OpenOffice – HSQLDB: Daten intern oder Daten extern?

In den Mailinglisten kommen hin und wieder Anfragen, wie eventuell noch Daten aus einer Datei zu retten sind, die bei einem Totalabsturz des PCs irgendwie beschädigt wurde. Eine entsprechende Beschreibung hierzu hatte ich im Netz gefunden und für die Leute, die mit englischen Texten Probleme haben, ins Deutsche übertragen.

Wer die erste Datenbankreparatur hinter sich hat fragt sich natürlich, ob nicht irgendwie für etwas mehr Datensicherheit gesorgt werden kann. Hierfür ist im Anschluss der Weg beschrieben, von der internen Datenbank auf eine externe Datenbank umzustellen und die gesamten Daten, Formulare, Abfragen und Berichte dennoch weiter nutzen zu können.

Wer dann schließlich noch die Möglichkeit sucht, mit mehreren Personen auf die eigene Datenbank zuzugreifen wählt schließlich die Server-Version der HSQLDB und verteilt sein *.odb-Päckchen mit den erstellten Abfragen, Formularen und Berichten an die anderen Gruppenmitglieder, die im Serverbetrieb dann alle den gleichen Datenbestand nutzen können.

Aber Achtung: Bei all dem lauern auch Fallen, die je nach verwendeter OpenOffice-Version dazu führen, dass sich interne Datenbanken plötzlich nicht mehr öffnen lassen. Was dann zu unternehmen ist wird entsprechend in dem Kapitel zur Datenbankreparatur erklärt.

Datenbankreparatur für *.odb-Dateien

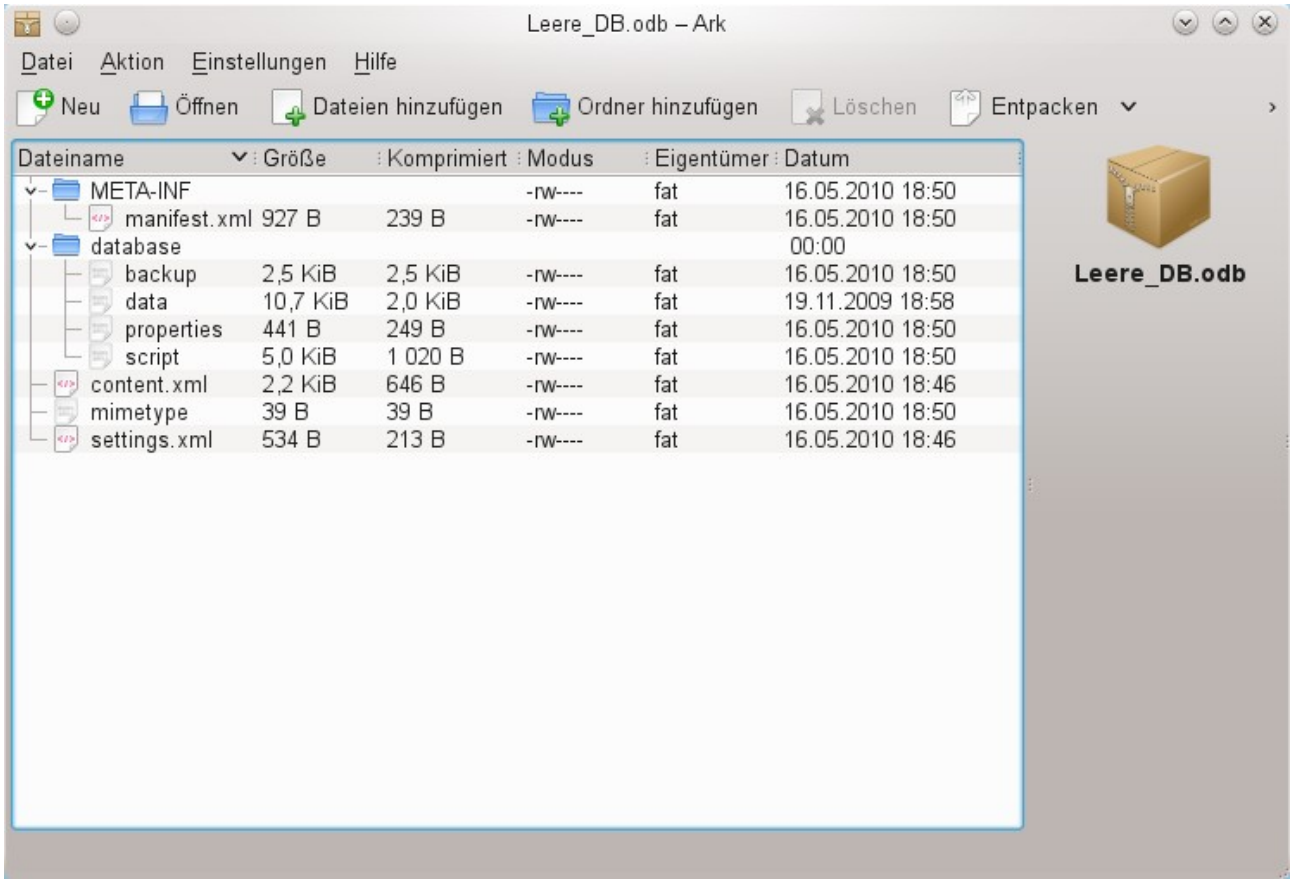
Regelmäßige Datensicherung sollte eigentlich Grundlage für den Umgang mit dem PC sein. Sicherheitskopien sind so der einfachste Weg, auf einen halbwegs aktuellen Datenstand zurückgreifen zu können. Doch in der Praxis mangelt es eben häufig an dieser Stelle.

