Datenbankfrontends mit OpenOffice.org

Manfred Noack

Frontends für die Bearbeitung von Datenbanken gibt es in beliebiger Funktionstiefe. Möchte man aber nur schnell und dennoch komfortabel auf eine Datenbank zugreifen oder eine Usability-Studie für ein neues Datenbank-Frontend durchführen, bietet auch OpenOffice.org einige interessante Möglichkeiten.

Bild 1: Aufrufen des Tabellenassistenten

Frontends für das Verwalten von Datenbankinhalten gibt es reichlich. Sie besitzen meist einen vielfältigen Funktionsumfang und können oft auch nur von Experten installiert und konfiguriert werden. Für den normalen Endanwender, der einfach nur Daten abfragen oder bearbeiten will, sind Softwarepakete wie phpMyAdmin oder PgAdmin eher ungeeignet. Benutzergruppen, die nur bestimmte Datensätze in einer Datenbank abfragen oder bearbeiten sollen, bekommen deshalb in der Regel speziell auf sie zugeschnittene Frontends zur Verfügung gestellt. Das ist sinnvoll, wenn man das Datenbank-Frontend einer größeren Gruppe von Anwendern zur Verfügung stellen will. Möchte man aber nur schnell und dennoch komfortabel auf eine Datenbank zugreifen oder eine Usability-Studie für ein neues Datenbank-Frontend durchführen, bietet auch OpenOffice.org einige interessante Möglichkeiten. Out-of-the-box kann man sich mit dem Officepaket von Sun mit vielen bekannten Datenbanken verbinden und formschöne Eingabemasken entwerfen und das unabhängig davon, ob die Daten in der Office-Anwendung selbst weiterverarbeitet werden sollen oder nicht. Nachfolgend soll deshalb am Beispiel einer DVD-Datenbank gezeigt werden, wie man mit OpenOffice.org und OOo Basic recht schnell und einfach zu einem benutzerfreundlichen Frontend für alle gängigen Datenbanken kommt.

Um zu zeigen, daß das Frontend, das in diesem Beitrag entworfen wird, universal einsetzbar ist, sollen zwei unterschiedlichen Datenbanken angesprochen werden. OpenOffice.org liefert mit Base bereits ein Datenbank-Frontend mit, mit dem man neben der ebenfalls mitgelieferten HSQL-Datenbank auch eine Vielzahl derzeit populärer Datenbank-Server anbinden kann. Während HSQL nur lokal zur Verfügung steht, können Client/Server-Datenbanken wie MySQL auch über das Netzwerk angesprochen werden. Um mit OO.o-Basic auf eine Datenbank zugreifen zu können, muß man zunächst mit Base eine Datenbankverbindung anlegen, die später im Basic-Programm referenziert wird.

Ruft man Base auf, erscheint zunächst ein Konfigurationsmenü, in dem man entweder eine neue Datenbank anlegen oder an eine bereits bestehende Datenbank anbinden kann. Für die DVD-Sammlung wird zunächst eine neue HSQL-Datenbank entworfen. Nach dem Speichern der Beispiel-

Bild 2: Datenbankfelder

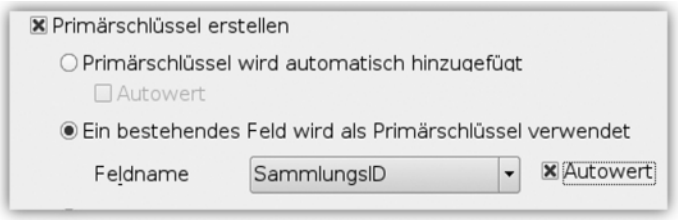

Bild 3: Dialog zur Primärschlüsselauswahl

datenbank mit dem Namen *dvds* wird mit dem Assistenten die Tabelle erzeugt.

