



Valstr()

Die Funktionen VAL() und VALSTR() sind bereits seit langem in der tdbengine bekannt.

Was man aber z.B. mit VALSTR() lösen kann, möchte ich an folgendem Beispiel zeigen:

Die Sortierung von markierten Datensätzen nach einem Datenfeld, welches in der gleichen Tabelle angesiedelt ist stellt kein Problem dar.

Über die Funktion sortmark() hat man dies schnell erledigt.

Wie aber kann man über ein Feld sortieren, welches nicht in der Primärtabelle steht?

Wir gehen von folgender Struktur aus:

in der Mitarbeitertabelle gibt es einen Linkfeld mit der Bezeichnung \$AutoID.Funktion, welches auf die Tabelle Funktion.dat verweist. Innerhalb der Tabellefunktion gibt es ein numerisches Feld mit der Bezeichnung SortID.

Wir möchten nun in der Mitarbeitertabelle die markierten Mitarbeiter nach der Funktion sortieren.

Hierzu können wir folgende, von Ulrich Kern geschriebene, universelle Prozedur verwenden:

```
Procedure sort_linked(db:INTEGER;sortstr, rev : STRING)
```

```
Var s_sub : STRING = dbname(db)
```

```
Var s : STRING[,]
```

```
Var i, j : Integer
```

```
PrimTable(db); Access(db,"Markierung")
```

```
InitArray(s[nmarks(db),1])
```

```
SUB _s_sub
```

```
s[i,0]:=valstr(sortstr)
```

```
s[i,1]:=str(recno(db))
```

```
i++
```

```
ENDSUB
```

```
StrSort(s,i-1)
```

```
delmarks(db);
```

```
nloop(j,i-1,setmark(db,val(s[j,1])))
```

```
IF rev[1]='-' THEN revmarks(db) END
```

```
access(db,-2)
```

```
EndProc
```

Und mit folgendem Aufruf sortieren:

```
sort_linked(M,"STR($MITARBEITER.AutoID_Funktion.SortID,4,0,'0')","-");
```

Da es sich um ein numerisches Feld handelt, muss man mit STR() arbeiten.

Der letzte Parameter gibt übrigens an, ob aufsteigend oder absteigend sortiert wird.



Damit dieser Aufruf nicht auf einen Fehler läuft, muss man unbedingt nach dem Öffnen der Tabellen RELATION aufrufen.

```
Var M : Integer = OpenDB("../database/mitarbeiter.dat")  
Var F : Integer = OpenDB("../database/funktion.dat")  
Relation
```

Mit Aufruf von RELATION wird intern eine Beziehungstabelle befüllt. Dabei werden die Tabellen der Reihenfolge des Öffnen nach gelesen. Jede Tabelle wird dabei nach Linkfelder durchsucht. Sobald ein Linkfeld gefunden wurde, wird eine Beziehung zwischen den beiden Tabellen eingetragen. In unserem Beispiel würde folgende Relationsbeziehung in der internen Beziehungstabelle stehen:

```
MITARBEITER[AutoID_Funktion]=FUNKTION[AutoID]
```

Gibt es nun in einer Tabelle mehrere Linkfelder auf die gleiche Tabelle, wie hier z.B. 3 Linkfelder auf die Tabelle Funktion

```
[STRUCTURE]  
KonzernID,STRING,20  
Name,STRING,50  
Vorname,STRING,50  
AutoID_Funktion,LINK,funktion  
AutoID_Ersatzfunktion,LINK,funktion  
AutoID_Ersatzfunktion2,LINK,funktion
```

so wird die Beziehung nur für das erste Linkfeld aufgebaut.

Aus diesem Grund wird der folgende Aufruf auf einen Fehler laufen:

```
sort_linked(M,"STR($MITARBEITER.AutoID_Ersatzfunktion  
.SortID,4,0,"','0')", "-");
```

Unter Umständen ist aber die Reihenfolge des Öffnen von Tabellen von Bedeutung.

Wird die interne Beziehungstabelle aufgrund der Reihenfolge der geöffneten Tabelle nicht so aufgebaut, wie man es erwartet so wird der Aufruf von z.B. `sort_linked()` mit einem Laufzeitfehler beendet, weil VALSTR den Ausdruck nicht auswerten kann.

In diesem Fall muss man die Reihenfolge des Öffnens so ändern, dass die Beziehungstabelle die benötigten Beziehungen beinhaltet.

Sicher wird uns Ulrich Kern in einen der folgenden Developer News noch internes zu ValStr und Relation verraten.