Bei plötzlichen Abstürzen des PC kann es passieren, dass geöffnete Datenbanken von OpenOffice (interne Datenbank HSQLDB) nicht mehr zu öffnen sind. Stattdessen wird beim Versuch, die Datenbank zu öffnen, nach einem entsprechenden Filter für das Format gefragt.

Das Ganze liegt daran, dass Teile der Daten der geöffneten Datenbank im Arbeitsspeicher liegen und lediglich temporär zwischengespeichert werden. Erst beim Schließen der Datei wird die gesamte Datenbank in die Datei zurückgeschrieben und gepackt.

Um eventuell doch nach an die Daten zu kommen kann das folgende Verfahren hilfreich sein:

1. Fertige eine Kopie Deiner Datenbank für die weiteren Schritte an.
2. Versuche die Kopie mit einem Packprogramm zu öffnen. Es handelt sich bei der *.odb-Datei um ein gepacktes Format, ein JAVA-Archiv. Lässt sich die Datei so nicht direkt öffnen, so funktioniert das Ganze vielleicht auch über die Umbenennung der Endung von *.odb zu *.zip.
   Funktioniert das Öffnen nicht, so ist vermutlich von der Datenbank nichts mehr zu retten.
3. Folgende Verzeichnisse siehst Du nach dem Öffnen einer Datenbankdatei im Packprogramm auf jeden Fall:



4. Die Datenbankdatei muss ausgepackt werden. Die entscheidenden Informationen für die Daten liegen im Unterverzeichnis „database“ in den Dateien „data“ und „script“.
5. Gegebenenfalls empfiehlt es sich, die Datei „script“ einmal anzuschauen und auf Ungereimtheiten zu überprüfen. Dieser Schritt kann aber auch erst einmal zum Testen

übersprungen werden. Die „script“-Datei enthält vor allem die Beschreibung der Tabellenstruktur.

6. Gründe eine neue, leere Datenbankdatei und öffne diese Datenbankdatei mit dem Packprogramm.
7. Ersetze die Dateien „data“ und „script“ aus der neuen Datenbankdatei durch die unter „4.“ entpackten Dateien.
8. Das Packprogramm muss nun geschlossen werden. War es, je nach Betriebssystem, notwendig, die Dateien vor dem Öffnen durch das Packprogramm nach *.zip umzubenennen, so ist das jetzt wieder nach *.odb zu wandeln.
9. Öffne die Datenbankdatei jetzt mit OpenOffice und Du kannst hoffentlich wieder auf Deine Tabellen zugreifen.
10. Wie weit sich jetzt auch Abfragen, Formulare und Berichte auf ähnliche Weise wiederherstellen lassen bleibt dem weiteren Testen überlassen.

Siehe hierzu auch: http://user.services.openoffice.org/en/forum/viewtopic.php?f=83&t=17125

Wenn, wie auf den folgenden Seiten beschrieben, die externe HSQLDB verwendet wird, kann eventuell ein weiteres Problem mit den *.odb-Dateien in Verbindung mit manchen OpenOffice-Versionen auftauchen. Wird eine externe HSQLDB genutzt, so ist der sicherste Weg der über das hasldb.jar-Archiv, das mit OpenOffice mitgeliefert wird. Wird ein anderes Archiv verwendet, so kann das dazu führen, dass die internen Datenbanken plötzlich nicht mehr zugänglich sind. Dies liegt daran, dass OpenOffice Schwierigkeiten hat, zwischen interner und externer HSQLDB zu unterscheiden und Meldungen von einem Versionskonflikt produziert. OOo 3.1.1 scheint hiermit keine Probleme zu haben, OOo 3.3 leider schon. Eine aktuelle Version des Programms bedeutet hier nicht unbedingt eine geringere Zahl an Problemstellen.

Lassen sich interne Datenbanken nicht mehr öffnen so hilft pragmatisch erst einmal nur, OOo 3.1.1 zu nutzen. Ansonsten muss als externe Datenbank die mitgelieferte hsqldb.jar-Datei genutzt werden. Außerdem muss aus der *.odb-Datei das database-verzeichnis extrahiert werden. Die Datei properties hat hier einen Eintrag, der in OOo 3.3 zu dieser Fehlermeldung führt:

```
version=1.8.1
```

steht in Zeile 11.