Die Datenbank-Anbindung

OpenOffice.org liefert in der Kategorie *Privat* bereits die Vorlage für die DVD-Sammlung aus, in der für die spätere Tabelle die Spalten *SammlungsID*, *Filmtitel*, *Darsteller*, *Regisseur* und *Genre* ausgewählt werden. Im zweiten Schritt können die Standard-Vorgaben für die Datentypen übernommen werden.
Als Primärschlüssel soll die SammlungsID mit automatischer Inkrementierung (Autowert) dienen.
Zum Schluß wird der Tabelle noch der Name *DVD-Sammlung* gegeben und schon ist sie bereit für das Eintragen der ersten Daten. Im nächsten Schritt wird eine MySQL-Datenbank mit der gleichen Struktur angelegt. Es wird an dieser Stelle davon ausgegangen, daß ein MySQL-Datenbankserver bereits konfiguriert ist und auch läuft.
Um die Tabelle anlegen zu können, wird zuerst eine Textdatei mit ein paar SQL-Anweisungen geschrieben, mit denen die Datenbanktabelle angelegt wird, dabei ist unbedingt darauf zu achten, daß Strings in Backticks stehen müssen:

```
CREATE DATABASE IF NOT EXISTS `dvds`;
USE `dvds`;
DROP TABLE IF EXISTS `DVD-Sammlung`;
CREATE TABLE `DVD-Sammlung` (
  `SammlungsID` int(10) unsigned
     NOT NULL auto_increment,
  `Filmtitel` varchar(100)
     collate latin1_general_ci
     NOT NULL default '',
  `Darsteller` varchar(100)
     collate latin1_general_ci
     NOT NULL default '',
  `Regisseur` varchar(100)
     collate latin1_general_ci
     NOT NULL default '',
  `Genre` varchar(100)
     collate latin1_general_ci
     NOT NULL default '',
  PRIMARY KEY (`SammlungsID`)
)
```

Anschließend wird der MySQL-Server auf der Kommandozeile mit den SQL-Befehlen gefüttert, die in der Datei *dvd.sql* abgelegt wurden:

```
# mysql -u dbadmin -p < dvd.sql
```

Ist beim Anlegen der Datenbank alles gutgegangen, kann man sie anschließend in OpenOffice.org registrieren. Dafür wird im Datenbank-Einrichtungsassistent zunächst eine Verbindung zu einer bestehenden Datenbank ausgewählt, anschließend wird zweimal auf *Weiter* geklickt, um im Schritt danach eine JDBC-Verbindung herzustellen. Am Ende der Einrichtungsprozedur sollte man die Verbindung noch testen. Falls dabei der Fehler »JDBC-Klasse konnte nicht geladen werden« erscheint, muß noch der JDBC-Treiber nachinstalliert werden. Bei einigen Linux/Unix-Distributionen wird er nicht automatisch mit OpenOffice.org installiert.
War der JDBC-Test erfolgreich, kann man noch die nächsten Schritte des Wizards durchlaufen, in denen die Verbindung getestet und am Ende die Einstellungen gespeichert werden.
Stimmen die Einstellungen für Server, Port, Benutzername und Kennwort, dürfen bei den Verbindungstests keine Fehler mehr auftreten. Sollte wider Erwarten dennoch eine Verbindung nicht zustande kommen, muß geprüft werden, ob die Zugriffsrechte des Datenbankservers richtig konfiguriert sind.
Hat soweit alles funktioniert, sind danach im Menü *Extras|Optionen|OpenOffice.org Base|Datenbanken* die bei-den Datenbankverbindungen mit ihrem Registrierungsnamen aufgeführt. Diese Namen werden später im Programmquelltext für den Verbindungsaufbau zu der jeweiligen Datenbank benötigt.

