung des Unterformulars ist ein Stück Geheimwissenschaft. Dazu lohnt sich wieder mal ein Blick auf's Datenmodell. Aus der m:n-Beziehung zwischen Kundenauftrag und Materialart hat sich ergeben, dass wir eine sog. "Zwischentabelle" tblKauf_Mat brauchen. Das Datenmodell liest sich also jetzt so:

Zu EINEM Kundenauftrag gehören MEHRERE kauf_mat.

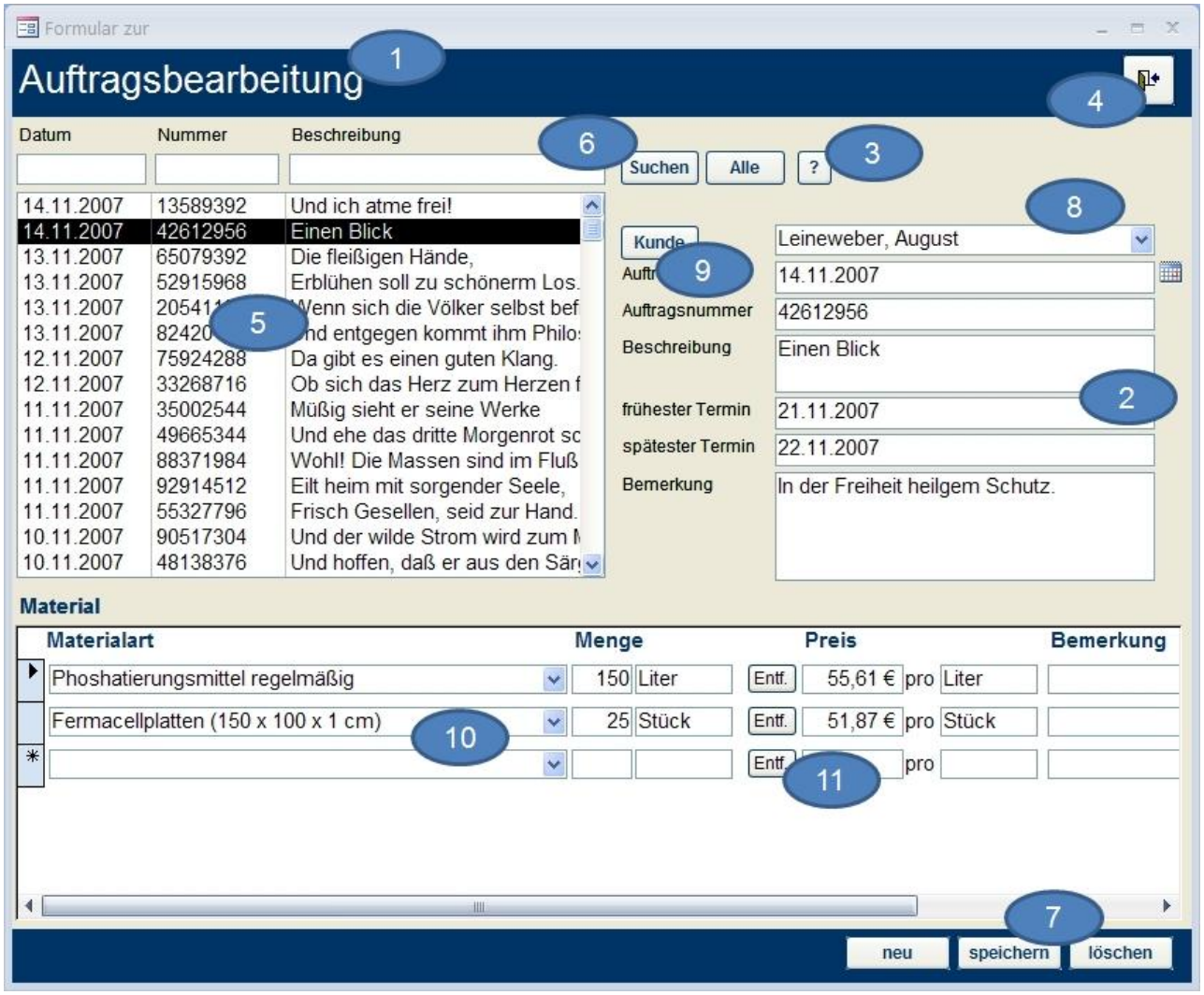
Was ist ein "kauf_mat"?? Nun, das ist zunächst mal eine künstliche Tabelle, die sich aus datenbanktechnischen Zwängen ergibt. Man könnte sie aber auch als "Materiallistenzeile" interpretie-