Diese Zeile ist zu ändern auf

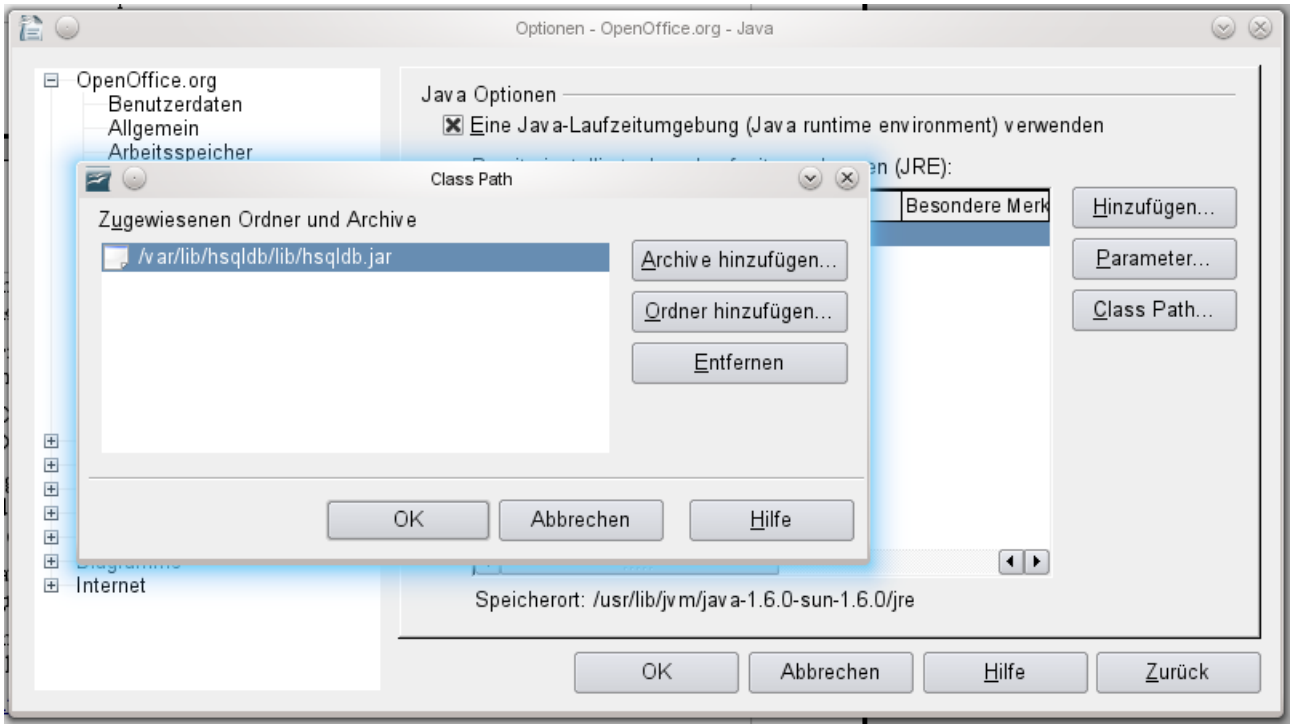
```
version=1.8.0
```

Danach ist das database-Verzeichnis wieder in das *.odb-Päckchen einzulesen und die Datenbank lässt sich auch wieder unter OOo 3.3 öffnen.

Datenbankverbindung zu einer externen HSQLDB

Die interne HSQLDB unterscheidet sich erst einmal nicht von der externen Variante. Wenn, wie im Folgenden beschrieben, erst einmal nur der Zugriff auf die Datenbank nach außerhalb gelegt werden soll, dann ist keine Serverfunktion erforderlich. Hier reicht schlicht das Archiv, was in OpenOffice mitgeliefert wurde. Im OpenOffice Pfad liegt unter /program/classes/hsqldb.jar. Die Verwendung dieses Archivs ist die sicherste Variante, da dann keine Versionsprobleme auftauchen.

Die externe HSQLDB steht unter http://hsqldb.org/ zum Download frei zur Verfügung. Ist die Datenbank installiert, so sind in OpenOffice folgende Schritte zu vollziehen:



Der Datenbanktreiber muss, sofern er nicht in dem Pfad der Java-Runtime liegt, als ClassPath unter Extras – Optionen – Java hinzugefügt werden.

Die Verbindung zu der externen Datenbank erfolgt über JDBC. Die Datenbankdateien sollen in einem bestimmten Verzeichnis abgelegt werden. Dieses Verzeichnis kann beliebig gewählt werden. Es liegt in dem folgenden Beispiel im home-Ordner. Nicht angegeben ist hier der weitere Verzeichnisverlauf sowie der Name der Datenbank.

Wichtig, damit auch Daten in die Datenbank über die GUI geschrieben werden können: ergänzend neben dem Datenbanknamen muss stehen `;default_schema=true`

Also:
`jdbc:hsqldb:file:/home/PfadZurDatenbank/Datenbankname;default_schema=true`

