HAUFE.
Ausgabe 02/2020

ACCESS

IM UNTERNEHMEN

RECHNUNGEN MIT ZUGFERD

Einstieg in die digitale Verarbeitung von Rechnungen mit Access (ab S. 57)

In diesem Heft:

Server und Pferde

In dieser Ausgabe schauen wir uns gleich zwei Themen schwerpunktmäßig an. Im ersten Schwerpunkt geht es um das Auslagern der Tabellen einer Access-Datenbank in eine MySQL-Datenbank auf einem virtuellen Server im Internet mit dem Ziel, von überall auf die Daten zugreifen zu können. Der zweite betrifft die Rechnungserstellung: Der ZUGFeRD-Standard könnte dafür sorgen, dass Rechnungen demnächst nicht nur von Menschen, sondern auch von Computern gelesen werden können.

»Wie bringe ich meine Access-Datenbank ins Internet?« Diese Frage ist wohl eine der populärsten unter Access-Entwicklern. Access ist zweifelsohne ein sehr effizientes Werkzeug, um schnell Datenbankanwendungen zu programmieren. Wenn man die Datenbank im Netzwerk verfügbar machen will, wird es schon aufwendiger. Wenn man dann noch eine performante und sichere Lösung braucht, kommt man um den Einsatz eines Datenbank-Management-Systems wie SQL Server nicht herum.

Richtig spannend wird es dann, wenn man die Daten nicht nur über das Firmennetzwerk, sondern von verschiedenen Standorten aus abrufen will. Dann bietet sich die Möglichkeit, einen Internetserver oder auch nur einen virtuellen Server zu mieten, der mit einem SQL-Datenbankserver ausgestattet ist. Eine sehr günstige Möglichkeit dazu beschreiben wir im Beitrag **MySQL im Web: Plesk mit Contabo** ab Seite 11. Hier können Sie für rund 10 EUR im Monat einen über das Internet erreichbaren Server mieten, auf dem Sie MySQL-Datenbanken laufen lassen können. Dieser Beitrag beschreibt die Grundlagen der Einrichtung mit dem Administrationstool **Plesk**.

Ein weiteres Administrationstool heißt **phpMyAdmin**. Während Sie mit Plesk eher allgemeine Belange wie Dateien, Benutzer, E-Mail-Konten, Anwendungen und Datenbanken verwalten, pflegen Sie mit phpMyAdmin die Inhalte von Datenbanken. Mehr zu diesem Thema unter **MySQL im Web: Verwaltung mit phpMyAdmin** ab S. 25.

Damit das alles funktioniert, müssen die Daten aus der Access-Datenbanken in die MySQL-Datenbank übertragen werden. Wie die Migration mit MySQL Workbench gelingt, lesen Sie im Beitrag **Access-Datenbank zu MySQL migrieren** ab Seite 33.

Und damit Sie dann auch noch von Access auf die Tabellen der MySQL-Datenbank zugreifen können, stellen wir in einer neuen Beitragsreihe mit dem ersten Teil **MySQL im Web: Tools für das Access-Frontend, Teil 1** ab Seite 46 Tools und Techniken für den Datenaustausch vor.

Im anderen Schwerpunkt geht es nur auf den ersten Blick um Pferde: Tatsächlich ist ZUGFeRD nämlich ein Akronym für ein System, mit dem die Daten einer Rechnung im computerlesbaren XML-Format an die eigentliche Rechnung im PDF-Format angehängt werden kann.

Im Beitrag **Rechnungen mit ZUGFeRD 1.0, Teil 1: .NET** zeigen wir Ihnen ab Seite 57, wie Sie eine Bibliothek zum Ausstatten der Rechnung mit den Daten im XML-Format in eine .NET-DLL packen, die Sie dann von Access aus nutzen können. Das ist notwendig, da es keine passende Bibliothek gibt, die wir unter Access nutzen könnten.

Der Beitrag **Rechnungen mit ZUGFeRD 1.0, Teil 2: Access-DLL** zeigt dann ab Seite 69, wie Sie Sie diese DLL unter Access einsetzen. Damit decken wir die Version 1.0 des ZUGFeRD-Standards ab – um modernere Versionen kümmern wir uns in naher Zukunft!

Und nun: Viel Spaß beim Lesen!

Ihr André Minhorst

Dateiverknüpfung per Drag and Drop hinzufügen

Es gibt verschiedene Möglichkeiten, Dateien mit Access zu verwalten. Je nach Speicherplatz speichert man diese in einem Anlage-Feld oder in einem externen Ordner und verweist dann über die Angabe des Pfades auf diese Datei. Es gibt zwar auch noch OLE-Felder, aber Anlagefelder sind intuitiver zu nutzen, da man diesen auch ohne Code Dateien hinzufügen kann. Aufgrund der begrenzten Größe von Access-Dateien mit zwei Gigabyte ist es oft sinnvoller, nur die Dateipfade zu speichern. Spannend wird es beim Hinzufügen der Dateien selbst: Dies geschieht in der Regel durch Öffnen eines Dateiauswahl-Dialogs. In vielen Fällen ist das Verzeichnis mit den Dateien aber bereits im Windows Explorer geöffnet und man könnte diese schneller per Drag and Drop hinzufügen. Wie das gelingt, zeigen wir im vorliegenden Beitrag.

Drag and Drop ins Anlagefeld?

Grundsätzlich wäre es praktisch, wenn man Dateien direkt in ein Textfeld ziehen könnte und der Pfad zu dieser Datei dann in das Textfeld eingetragen wird. Das gelingt allerdings nicht mit Bordmitteln, denn die eingebauten Steuerelemente erlauben kein Drag and Drop. Ein Mangel, mit dem viele Access-Entwickler hadern.

Die einzigen Steuerelemente, welche die Behandlung von Drag and Drop-Ereignissen erlauben, sind in der Bibliothek **MSCOMCTL.ocx** enthalten. Dabei handelt es sich um Steuerelemente wie das **ListView**-, das **TreeView**- oder das **ImageCombo**-Steuerelement.

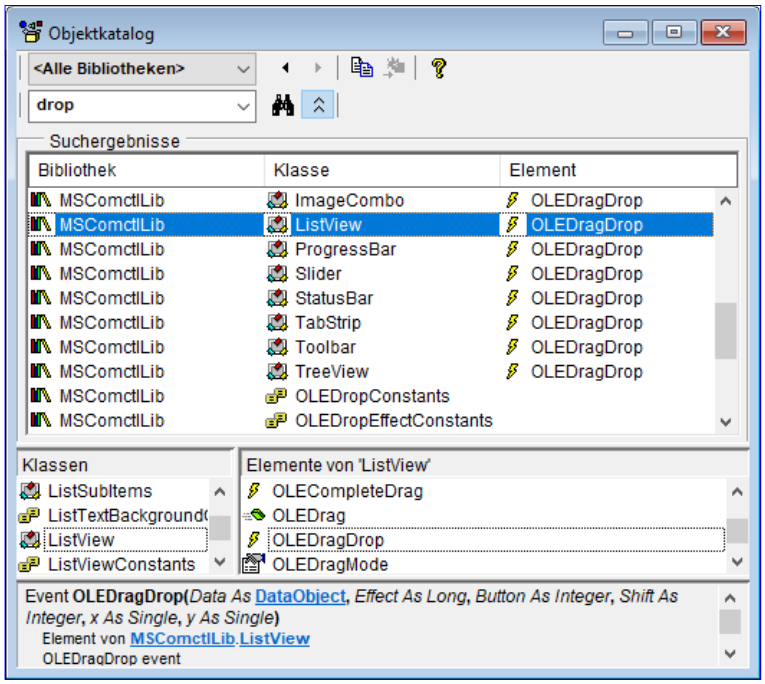

Bild 1: Steuerelemente mit Drag and Drop-Ereignissen

Das erfahren Sie, wenn Sie eines dieser Steuerelemente zu einem Formular einer Datenbankdatei hinzufügen und dann im VBA-Editor den Objektkatalog öffnen, wo Sie wie in Bild 1 nach dem Suchbegriff **Drop** suchen.

Wir können also nicht direkt per Drag and Drop Dateien direkt in ein Textfeld ziehen. Das ließe sich mit API-Programmierung erreichen, wovon wir aber in diesem Beitrag absehen. Stattdessen werden wir dazu ein **ListView**-Steuerelement verwenden, das wir an geeigneter Stelle in dem Formular unterbringen, welches wir mit der Drag and Drop-Funktion ausstatten wollen.

Wenn der Benutzer dann eine Datei auf dieses Steuerelement zieht, soll ein entsprechendes Ereignis ausgelöst werden, das den Dateipfad erkennt und diesen in das gewünschte Feld einträgt.