ren. (Achtung: NICHT als "Materialliste"!! Das würde ja bedeuten, dass zu EINEM Kundenauftrag MEHRERE Materiallisten gehören!)
Also: Zu EINEM Kundenauftrag gehören MEHRERE Materiallistenzeilen und zu EINER Materiallistenzeile gehört wiederum EINE Materialart. Der Datenbankentwickler denkt also etwas verquer! Der "gesunde Menschenverstand" sagt uns: "Zu EINEM Kundenauftrag gehören MEHRERE Materialarten; und zwar jeweils eine bestimmte Menge". Der Datenbankentwickler muss sein Denken aber streng am Datenmodell ausrichten - und das sagt ihm: "Zu EINEM Kundenauftrag gehören MEHRERE Materiallistenzeilen (=Mengen!) und zu jeder Materiallistenzeile gehört wiederum EINE Materialart".
Und so entwickelt er auch sein Unterformular: Es ist zunächst mal eine Liste der Zeilen aus der Tabelle tblKauf_Mat (= Unterformular vom Typ „Endlosformular“). Zu jeder einzelnen Zeile gehört laut Datenmodell EINE Materialart. Also fügen wir ein Kombinationsfeld ein, mit dessen Hilfe wir die Materialart auswählen können. Das funktioniert wieder genau so, wie das Kombinationsfeld cboKunde im Auftragsbearbeitungs-Formular: Durch den Click in das Kombinationsfeld cboMaterialart wählen wir einen mat_id aus (=gebundene Spalte des Kombinationsfeldes!) und tragen ihn als Fremdschlüssel in die Tabelle tblKauf_Mat ein.
Wenn das Unterformular fertig ist, fügen wir es in das Hauptformular ein. Dabei ist auf die richtige Verknüpfung zwischen Haupt- und Unterformular zu achten (im zweiten Schritt des Unterformular-Assistenten: „Verknüpfen von: kauf_id_f“ – „Verknüpfen nach: kauf_id“)).

(11) Wo man etwas hinzufügen kann, muss generell auch die Möglichkeit bestehen, es wieder wegzunehmen. Darum gibt es in jeder Zeile des Unterformulars einen "Entf."-Button. Er macht folgendes:

```
strSQLtext = "DELETE FROM tblKauf_Mat WHERE kaufmat_id=" & Str(Me!kaufmat_id)
'MsgBox strSQLtext
CurrentDb.Execute strSQLtext
Requery
```

- natürlich erst nach Rückfrage beim Benutzer! Man beachte wieder eine Kleinigkeit beim Fenster "Wollen Sie wirklich ...?": Dort ist der "Nein"-Button etwas stärker umrandet. Dadurch wird der sog. "DefaultButton" markiert. Er wird betätigt, wenn der Benutzer nicht mit der Maus clickt, sondern einfach auf die Enter-Taste drückt. Dadurch sollte immer die ungefährlichere der beiden Alternativen ausgelöst werden. Dieses Verhalten wird durch "vbDefaultButton2" in "Private Sub cmdLoeschen_Click()" programmiert.
Noch ein Tipp für die VBA-Entwicklung: Wenn man ein SQL-Statement ausführen will, sollte man es sich erst einmal zur Sichtkontrolle anzeigen lassen. Den entspr. Befehl („MsgBox sqltext“) sollte man nicht löschen, sondern mit Kommentarzeichen stehen lassen. Wer weiß, wofür man ihn noch einmal gebrauchen kann!


Formular zur
Auftragsbearbeitung
1
4
Datum
Nummer
Beschreibung
6
Suchen
Alle
?
3
14.11.2007 13589392 Und ich atme frei!
14.11.2007 42612956 Einen Blick
13.11.2007 65079392 Die fleißigen Hände,
13.11.2007 52915968 Erblühen soll zu schönerm Los.
13.11.2007 Wenn sich die Völker selbst bef
13.11.2007 nd entgegen kommt ihm Philo
12.11.2007 75924288 Da gibt es einen guten Klang.
12.11.2007 33268716 Ob sich das Herz zum Herzen f
11.11.2007 35002544 Müßig sieht er seine Werke
11.11.2007 49665344 Und ehe das dritte Morgenrot sc
11.11.2007 88371984 Wohl! Die Massen sind im Fluß
11.11.2007 92914512 Eilt heim mit sorgender Seele,
11.11.2007 55327796 Frisch Gesellen, seid zur Hand.
10.11.2007 90517304 Und der wilde Strom wird zum
10.11.2007 48138376 Und hoffen, daß er aus den Sär
5
8
Kunde
Leineweber, August
9
Auftr
14.11.2007
Auftragsnummer
42612956
Beschreibung
Einen Blick
2
frühester Termin
21.11.2007
spätester Termin
22.11.2007
Bemerkung
In der Freiheit heilgem Schutz.
Material
Materialart
Menge
Preis
Bemerkung
Phoshatierungsmittel regelmäßig
150 Liter
Entf.
55,61 € pro Liter
Fermacellplatten (150 x 100 x 1 cm)
10
25 Stück
Entf.
51,87 € pro Stück
Entf.
pro
11
7
neu
speichern
löschen