In dem Ordner befinden sich die Dateien

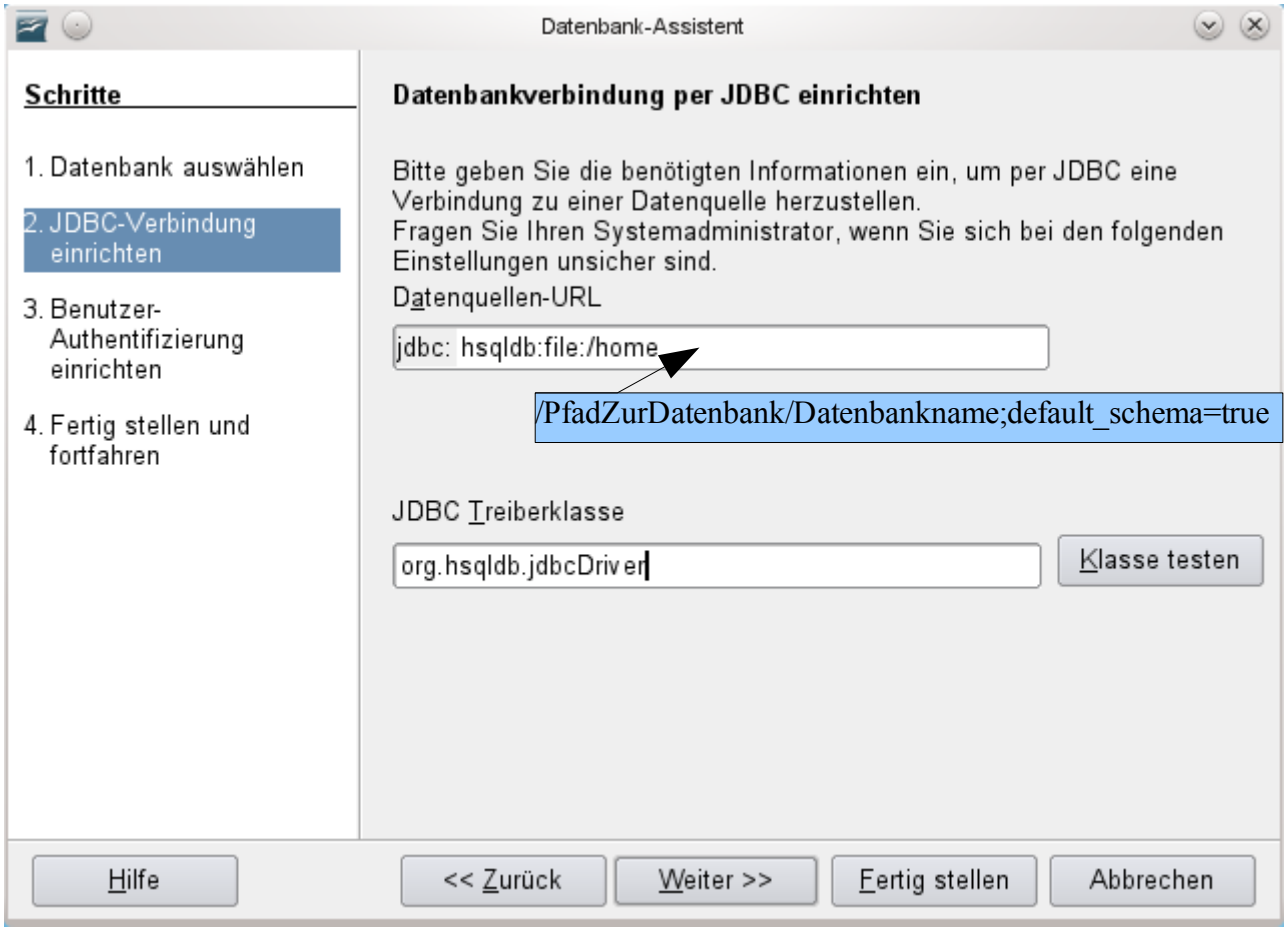
`Datenbankname.backup`

`Datenbankname.data`

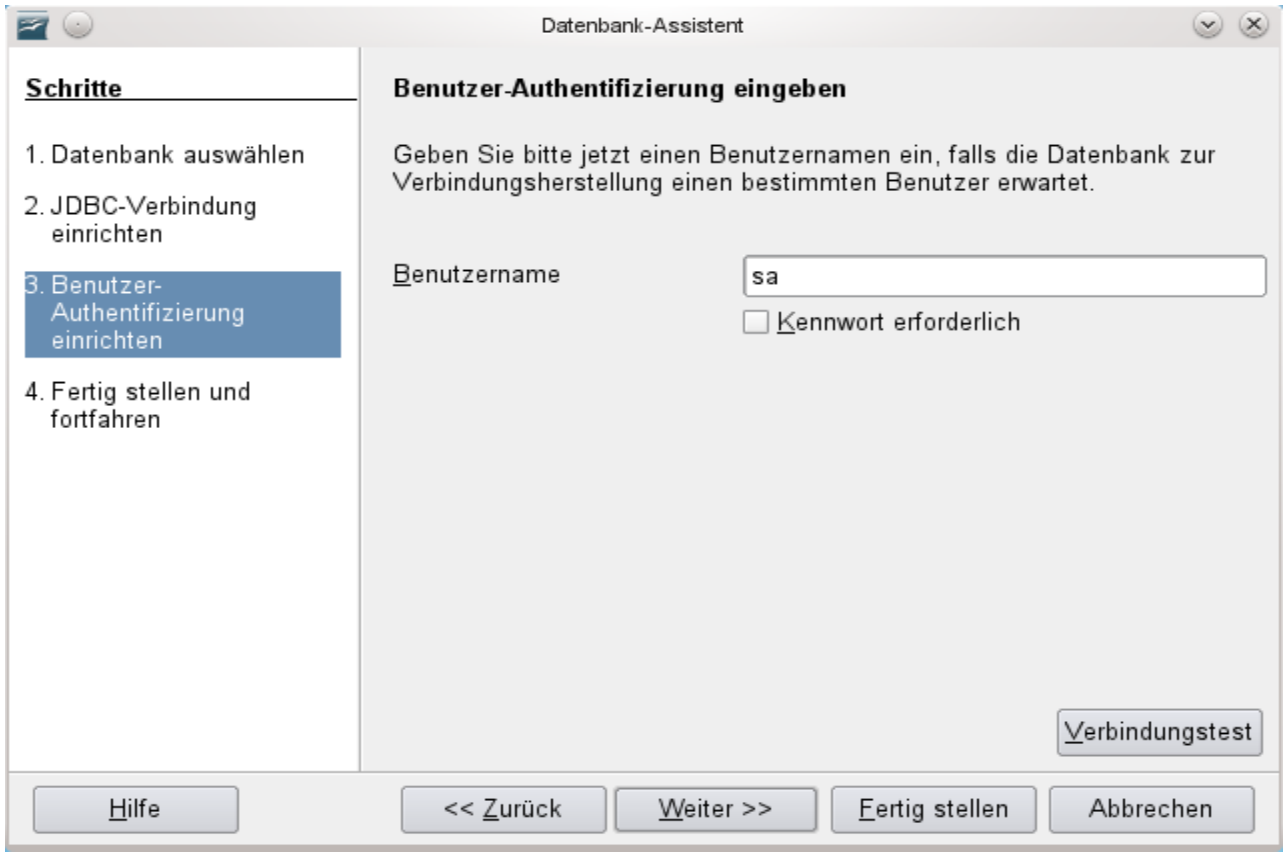
`Datenbankname.properties`

`Datenbankname.script`

`Datenbankname.log`



Weiter geht es mit der Angabe des Standardnutzers, sofern nicht irgendetwas an der HSQLDB-Konfiguration geändert wurde:



Damit ist die Verbindung erstellt und es kann auf die Datenbank zugegriffen werden.

Änderung der Datenbankverbindung zur externen HSQLDB

Die interne HSQL-Datenbank hat den Nachteil, dass die Abspeicherung der Daten innerhalb eines gepackten Archivs erfolgt. Erst mit dem Packen werden alle Daten festgeschrieben. Dies kann leichter zu Datenverlust führen als es bei der Arbeit mit einer externen Datenbank der Fall ist. Im folgenden werden die Schritte gezeigt, die notwendig sind, um den Umstieg einer bestehenden Datenbank vom *.odb-Päckchen zur externen Version in HSQL zu erreichen.

Aus einer Kopie der bestehenden Datenbank wird das Verzeichnis „database“ extrahiert. Der Inhalt wird in das oben beschriebene frei wählbare Verzeichnis kopiert. Dabei sind die enthaltenen Dateien um den Datenbanknamen zu ergänzen:

```
Datenbankname.backup
```

```
Datenbankname.data
```

```
Datenbankname.properties
```

```
Datenbankname.script
```

```
Datenbankname.log
```

Jetzt muss noch die content.xml aus dem *odb-Päckchen extrahiert werden. Hier sind mit einem einfachen Texteditor die folgenden Zeilen zu suchen:

```
<db:connection-data><db:connection-resource
xlink:href="sdbc:embedded:hsqldb"/><db:login db:is-password-
required="false"/></db:connection-data><db:driver-settings/>
```

Diese Zeilen sind mit der Verbindung zur externen Datenbank zu ersetzen, hier der Verbindung zu einer Datenbank mit dem Namen „verein“, die jetzt im Verzeichnis „hsqldb_data“ liegt.

```
<db:connection-data><db:connection-resource
xlink:href="jdbc:hsqldb:file:/home/robby/Dokumente/Datenbanken/hsq
ldb_data/verein;default_schema=true"/><db:login db:user-name="sa"
db:is-password-required="false"/></db:connection-data><db:driver-
settings db:java-driver-class="org.hsqldb.jdbcDriver"/>