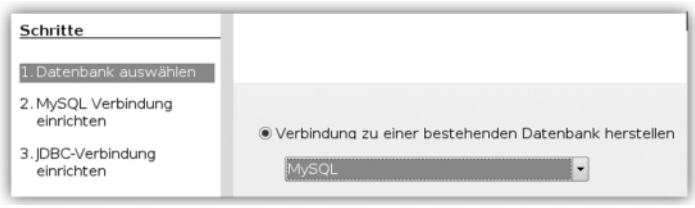

Bild 4: Der Dialog für die Datenbankanbindung

Das Frontend

Sind die Datenbanken eingerichtet, geht es jetzt an das Entwickeln des

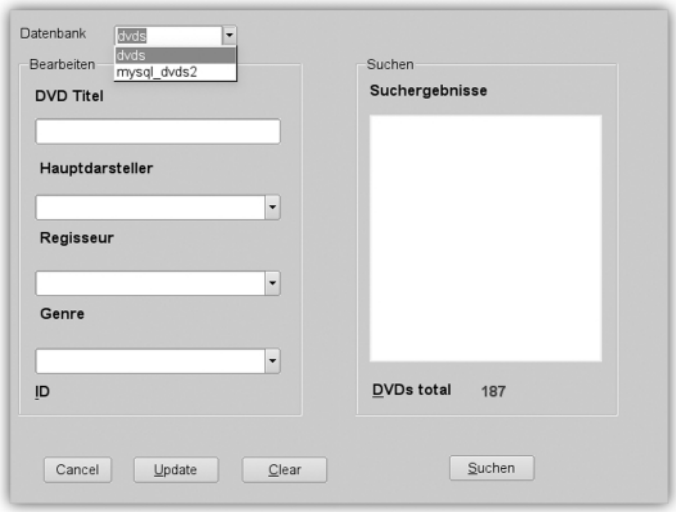

Bild 5: Das Datenbankfrontend

Frontends. Bild 5 zeigt ein Beispiel, links oben ist der Datenbankauswahldialog zu sehen. Auf der linken Seite befinden sich die Comboboxen für die gleichzeitige Bearbeitung der Daten und die Suche. In diesem Beispiel sind das Eingabefeld *DVD-Titel* und die Comboboxen miteinander UND-verknüpft. Nach dem Titel gesucht wird in diesem Beispiel mit dem SQL-Operator *LIKE value%*, wodurch eine Suche nach Teilstrings möglich ist. Die Comboboxen sind mit den entsprechenden Werten aus der Datenbank vorbelegt und können für die Suche mit dem Text im Titelfeld kombiniert werden.

Bild 6: Datenbanken

Im Frame auf der rechten Seite werden die Suchergebnisse ausgegeben. In der Ergebnisliste werden die Sammlungs-ID und der Filmtitel angezeigt. Wird auf ein Ergebnis geklickt, werden die Daten zur Weiterverarbeitung in den linken Bearbeiten-Frame übernommen, in dem die Daten bearbeitet und ergänzt werden können.

Nachdem das Frontend den eigenen Vorstellungen entsprechend gestaltet ist, kann man sich an den Quelltext machen. Er ist auf der freeX-DVD im Verzeichnis */freeX/80-OOBasic* einmal als Textdatei mit Zeilennummern (auf die sich die Ausführungen hier beziehen) und als Basic-Quelltext ohne diese Orientierungshilfe abgelegt.

Der Programmquelltext

Damit sich die Anwendung später leicht erweitern und anpassen läßt,

Name	Zeile	Dialog	Event	Beschreibung
getDVDCount	101	–	Main, InsertDVDdata	Liefert die Anzahl der DVDs und setzt DVDCountLabel.
getComboBoxData	120	–	Main, clearComboBoxes,DBChange	Füllt die Comboboxen 1-3.
clearDLG	143	ClearButton	Maustaste losgelassen	Löscht alle Dialoginhalte.
clearComboBoxes	158	–	ClearDLG	Löscht Inhalt der Comboboxen.
clearListBoxes	173	–	ClearDLG, SearchDVDdata	Löscht die Suchergebnisse.
UpdateDVD	183	UpdateButton	Maustaste losgelassen	Prüft, ob Insert oder Update von Daten.
checkDVDTitle	207	–	UpdateDVD	Prüft, ob ein Titel bereits vorhanden ist. Wenn ja, wird die SammlungsID zurückgeliefert.
getDlgData	235	–	UpdateDVDdata, InsertDVDdata	Sammelt die aktuellen Werte der Dialoge im Bearbeiten-Feld.
SearchDVDdata	298	SuchButton	Maustaste losgelassen	Baut den Suchstring zusammen, startet die Suche und füllt mit den Ergebnisse ListBox1.
getListBoxData	343	ListBox1	Maustaste losgelassen	Liest das angeklickte Suchergebnis aus der Liste und liefert die SammlungsID zurück.
setDialogData	360	–	getListBoxData	Setzt die Werte des ausgewählten Suchergebnisses in die Dialoge im Bearbeiten-Frame.
DBChange	390	Combobox4	Status geändert	Wechselt die Datenbank.

Tabelle 1: Frontend-Funktionen im Programm

werden zuerst einige globale Werte in Arrays abgelegt, anschließend werden diese Variablen in der Routine *Main* belegt. Dieses triviale Vorgehen soll hier nicht weiter beleuchtet werden. Interessent wird es ab Zeile 31, wo der Dialog geladen wird:

```
DialogLibraries.LoadLibrary("Standard")
Dlg = CreateUnoDialog(_
      DialogLibraries.DVD.DVD_dlg)

DDlg = Dlg.getControl("ComboBox4")
DDlg.additems(Databases,0)
```

Der Datenbankauswahldialog bekommt in Zeile 36 die Standard-Datenbank voreingestellt:

```
DDlg.Text = Databases(0)
```

Danach füllt die Prozedur *getComboBoxData()* zunächst die Comboboxen und *getDVDCount()* liest die Anzahl der DVDs aus, die dem Label *DVDCount* zugeordnet werden. Am Ende von *Main* wird der Dialog erzeugt und angezeigt.