Alternative: Datei zusätzlich in bestimmtes Verzeichnis kopieren

Gegebenenfalls sollen die Dateien, deren Pfad in der Tabelle gespeichert werden sollen, zuvor in ein spezielles Verzeichnis kopiert oder verschoben werden. Das kann beispielsweise ein Verzeichnis sein, das sich im gleichen Ordner wie die Datenbankdatei befindet. Diese Funktion wollen wir alternativ vorsehen.

Tabelle zum Speichern der Dateipfade

Die Tabelle, in der wir die Pfade zu den per Drag and Drop einzufügenden Dateien speichern, sieht in der Entwurfsansicht wie in Bild 2 aus. Neben dem Primärschlüsselfeld namens **ID** soll die Tabelle nur noch das Feld **Dateipfad** enthalten.

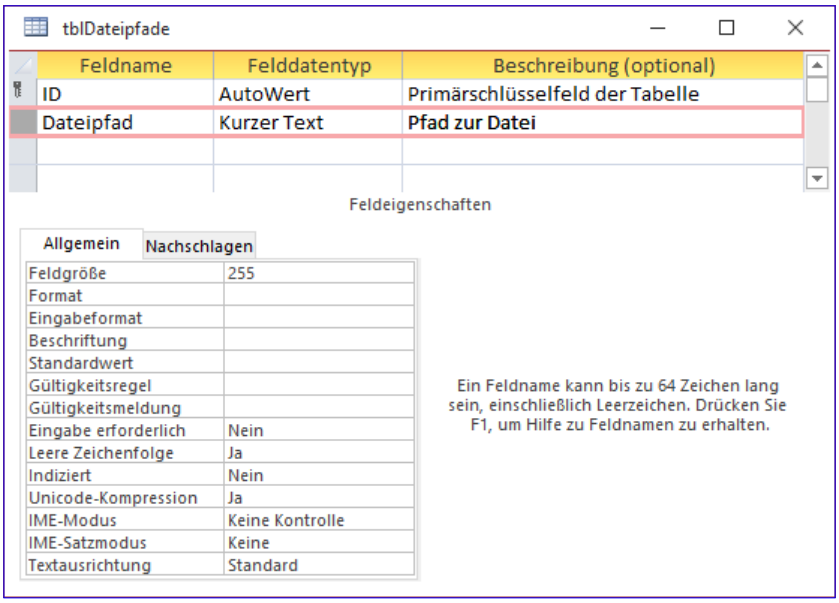

Bild 2: Tabelle zum Speichern der Dateipfade

Formular mit der Drag and Drop-Funktion

Das Formular, dem wir die Drag and Drop-Funktion hinzufügen wollen, verwendet die Tabelle **tblDateipfade** als Datensatzquelle. Wir ziehen beide Felder dieser Tabelle in den Formularentwurf. Außerdem fügen wir über den Dialog **ActiveX-Steuerelemente einfügen** ein Steuerelement des Typs **Microsoft ListView Control, version 6.0** zum Formular hinzu. Dieses sieht anschließend wie in Bild 3 aus.

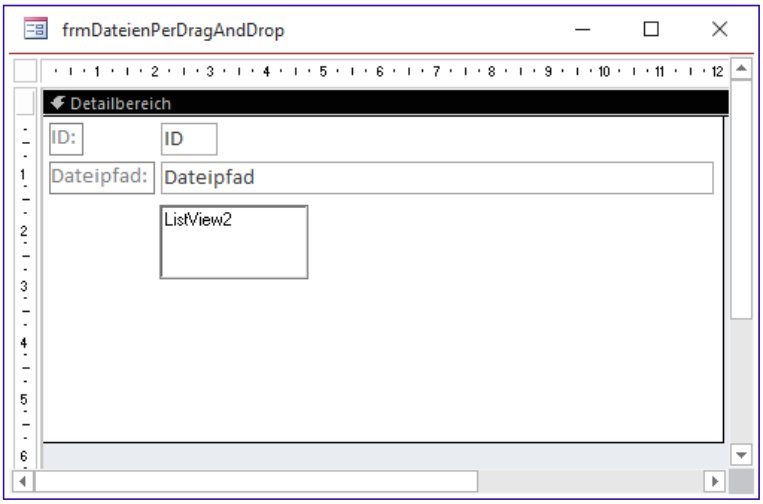

Bild 3: Formular zum Hinzufügen von Dateipfaden

Darstellung des Drag and Drop-Ziels

Einfach nur ein **ListView**-Steuerelement als Drag and Drop-Ziel zu verwenden, ist die einfachste Lösung. Aber ergonomischer wäre es, wenn das **ListView**-Element sich direkt als Drag and Drop-Ziel zeigt – beispielsweise, indem wir einen passenden Text oder ein Icon darin abbilden.

Das soll später so aussehen wie in Bild 4. Im Ruhezustand soll das Ziel den Text **Ziehen Sie die Datei in diesen Bereich** anzeigen, beim Ziehen einer Datei auf das Ziel den Text **Lassen Sie die Maustaste los, um die Datei hinzuzufügen**.

ImageList vorbereiten

Damit das geschieht, benötigen wir zunächst einmal zwei verschiedene Bilder, die wir je nach Zustand im **ListView** anzeigen. Außerdem müssen wir die Größe

der Bilddatei festlegen. Das ist der erste Schritt:

Fügen Sie über den Dialog **ActiveX-Steuerelemente einfügen** ein Element des Typs **Microsoft ImageList, version 6.0** hinzu, das Sie gleich in **ctlImageList** umbenennen. Das Steuerelement erscheint dann wie in Bild 5 links oben im Formular. Wenn Sie in die Formularansicht wechseln, wird das Steuerelement jedoch nicht sichtbar sein.

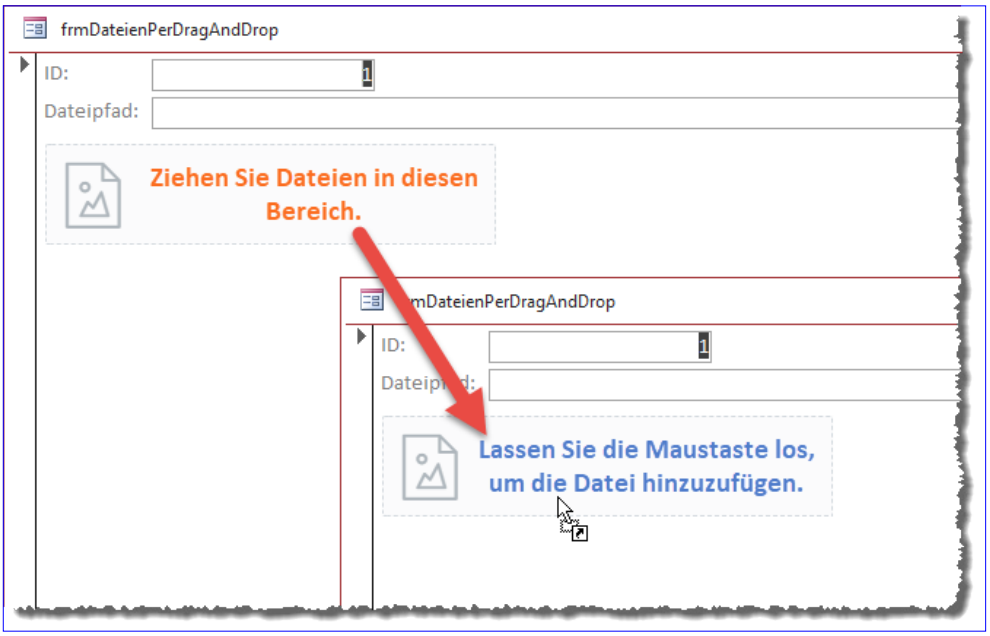

Bild 4: Ändern des Drag and Drop-Ziels beim Ziehen mit der Maus

Nach einem Doppelklick auf das Steuerelement in der Entwurfsansicht erscheint das Eigenschaftsfenster aus Bild 6. Hier stellen Sie für die Option **Custom** die Größe der zu verwendenden Bilddatei ein, hier **71 x 311**.