```

Falls, wie oben geschrieben, die Grundkonfiguration der HSQLDB nicht angetastet wurde stimmt auch der Nutzername und die nicht erforderliche Passworteinstellung.

Nach Änderung des Codes muss die content.xml wieder in das *.odb-Päckchen eingepackt werden. Das Verzeichnis „database“ ist in dem Päckchen jetzt überflüssig. Die Daten werden in Zukunft durch die externe Datenbank zur Verfügung gestellt.

Änderung der Datenbankverbindung für einen Mehrbenutzerbetrieb

Für die Mehrbenutzerfunktion muss die HSQLDB über einen Server zur Verfügung gestellt werden. Wie die Installation des Servers erfolgt ist je nach Betriebssystem unterschiedlich. Für OpenSuSE war nur ein entsprechendes Paket herunter zu laden und der Server zentral über YAST zu starten (Runlevel-Einstellungen). Nutzer anderer Betriebssysteme und Linux-Varianten finden sicher geeignete Hinweise im Netz.

Im Heimatverzeichnis des Servers, unter SuSE /var/lib/hsqldb, befinden sich unter anderem ein Verzeichnis „data“, in dem die Datenbank abzulegen ist, und eine Datei „server.properties“, die den Zugang zu den (eventuell also auch mehreren) Datenbanken in diesem Verzeichnis regelt.

Die folgenden Zeilen geben den kompletten Inhalt dieser Datei auf meinem Rechner wieder. Es wird darin der Zugang zu 2 Datenbanken geregelt, nämlich der ursprünglichen Standarddatenbank (die als neue Datenbank genutzt werden kann) als auch der Datenbank, die aus der *.odb-Datei extrahiert wurde.

```
# Hsqldb Server cfg file.
# See the Advanced Topics chapter of the Hsqldb User Guide.

server.database.0   file:data/db0
server.dbname.0     firstdb
server.urlid.0      db0-url

server.database.1   file:data/verein
server.dbname.1     verein
server.urlid.1      verein-url

server.silent       true
server.trace        false

server.port         9001
server.no_system_exit         true
```

Die Datenbank 0 wird mit dem Namen „firstdb“ angesprochen, obwohl die einzelnen Dateien in dem Verzeichnis data mit „db0“ beginnen. Meine eigene Datenbank habe ich als Datenbank 1 hinzugefügt. Hier sind Datenbankname und Dateibeginn identisch.