Codierung des Datenbankzugriffs

Der eigentliche Datenbankzugriff findet in der Funktion *DBConnect()* statt, die ein Verbindungsobjekt zurückliefert, mit dem die Prozeduren die SQL-Befehle an die Datenbank übergeben und anschließend das Ergebnis zurückbekommen.

Um an die gespeicherten Daten heranzukommen, muß zunächst ein Datenbankkontext erzeugt werden (Zeile 55), mit dem man auf die Datenquelle zugreifen kann (Zeile 56). Hierfür wird beim Aufruf der Methode *getByName(dbregistername)* als Parameter der Registrierungsname der Datenbank übergeben.

Dieser Registrierungsname entspricht der aktiven Auswahl in dem Dialogfeld von *Combobox4* (Zeile 52 bis 53), der zuvor die Registrierungsnamen der Datenbanken zugewiesen wurden (Zeile 34 bis 36). In den nachfolgenden Zeilen (56 bis 61) wird überprüft, ob die Dantenbank-Verbindung ein Paßwort benötigt. Wenn ja, erscheint ein Paßwortdialog, bevor die Variable *Connection* das Verbindungsobjekt zurückliefert.

Damit die SQL-Befehle auch datenbankübergreifend funktionieren, muß man beim Zusammenbau der SQL-Anweisungen auf die Anführungsstriche achten. Während bei den SELECT-Befehlen in MySQL und HSQL gleichermaßen ', ` und " erlaubt sind, geht dies bei den INSERT- oder UPDATE-Befehlen nicht. Während die HSQL-Datenbank nur doppelte Hochkommata (") in den die SQL-Anweisungen zuläßt, verlangt MySQL hier Backticks (`). Damit man nun nicht für jede Datenbank eine eigene SQL-Anweisung entwickeln muß, die sich nur durch die Hochkommata unterscheidet, wurde die Funktion *tickCheck()* (Zeile 398) eingeführt, die je nach Datenbanktyp das passende Hochkomma zurückliefert. Die beiden Prozeduren *UpdateDVDdata()* und *InsertDVDdata* rufen die Funktion auf und bauen mit dem zurückgelieferten Hochkommata den SQL-String zusammen (Zeile 261 und 284).

Der Zugriff auf die Datenbank über ihren Registrierungsnamen ist äußerst praktisch. Der Datenbankdialog übergibt bei jedem Wechsel einfach den Registrierungsnamen an die Funktion *DBChange()* (Zeile 390) und ermöglicht es, die Datenbank sozusagen on-the-Fly zu wechseln. Da die Eingabefelder bei einem Datenbankwechsel nicht geleert werden, lassen sich beispielsweise Daten direkt von einer Datenbank in eine andere kopieren. Dabei ist in diesem Beispiel allerdings zu berücksichtigen, daß bei identischen Datensätzen die *SammlungsID* unter Umständen nicht übereinstimmen, da diese ja automatisch erzeugt wird.

Ein anderes Anwendungsszenario wäre beispielsweise der Abgleich einzelner Datensätze in einer entfernten produktiven und einer lokalen Backup-Datenbank. Die Datensätze lassen sich so relativ bequem überprüfen und synchronisieren – wobei diese Vorgehensweise nicht unbedingt auf SQL-Datenbanken beschränkt ist. Wie in Bild 6 zusehen ist, bietet OpenOffice.org auch Möglichkeiten des Zugriffs auf andere Datenquelle wie beispielsweise ein Tabellendokument oder das KDE-Adreßbuch.

Universelles Frontend mit wenig Aufwand

Es wurde versucht, das dargestellte Beispiel so generisch wie möglich zu halten, so daß die Anwendung leicht abgeändert und den speziellen Bedürfnissen angepaßt werden kann.

Die Tabellen 1 und 2 listen die Funktionen und Prozeduren ihrem hauptsächlichen Verwendungszweck nach auf. Dabei sind die reinen Datenbankfunktionen in der Minderheit; eigentlich sind es nur die drei Funktionen *DBConnect*, *DBSelect* und *DBUpdate*, die die Kommunikation mit der Datenbank übernehmen. Mit den Funktionen für die Frontend-Steuerung soll aufgezeigt werden, daß mit relativ wenig Quelltext der Bedienkomfort nicht auf der Strecke bleiben muß. ◆

Name	Zeile	Beschreibung
DBConnect	47	Datenbankanbindung.
DBSelect	72	SELECT-Abfragen.
DBUpdate	89	Einfügen und Ändern von Daten.
UpdateDVDdata	252	Datenbank-Update.
InsertDVDdata	276	Datenbank-Insert.
tickCheck	398	Liefert die richtigen Hochkommata zurück.

Tabelle 2: Datenbank-Funktionen im Programm