Auf der zweiten Registerseite fügen Sie mit der Schaltfläche **Insert Picture...** die beiden Bilder hinzu, die im Ruhezustand und beim Ziehen

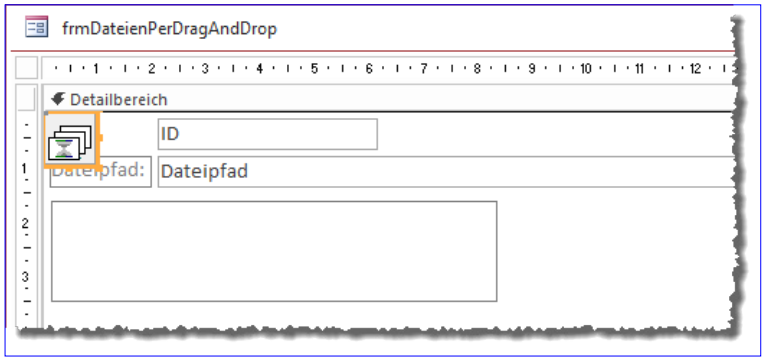

Bild 5: Das neu hinzugefügte **ImageList**-Steuerelement

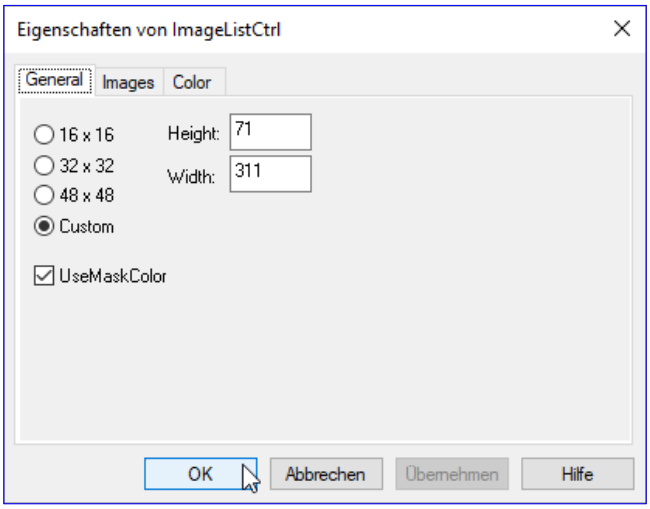

Bild 6: Einstellen der Größe der Icons im **ImageList**-Steuerelement

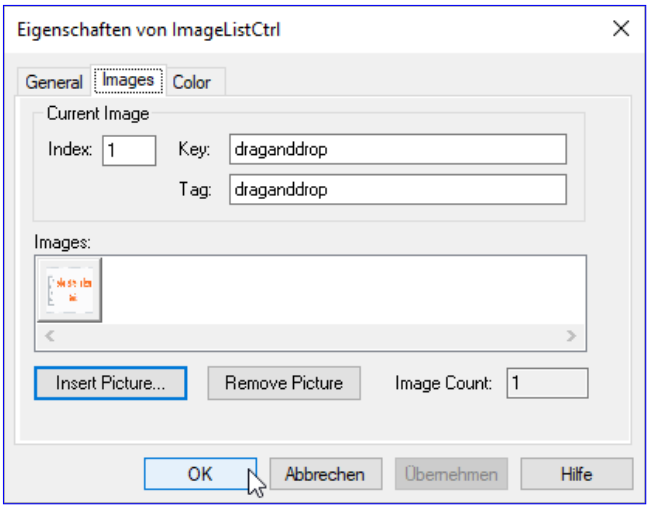

Bild 7: Einfügen des ersten Bildes in das **ImageList**-Steuerelement

eines Elements über dem **ListView**-Steuerelement angezeigt werden sollen. Die Index-Werte für die beiden Bilder lauten **1** und **2**. **Key** und **Tag** können Sie ausfüllen, aber die Eigenschaften werden nicht benötigt (siehe Bild 7).

ListView programmieren

Nun müssen wir das **ListView**-Steuerelement erst einmal so programmieren, dass es das erste Bild aus dem **ImageList**-Steuerelement anzeigt. Dazu deklarieren wir im Klassenmodul des Formulars zunächst einmal zwei Variablen. Die erste nimmt einen Verweis auf das Steuerelement **lvwDragAndDrop** auf und erhält zusätzlich das Schlüsselwort **WithEvents**, damit wir im gleichen Klassenmodul Ereignisse für dieses Steuerelement deklarieren können:

```
Dim WithEvents objDragAndDrop As MSComctlLib.ListView
```

Die zweite soll einen Verweis auf das erste im **ListView**-Steuerelement enthaltene Element enthalten:

```
Dim objListItem As MSComctlLib.ListItem
```

Den Hauptteil der Arbeit erledigen wir in der Prozedur **Form_Load**, die durch das Ereignis **Beim Laden** des Formulars ausgelöst wird. Dieses deklariert eine weitere Variable namens **objImageList**, mit der wir auf das **ImageList**-Steuerelement **ctlImageList** verweisen wollen:

```
Private Sub Form_Load()
    Dim objImageList As MSComctlLib.ImageList
```

Dann ordnen wir der modulweit deklarierten Variable **objDragAndDrop** einen Verweis auf die Eigenschaft **Object** des Steuerelements **lvwDragAndDrop** zu und schließlich der soeben deklarierten Variable **objImageList** einen Verweis auf die Eigenschaft **object** des **ImageList**-Steuerelements.

```
    Set objDragAndDrop = Me!lvwDragAndDrop.Object
    Set objImageList = Me!ctlImageList.Object
```

Für das mit **objDragAndDrop** referenzierte **ListView**-Steuerelement stellen wir nun einige Eigenschaften ein. Die Eigenschaft **SmallIcons** füllen wir mit einem Verweis auf das **ImageList**-Steuerelement:

```
    With objDragAndDrop
        Set .SmallIcons = objImageList
```

Für das Erscheinungsbild stellen wir unter **Appearance** **ccFlat** ein (im Gegensatz zu **cc3d**). **BorderStyle** erhält den Wert **ccNone**. Damit stellen wir sicher, dass man das **ListView**-Steuerelement nicht an der Darstellung oder am Rahmen als solches erkennen kann. Für die Eigenschaft **View** stellen wir **lvwList** ein, da so das Bild angezeigt wird, das wir gleich zuweisen:

```
        .Appearance = ccFlat
        .BorderStyle = ccNone
        .View = lvwList
```

Mit dem Wert **ccOLEDragAutomatic** für die Eigenschaft **OLEDragMode** und dem Wert **ccOLEDropManual** für die Eigenschaft **OLEDropMode** legen wir fest, dass wir mit Ereignisprozeduren auf Drag and Drop-Ereignisse reagieren können:

```
        .OLEDragMode = ccOLEDragAutomatic
        .OLEDropMode = ccOLEDropManual
```

Danach leeren wir die Auflistung **ListItems** des **ListView**-Steuerelements mit der **Clear**-Methode, damit eventuell bereits vorhandene Elemente gelöscht werden:

```
        objDragAndDrop.ListItems.Clear
```

Schließlich fügen wir ein neues Listenelement hinzu und nutzen dazu die **Add**-Methode von **ListItems**:

```
        Set objListItem = _
            objDragAndDrop.ListItems.Add(, "a", , , 1)
    End With
```

MySQL im Web: Plesk mit Contabo

Mich erreichen immer wieder Anfragen, wie man die Daten seiner Access-Datenbank ins Internet bringt. Dazu gibt es mehrere Möglichkeiten, die alle auf einen Punkt hinauslaufen: eine Datenbank auf Basis eines RDBMS wie MySQL, SQL Server oder eines anderen Systems, das auf einem Webserver liegt, nimmt die Daten auf und man greift von Access aus über ODBC auf diese Daten zu. Während Microsoft mit seinem Azure-Dienst eine Möglichkeit anbietet, SQL Server-Datenbanken über das Internet verfügbar zu machen, gibt es genügend weitere Internetanbieter, die virtuelle Webserver für sie hosten und damit auch einen MySQL-Server bereitstellen. Und man muss ja nicht gleich eine Webseite bauen, sondern kann auch nur die MySQL-Datenbank nutzen. Wie das gelingt, zeigen wir am Beispiel eines recht günstigen Anbieters, nämlich Contabo. Im ersten Teil der Beitragsreihe zeigen wir, wie Sie einen Virtual Private Server mieten und unter Plesk Datenbanken verwalten.

Contabo ist ein deutscher Anbieter, der für kleinere Pakete mit der Bezeichnung VPS einen recht günstigen Preis bietet. VPS bedeutet dabei Virtual Private Server und meint, dass Sie zwar keinen eigenen Server haben, aber einen virtuellen Server auf einem Server verwenden können.