Die beiden Datenbanken werden mit folgenden Zugängen angesprochen:

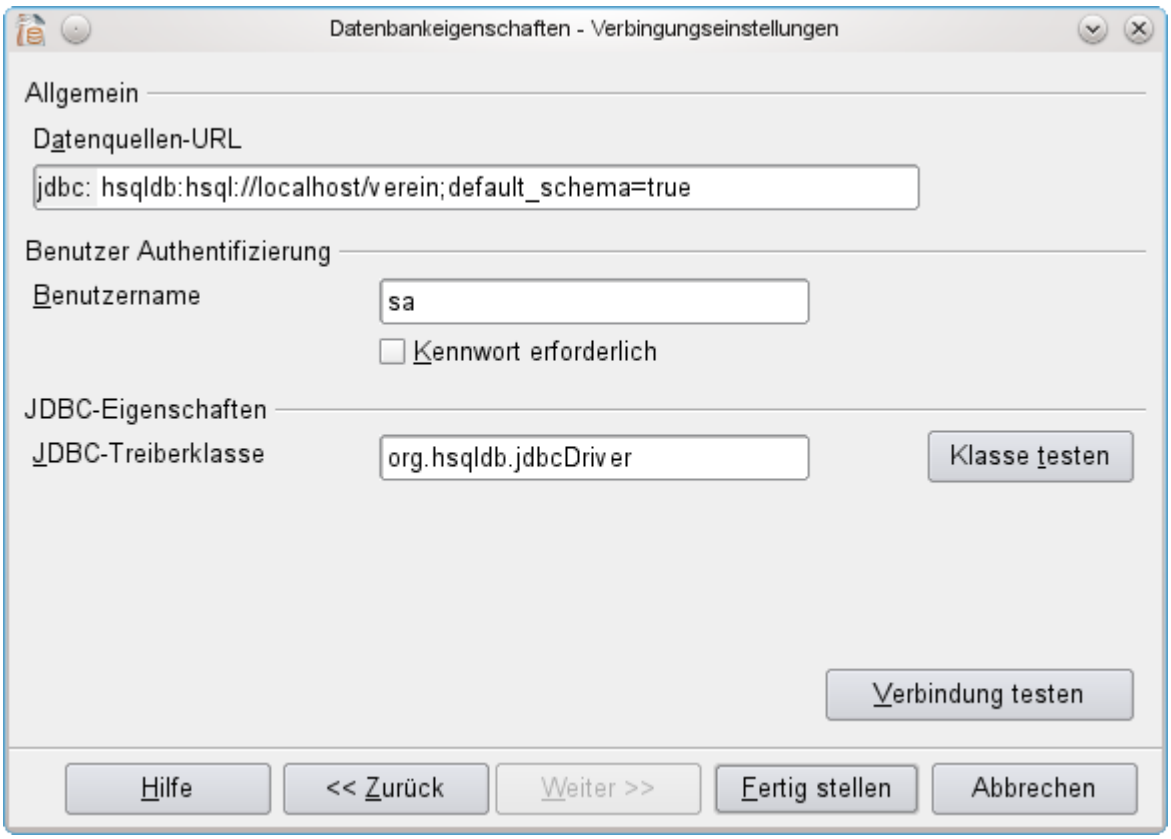
```
jdbc:hsqldb:hsql://localhost/firstdb;default_schema=true
username sa
password

jdbc:hsqldb:hsql://localhost/verein;default_schema=true
username sa
password
```

Die URL wurde hier bereits jeweils um den für den Schreibzugang über die grafische Benutzeroberfläche von OpenOffice erforderlichen Zusatz `;default_schema=true` ergänzt.

Wenn tatsächlich im Serverbetrieb gearbeitet werden soll ist natürlich aus Sicherheitsgründen zu überlegen, ob die Datenbank nicht mit einem Passwort geschützt werden soll.

Nun erfolgt die Serververbindung über OpenOffice:



Mit diesen Zugangsdaten wird auf den Server des eigenen Rechners zugegriffen. Im Netzwerk mit anderen Rechnern müsste dann entweder über Rechnernamen oder die IP-Adresse auf den Server, der ja auf meinem Rechner läuft, zugegriffen werden.

Beispiel: Mein Rechner hat die IP 192.168.0.20 und ist im Netz bekannt mit dem Namen lin_serv. Jetzt ist an anderen Rechnern für die Verbindung zur Datenbank einzugeben:

```
jdbc:hsqldb:hsql://192.168.0.20/verein;default_schema=true
```

bzw.:

```
jdbc:hsqldb:hsql://lin_serv/verein;default_schema=true
```

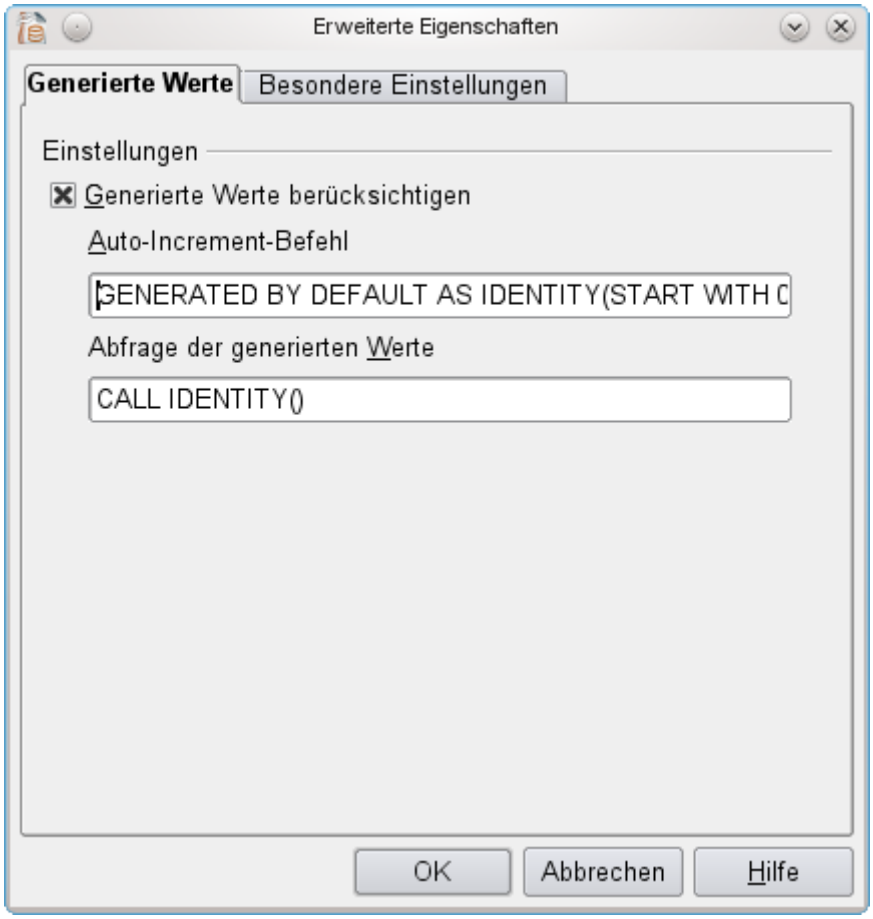
Die Datenbank ist nun angebunden und kann beschrieben werden. Schnell taucht allerdings ein zusätzliches Problem auf. Die vorher automatisch generierten Werte werden plötzlich nicht mehr hochgeschrieben. Hier fehlt es noch an einer zusätzlichen Einstellung.

Autoinkrementwerte mit der externen HSQLDB

Die Nutzung der Auto-Werte gelingt in den momentanen Fassungen von OpenOffice (bis 3.3) leider nur, wenn die Tabelle bereits existiert. Hierzu sind die folgenden Einträge unter

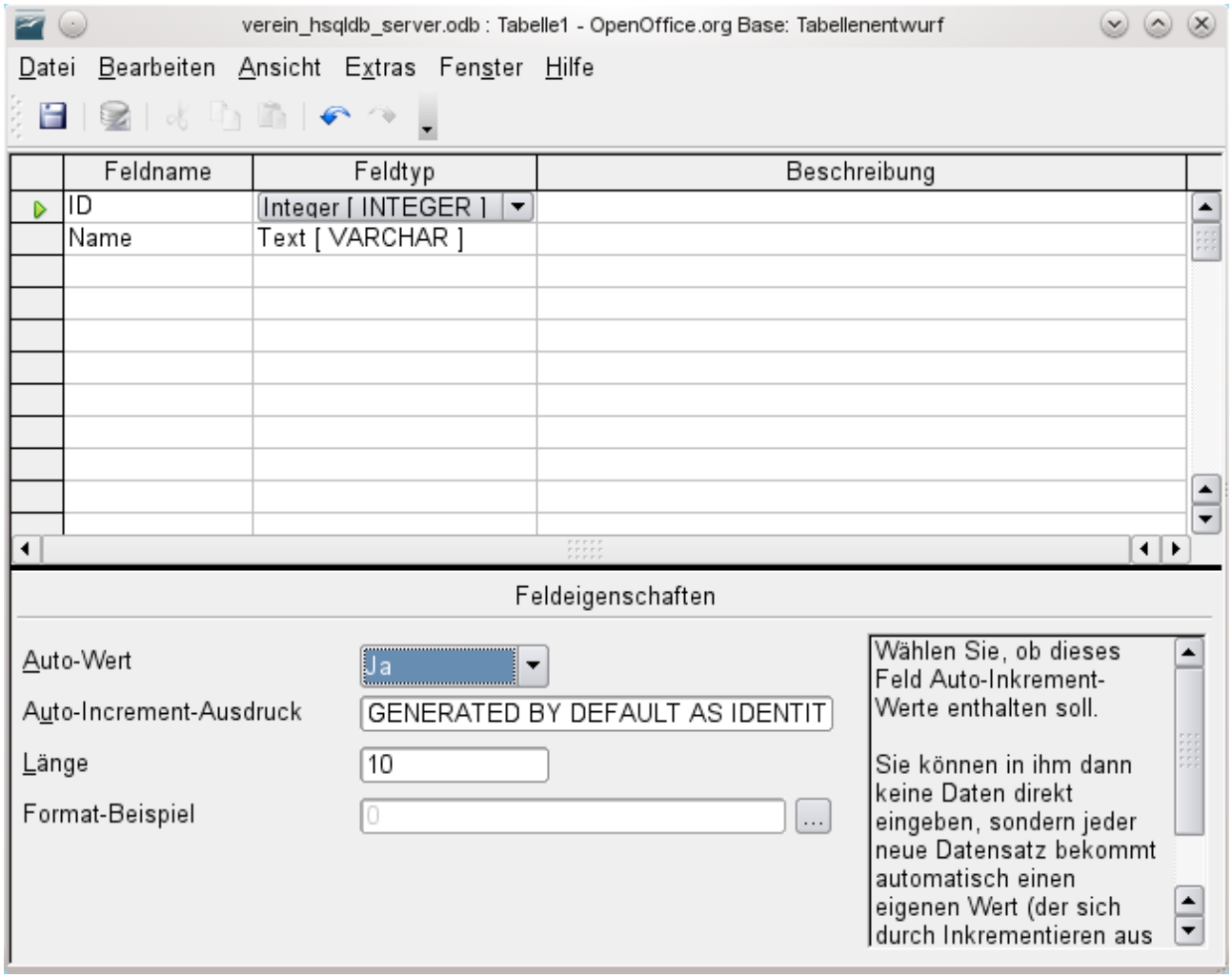
Bearbeiten – Datenbank – Erweiterte Einstellungen

erforderlich:

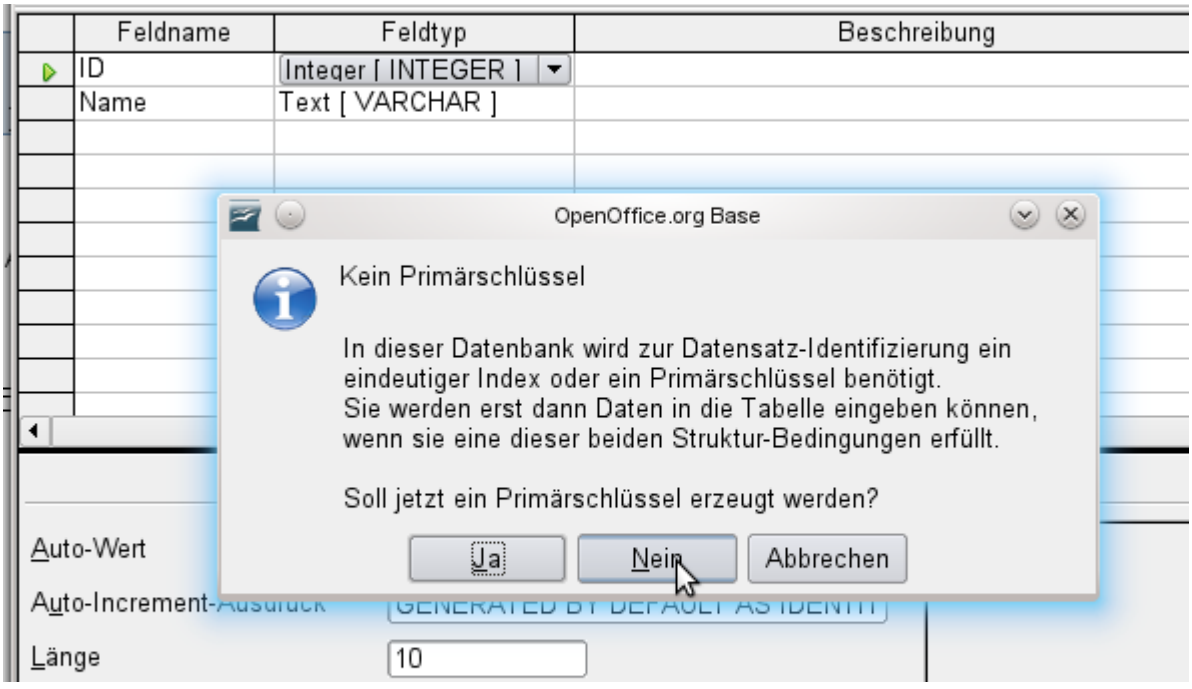


Mit dem Zusatz `GENERATED BY DEFAULT AS IDENTITY(START WITH 0)` soll die Funktion des automatisch hochzählenden Wertes für den Primärschlüssel erstellt werden. Die GUI von OpenOffice übernimmt zwar diesen Befehl, schreibt davor aber leider die Anweisung `„NOT NULL“` , so dass die Reihenfolge der Befehlsfolge für die HSQLDB nicht lesbar ist. Hierbei ist zu berücksichtigen, dass die HSQLDB mit dem obigen Befehl ja bereits mitgeteilt bekommt, dass das entsprechende Feld den Primärschlüssel enthält.

Bei der Erstellung der Tabelle muss also folgendermaßen vorgegangen werden:



Das Feld mit der Bezeichnung „ID“ hat den Typ „Integer“ und bekommt zusätzlich den Auto-Wert zugeschrieben. In der Übersicht erscheint jetzt der einzufügende Ausdruck dazu. Da dieser Ausdruck bereits die Definition des Primärschlüssels enthält wird nicht noch einmal in der GUI die Eigenschaft „Primärschlüssel“ zugewiesen.



Auch auf diese Frage muss unbedingt mit „Nein“ geantwortet werden. Der Primärschlüssel wird dann trotzdem richtig geschrieben und die Tabelle hat ihren Auto-Wert.

Mit dem Auslesen des letzten Wertes und dem Hochlesen zum nächsten Wert hingegen klappt es über den Befehl `CALL IDENTITY().` Dies trifft dann z.B. auf die von mir verfolgte Lösung zu, die Datenbank zuerst einmal als *.odb-Päckchen zu erstellen, gründlich zu testen und danach dann die Datenbanktabellen einfach auszulagern.

Sämtliche Abfragen, Formulare und Berichte lassen sich so weiter nutzen, da die Datenbank für die *.odb-Datei weiter auf die gleiche Weise angesprochen wird und eventuell spezifische SQL-Befehle mit der externen HSQLDB weiter gelesen werden können.