Diesen müssen Sie sich mit anderen Nutzern teilen, es sind also mehrere virtuelle Server auf einem Server untergebracht. Das ist aber für die meisten Projekte ausreichend, und die Performance lässt sich, wenn man genügend Budget hat, beliebig nach oben skalieren. Den im Beispiel genutzten virtuellen Server richten Sie ein, indem Sie unter dem folgenden Link den Bereich **VPS** auswählen:

contabo.de

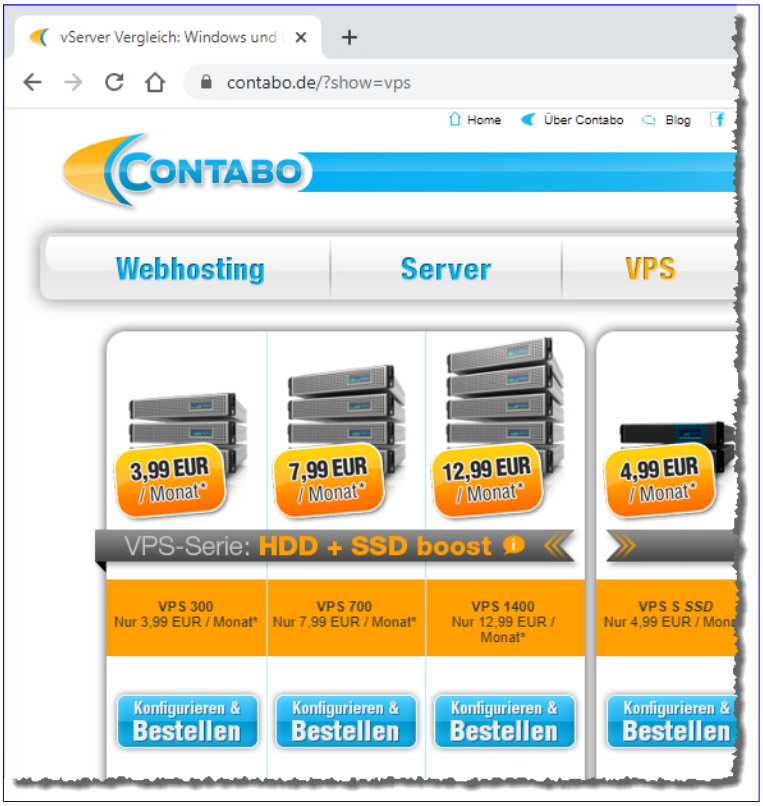

Bild 1: VPS-Angebote von Contabo

Hier finden Sie beispielsweise das Paket **VPS 300** für 3,99 EUR/Monat, das zum Ausprobieren und gegebenenfalls noch für viel mehr ausreicht. Wenn Sie den Beispielen in diesem Beitrag folgen wollen, sollten Sie allerdings noch ein Administrationspanel hinzufügen, das nochmal 6,49 EUR/Monat zusätzlich kostet. Damit erhalten Sie eine extrem komfortable Möglichkeit, alle Aspekte des Webspace und vor allem der MySQL-

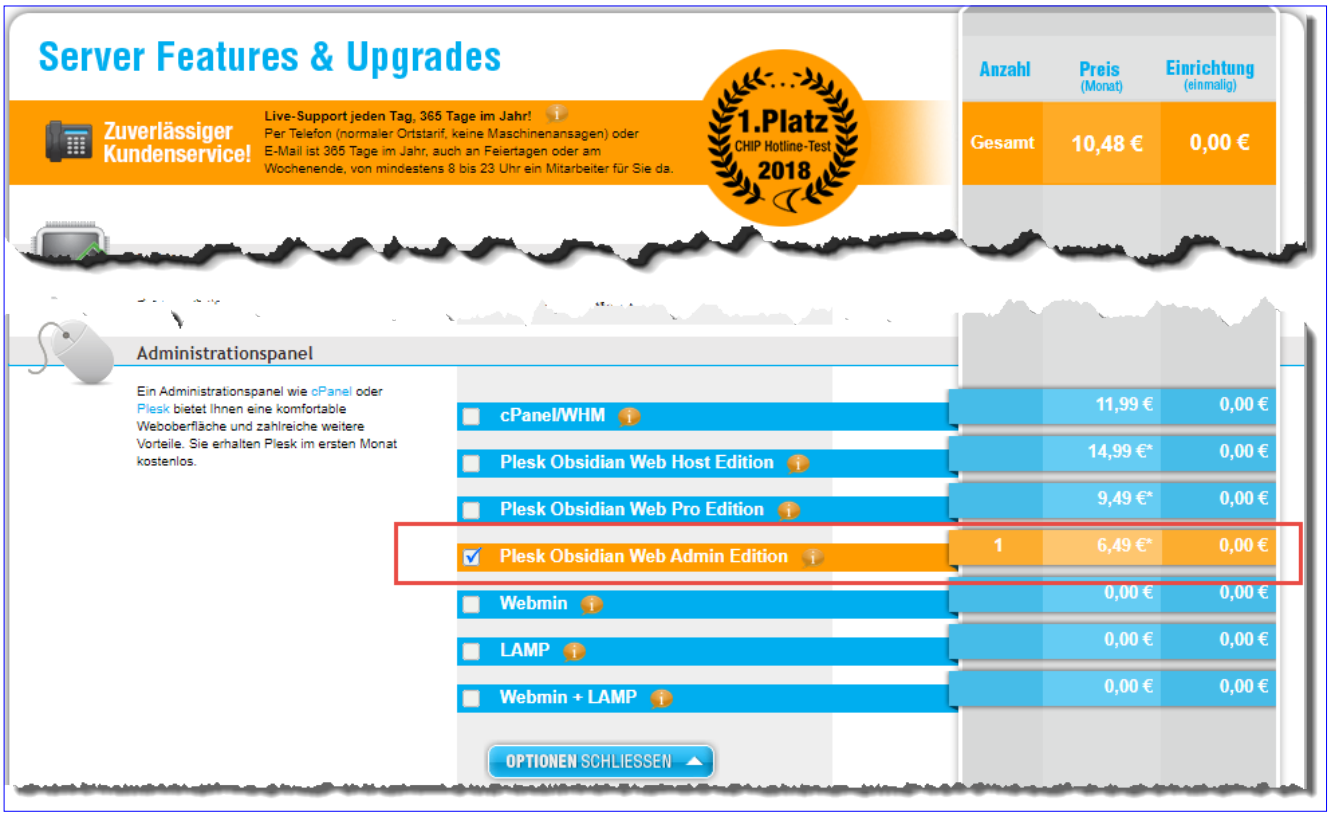

Bild 2: Hinzufügen eines Administrationspanels

Datenbank zu verwalten. Dieses wählen Sie aus, nachdem Sie wie in Bild 1 auf die Schaltfläche **Konfigurieren & Bestellen** geklickt haben.

Administrationspanel hinzufügen

Das Administrationspanel fügen Sie auf der folgenden Seite hinzu, wo Sie nach unten scrollen und dann den Eintrag aus Bild 2 hinzufügen. Schließlich wählen Sie noch ein passendes Betriebssystem aus, wobei wir mit **Cent OS 7** gut gefahren sind.

Sie können auch noch weitere Optionen hinzubuchen wie beispielsweise Backup-Speicherplatz für die MySQL-Datenbanken. Außerdem legen Sie hier noch fest, wie lange der Vertrag laufen soll – und erhalten damit eine mit kürzerer Vertragsdauer höhere Einrichtungsgebühr.

Anschließend klicken Sie unten auf **Konfiguration bestellen**, um die Bestellung fortzusetzen. Auf der nächsten Seite geben Sie dann noch die Kundendaten ein und legen die Zahlungsdaten fest. Wenn Sie möchten, geben Sie auch noch eine Domain an, die Sie für 11,88 EUR pro Jahr dazu buchen – diese können Sie dann beispielsweise nutzen, um die Daten Ihrer MySQL-Datenbank über eine PHP-Seite anzuzeigen.

Nach dem Bestätigen verschiedener Optionen wie AGB et cetera schließen Sie dann die Bestellung ab. Danach erhalten Sie eine E-Mail mit einem Link, über den Sie den Bestellstatus prüfen können.

Es dauert dann eine Weile, bis Sie eine weitere E-Mail erhalten, mit der die Einrichtung bestätigt wird – nach Angabe von Contabo bis zu 24 Stunden.

Bei der Bestellung, die wir für die Beispiele dieses Beitrags durchgeführt haben, hat Contabo übrigens nur eine knappe Stunde benötigt.

Diese E-Mail enthält dann wichtige Daten, von denen die folgenden für uns interessant sind und die zum Administrations-Panel gehören:

- **IP-Adresse**
- **Benutzername** (meist **root**)
- **Passwort**
- **Adresse** (meist **https://vmdxxxxx.contaboserver.net:8443/**, wobei **xxxxx** ein Platzhalter für eine Zahl ist). Hierüber können Sie die Administrationsoberfläche im Webbrowser öffnen.

Weiter unten in der Mail finden Sie außerdem den Aktivierungscode für die Plesk-Lizenz, also für das Administrations-Panel.

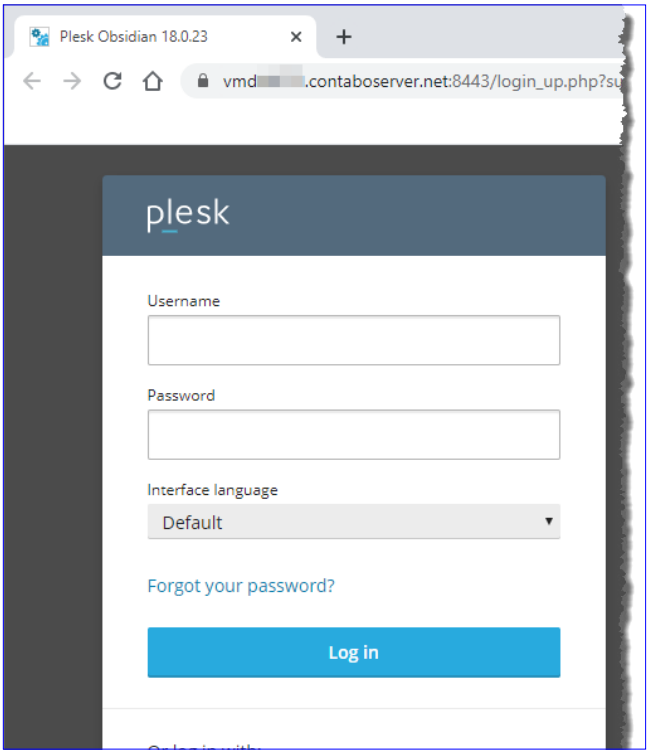

Bild 3: Login für das Administrations-Panel

Plesk aufrufen

Um zur Administrationsoberfläche zu gelangen, klicken Sie einfach auf den oben beschriebenen Link in der Bestätigungsmail. Danach erscheint das Login-Fenster von Plesk, wo Sie die Zugangsdaten aus der E-Mail für das Administrations-Panel eingeben können (siehe Bild 3).

Danach landen Sie auf der Seite aus Bild 4. Hier geben Sie oben Ihre Kontaktdaten ein, also Ihren Namen und Ihre E-Mail-Adresse. Darunter legen Sie das Passwort fest, unter dem Sie sich künftig an der Plesk-Benutzeroberfläche anmelden wollen. Als Benutzername verwenden Sie dann, was gern überlesen wird, den Namen **admin**.

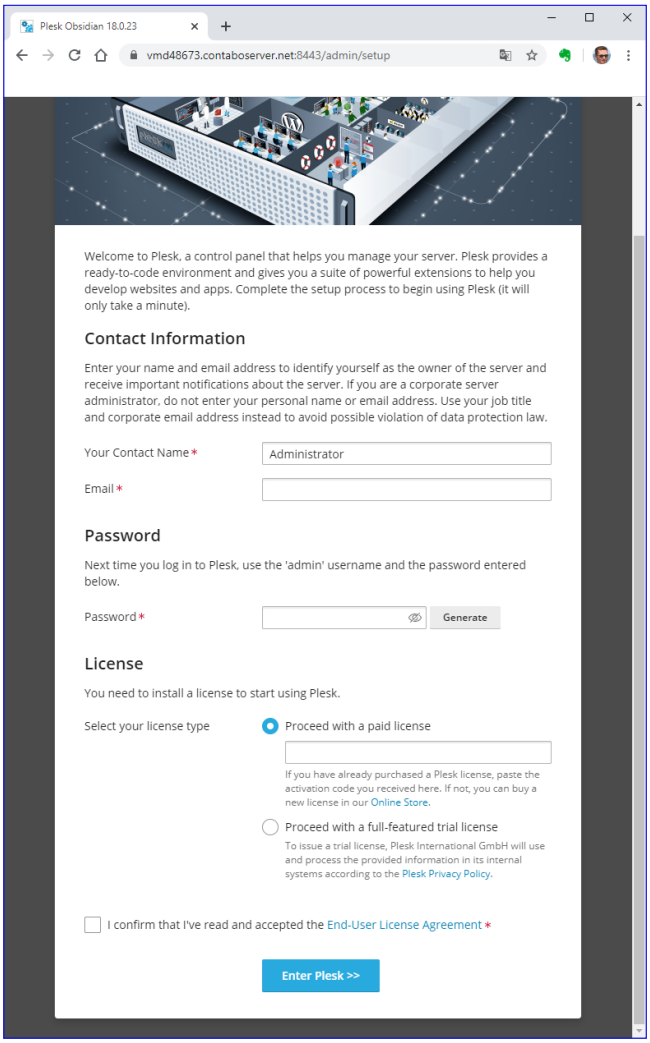

Bild 4: Eingabe einiger wichtiger Daten

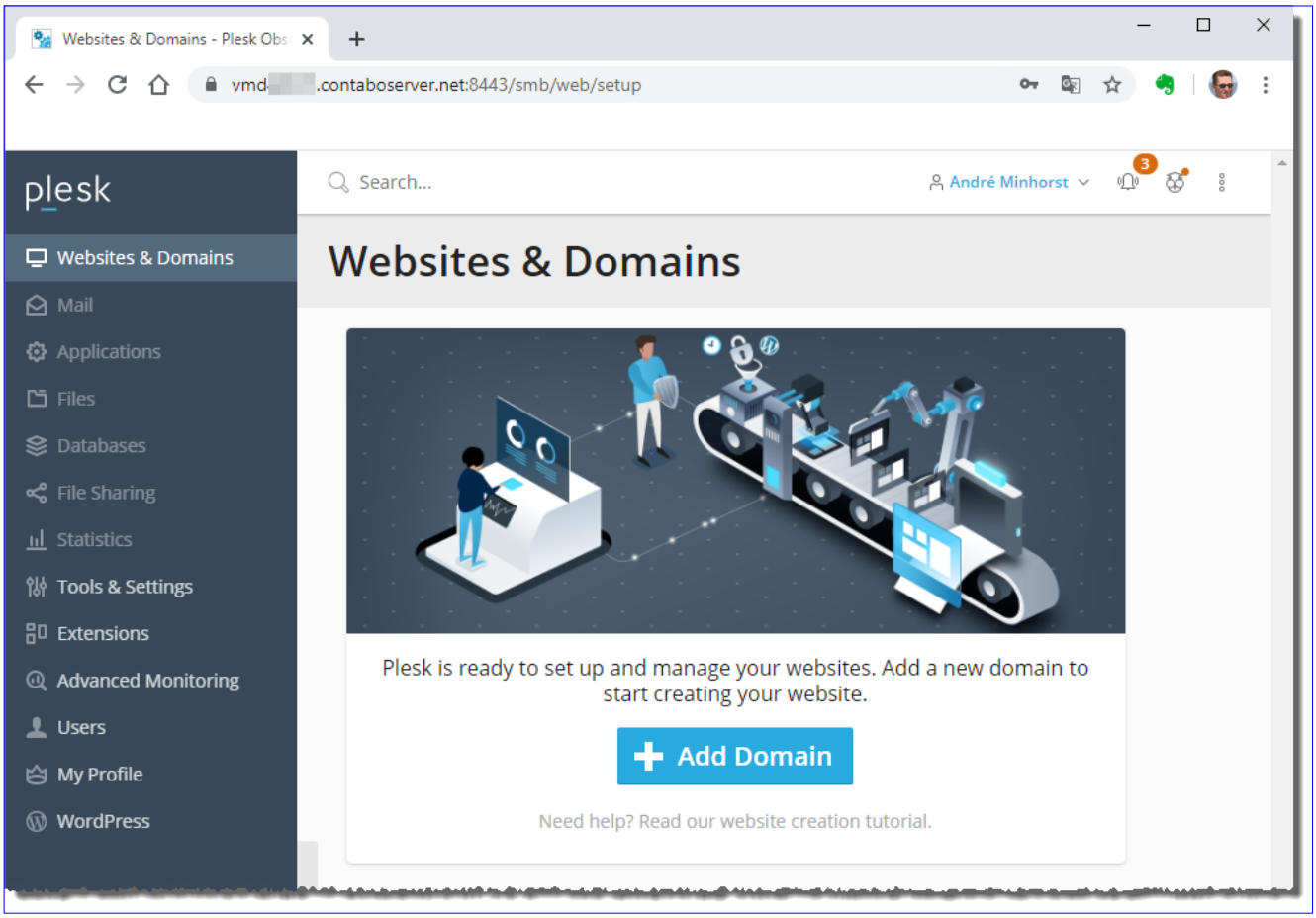

Bild 5: Benutzeroberfläche von Plesk

Schließlich geben Sie unten noch den Aktivierungscode für die Freischaltung der Plesk-Lizenz ein, die Sie ebenfalls in der E-Mail finden. Dann fehlt noch ein Haken für die Endbenutzer-Lizenzvereinbarung und Sie können mit einem Klick auf die Schaltfläche **Enter Plesk >>>** starten.

Plesk wird dann zunächst initialisiert, was eine Weile dauern kann. Nach einigen Minuten erwartet Sie dann allerdings auch schon die Benutzeroberfläche, die wie in Bild 5 aussieht.

Hier werden Sie direkt mit der Aufforderung begrüßt, eine Domain hinzuzufügen – das sollte Sie aber nicht verunsichern, da Sie für die reine Nutzung einer MySQL-Datenbank eigentlich keine Domain benötigen.

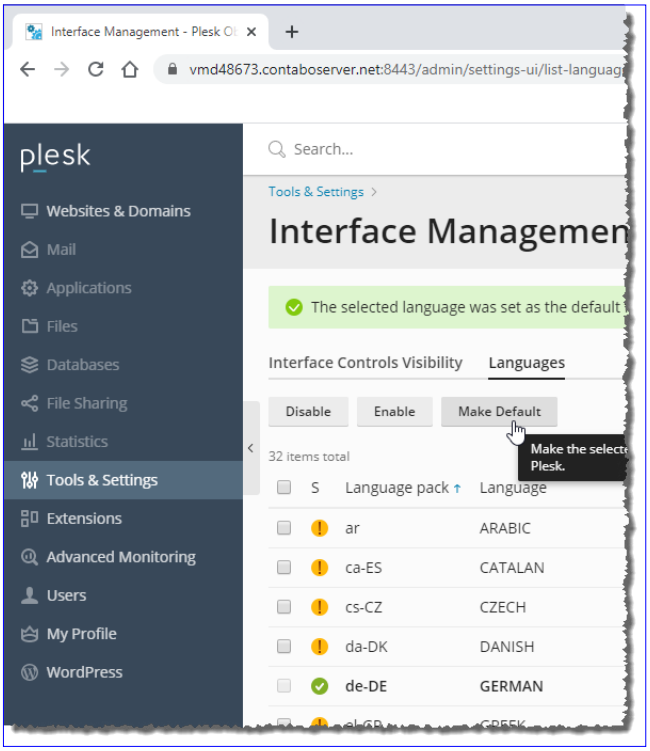

Bild 6: Einstellen der Sprache

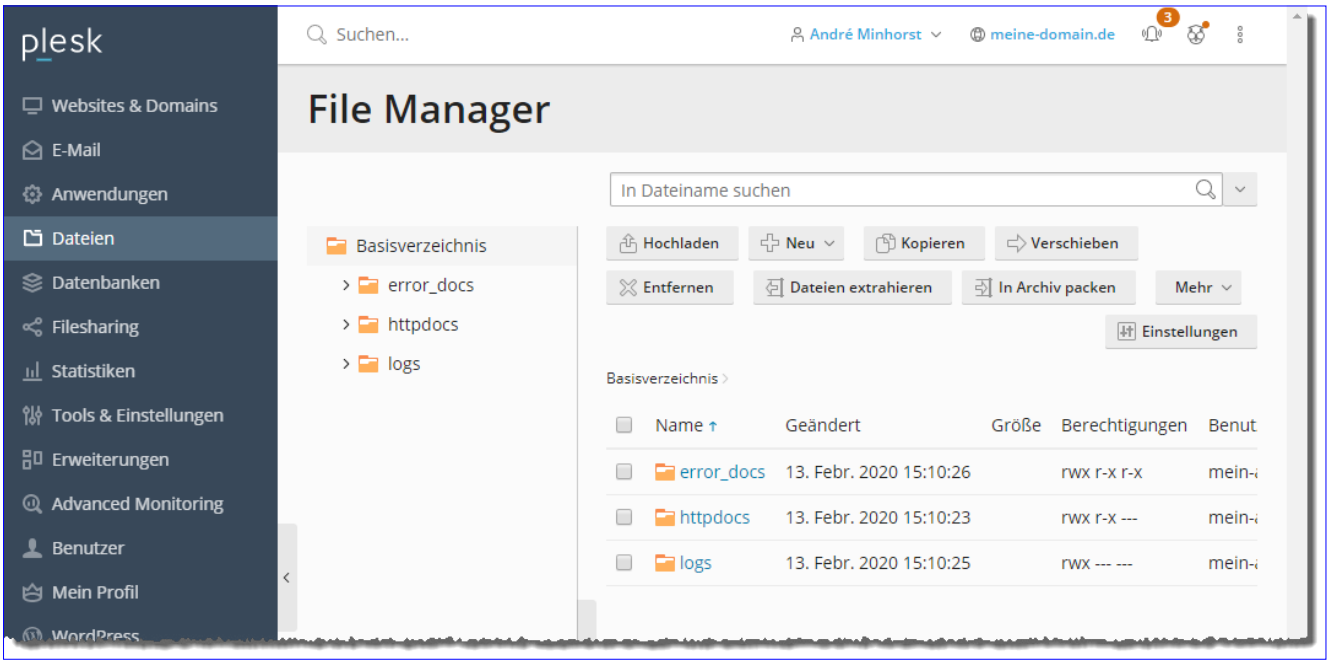

Bild 7: Verwalten der Dateien in Plesk

Außerdem ist mit dem Hinzufügen einer Domain nicht das Registrieren einer Domain gemeint – es entstehen also keine Zusatzkosten.

Allerdings benötigt Plesk die Angabe einer Domain, da sonst die übrigen Bereiche nicht freigeschaltet werden und Elemente wie Mails, Dateien und Datenbanken jeweils einer Domain zugeordnet werden sollen.

Die Angabe der Domain ist ein rein organisatorischer Vorgang, der Sie nicht verpflichtet, eine Domain zu reservieren – Sie geben einfach irgendeine Domain an, beispielsweise **www.meine-domain.de**.

Anwendungssprache von Plesk einstellen

Bevor wir uns darum kümmern, stellen wir allerdings zunächst die Anwendungssprache auf Deutsch ein.

Das gelingt wie folgt:

- Wechseln Sie zum Bereich **Tools & Settings**.
- Klicken Sie dort unter **Plesk Appearance** auf **Languages**.
- Hier markieren Sie in der Liste unter **Languages** die Sprache **GERMAN** und klicken dann auf die Schaltfläche **Make Default** (siehe Bild 6).
- Loggen Sie sich anschließend aus und wieder ein und Sie finden die komplette Benutzeroberfläche in deutscher Sprache vor (siehe Bild 7).

Domain hinzufügen

Damit hat sich auch der Befehl zum Hinzufügen einer Domain geändert. Dieser lautet jetzt **Domain hinzufügen**. Nachdem Sie diese Schaltfläche angeklickt haben, landen Sie auf der Seite **Hinzufügen eines neuen Webspace**. Hier geben Sie den gewünschten Domainnamen ein und legen ein Benutzerkonto mit Benutzername und Passwort an. Ein Klick auf die Schaltfläche **OK** startet dann den Prozess, der zum Anlegen der Webseite führt.

Funktionen von Plesk

Im linken Bereich finden Sie eine Liste der verschiedenen Funktionen von Plesk, die nun nach dem Anlegen einer Domain auch alle freigeschaltet sind. Unter **Websites & Domains** können Sie Domains hinzufügen, die Sie auf

diesem virtuellen Server hosten wollen. Unter **E-Mail** können Sie E-Mail-Adressen anlegen, allerdings nur für die Domains, die Sie zuvor über **Domain hinzufügen** zu Plesk hinzugefügt haben. Unter Anwendungen können Sie beliebte Anwendungen wir **WordPress**, **joomla** oder **Drupal** installieren. Es stehen auch noch weitere Anwendungen zur Verfügung. Das wollen wir hier allerdings nicht erledigen. Unter **Dateien** finden Sie den **File Manager**, der die Verzeichnisstruktur des virtuellen Servers abbildet.

Unter **httpdocs** befinden sich üblicherweise eine Seite namens **index.html** oder **index.php**, die als Startseite der Webseite aufgerufen wird. Dort finden Sie auch noch andere Dateien, die in der Regel in verschiedenen Verzeichnissen organisiert sind.

Ausflug: Anzeigen der Webseite

Wie können Sie diese Webseite nun in ihrem Browser anzeigen? Dazu ist nur eine kleine Änderung nötig, die Sie in einer speziellen Datei auf Ihrem System vornehmen – nämlich der **hosts**-Datei. Diese finden Sie hier:

```
C:\Windows\System32\drivers\etc\hosts
```

Diese Datei öffnen Sie mit einem Texteditor. Zum Ändern der Datei sind Administratorrechte erforderlich.

Der Datei fügen Sie eine Zeile hinzu, die als erstes die IP Ihres virtuellen Webservers enthält und als zweites die Domain, die Sie für Ihre Webseite festgelegt haben.

Die neue Zeile sieht dann beispielsweise so aus:

```
173.249.57.105 meine-domain.de
```

Wenn Sie nun im Browser die Adresse **meine-domain.de** eingeben, finden Sie die Seite aus Bild 8 vor. Das ist die Seite, die auf der **index.html**-Seite definiert wird, die sich

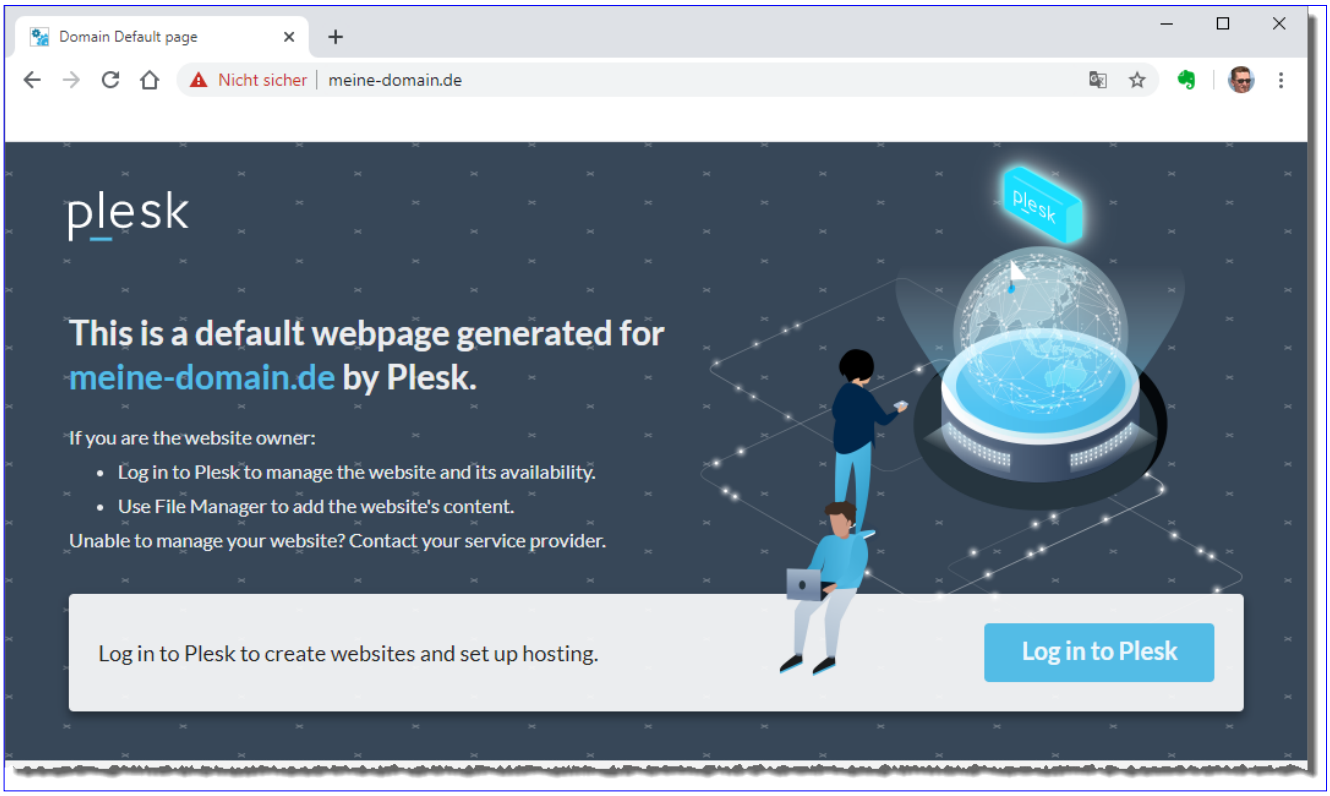

Bild 8: Die neue Webseite

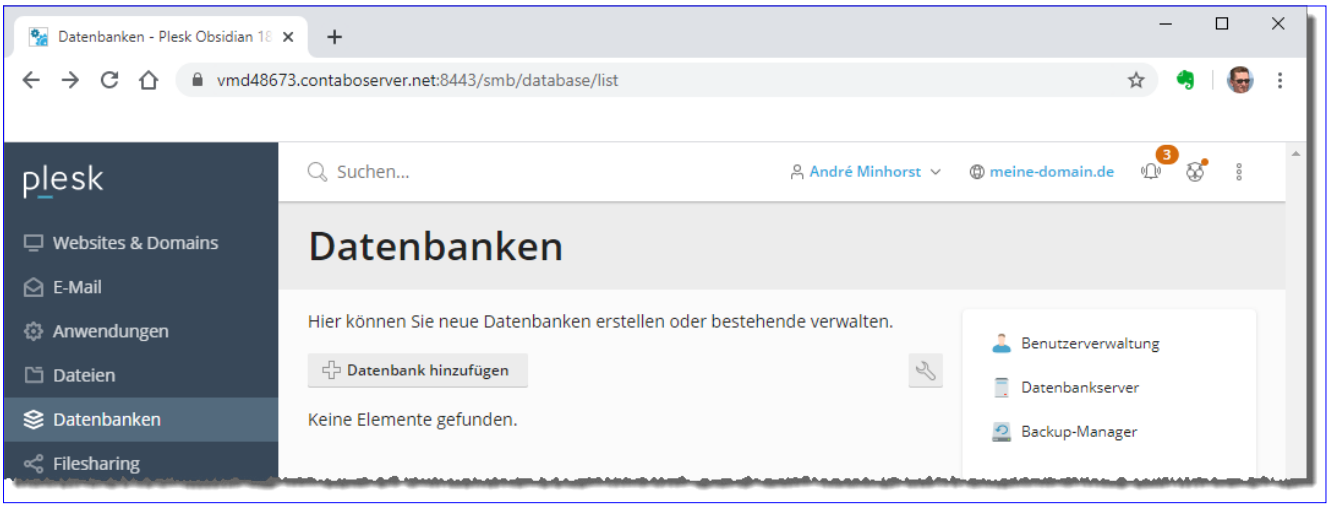

Bild 9: Der Bereich **Datenbanken** von Plesk

im Ordner **httpdocs** des Servers befindet. Das ist spannend, da Sie normalerweise unter **meine-domain.de** eine ganz andere Webseite vorfinden, da diese Domain bereits von einem anderen Unternehmen registriert ist.

Was ist hier geschehen? Wir teilen unserem System über die **hosts**-Datei mit, dass es, wenn wir die Adresse **meine-domain.de** in einen Browser eingeben, nicht die Name-Server durchsucht, die normalerweise Informationen enthalten, unter welcher Domain diese Webseite erreichbar ist, sondern unter der zu Beginn der Zeile angegebenen IP nach der Webseite suchen soll. Und da wir unter der in der **hosts**-Datei angegebenen IP eine Domain namens **meine-domain.de** angelegt haben, wird beim Aufruf im Browser unsere Webseite aufgerufen statt die, die normalerweise angezeigt wird.

Sie können im Bereich **Dateien** auch die Datei **index.html** aufrufen und diese verändern und sich dann nach dem Aktualisieren der Internetadresse **meine-domain.de** davon überzeugen, dass die Seite tatsächlich von Ihrem Webspace stammt.

Diese Information nur als Hinweis, falls Sie nicht nur die Datenbankfunktionen Ihres Pakets nutzen wollen, sondern auch direkt mit der Webseite experimentieren wollen.

Weitere Funktionen von Plesk

Damit zurück zu den Funktionen, die Sie über die linke Leiste von Plesk aufrufen können. Hier finden wir als nächstes den Eintrag **Datenbanken** und damit den Eintrag, der uns eigentlich interessiert.

Wechseln wir dorthin, finden Sie neben einigen weiteren Elementen die Schaltfläche **Datenbank hinzufügen** vor (siehe Bild 9). Klicken Sie diese Schaltfläche an, um eine neue Datenbank zu erstellen.

Danach erscheint der Dialog **Datenbank hinzufügen** (siehe Bild 10). Hier geben Sie folgende Informationen ein:

- **Datenbankname**: Name der Datenbank, beispielsweise **meine-datenbank**
- **Zugehörige Website**: Kann auf **Keine zugehörigen Websites** eingestellt bleiben, aber Sie können auch die erstellte Webseite auswählen.
- **Datenbankbenutzer erstellen**: Diese Option aktivieren wir, weil wir noch keinen Datenbankbenutzer erstellt haben. Später können Sie diese Option deaktiviert lassen, wenn Sie einen bereits vorhandenen Datenbankbenutzer verwenden wollen.

MySQL im Web: Verwaltung mit phpMyAdmin

Im ersten Teil der Beitragsreihe haben wir gezeigt, wie Sie einen virtuellen Server bei einem Anbieter einrichten und dort eine Datenbank erstellen. Die Verwaltung des virtuellen Servers erfolgt dort mit Plesk. Zur Verwaltung der Datenbank gibt es ein weiteres Programm namens phpMyAdmin. Diese ist ebenfalls Teil des gebuchten Pakets. In diesem Beitrag schauen wir uns an, wie Sie damit die MySQL-Datenbank verwalten.

Im ersten Teil der Beitragsreihe haben wir gezeigt, wie Sie Datenbanken mit Plesk erstellen. Wir greifen dieses Wissen nun auf und erstellen eine neue Datenbank namens **Zeiterfassung**. Eine Zeiterfassung ist ein gutes Beispiel, wenn es darum geht, von verschiedenen Standorten auf die Daten einer Datenbank zuzugreifen. Wir wollen im Rahmen dieses Beitrags aber auch nur die notwendigen Tabellen erstellen, um das Frontend der Datenbank kümmern wir uns später.

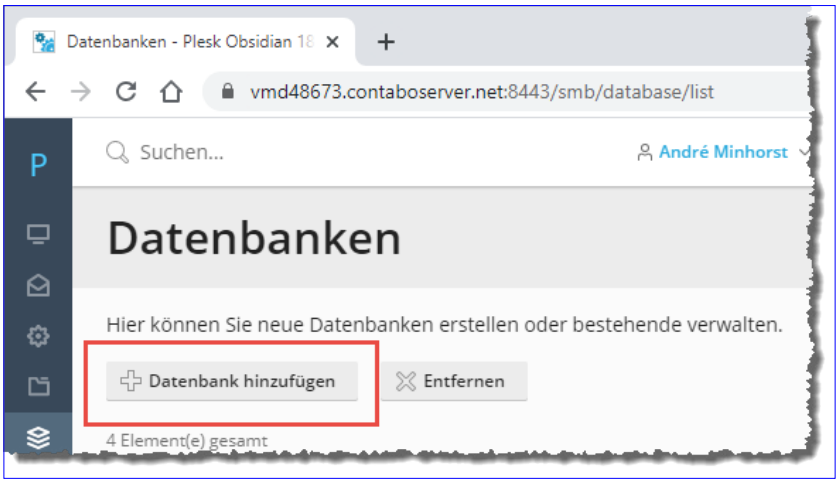

Bild 1: Anlegen einer neuen Datenbank

Datenbank erstellen

Um mit der Datenbank unter **phpMyAdmin** zu arbeiten, müssen wir diese zunächst erstellen. Das erledigen wir noch unter Plesk. Dazu gehen Sie zum Bereich **Datenbanken** und klicken dort auf die Schaltfläche **Datenbank hinzufügen** (siehe Bild 1).

Auf der nun erscheinenden Seite **Datenbank hinzufügen** geben Sie den Namen der neuen Datenbank ein. Da wir später von Access aus auf diese Datenbank zugreifen wollen, verknüpfen wir diese mit keiner Webseite. Einen neuen Benutzer legen wir auch nicht an. Das sorgt dafür, dass der Datenbank die Benutzer, für die unter **Datenbank** der Wert **Beliebig** eingestellt ist, als Benutzer hinzugefügt

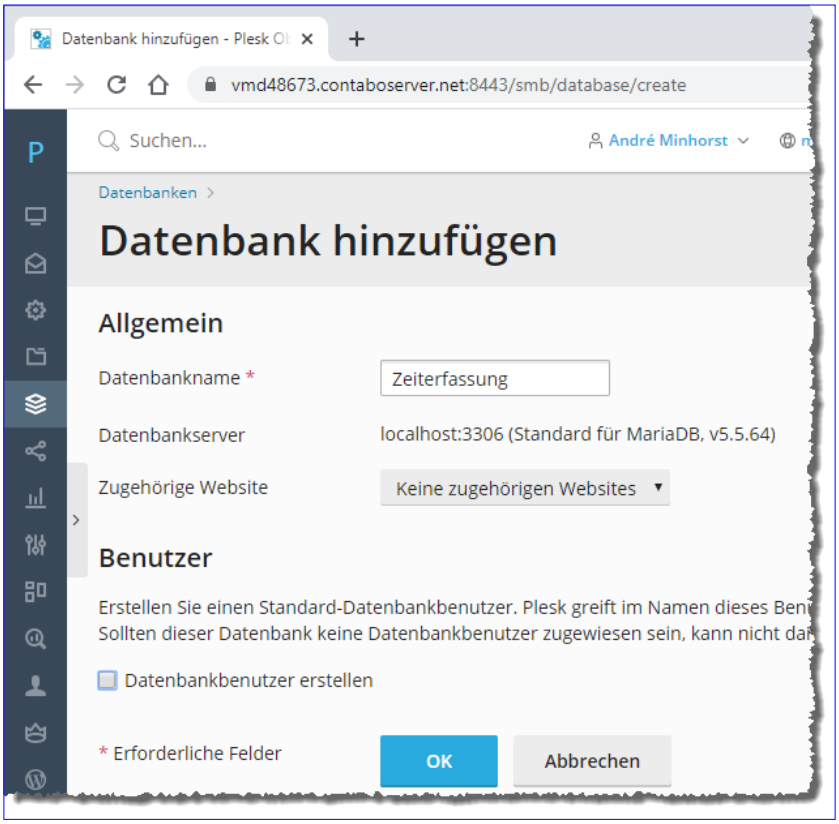

Bild 2: Hinzufügen der neuen Datenbank

werden (siehe Bild 2). Nach dem Erstellen der Datenbank zeigt Plesk wieder die Seite **Datenbanken** an, diesmal mit der neu hinzugefügten Datenbank in der Liste.

Klicken Sie auf die Schaltfläche rechts in der Liste der Datenbanken, erscheint eine Art Kontextmenü mit einigen Befehlen. Uns interessiert hier der Befehl **phpMyAdmin** (siehe Bild 3).

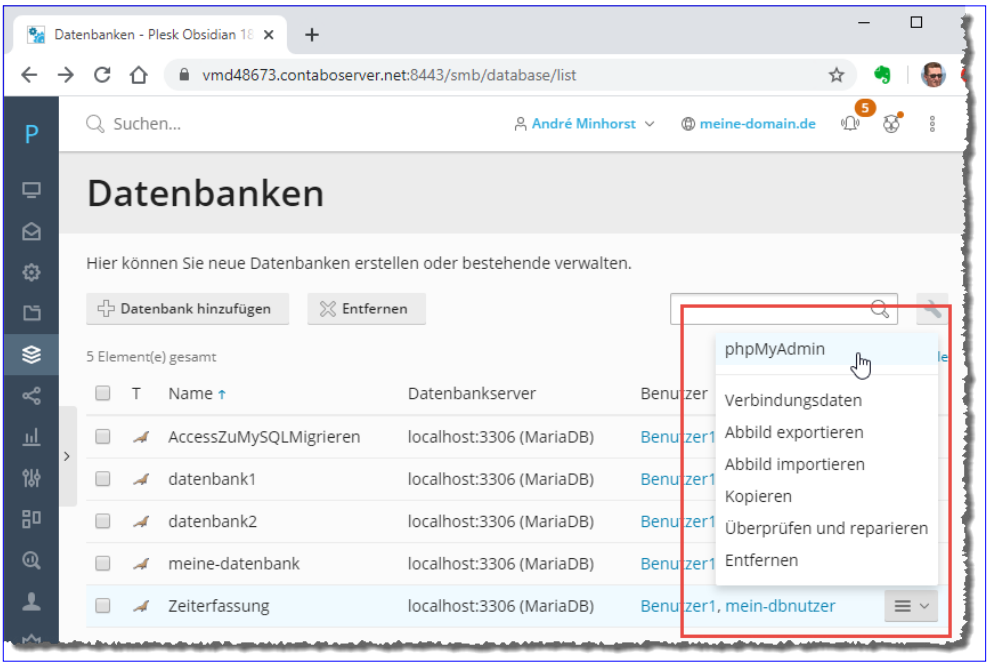

Bild 3: Kontextmenü der neuen Datenbank

Klicken wir diesen an, erscheint eine neue Webseite, die wie in Bild 4 aussieht. Diese Seite ist noch recht überschaubar, denn sowohl der Bereich links zeigt noch keine Tabellen an und auch im Hauptbereich finden wir keinerlei Einträge vor. Dafür sehen wir aber schon einige Möglichkeiten, Befehle aufzurufen – unter anderem eine Schaltfläche mit der Beschriftung **Erzeuge Tabelle**. Allerdings handelt es sich dabei gar nicht um eine Schaltfläche, sondern um die Überschrift des entsprechenden Bereichs, in dem wir den Namen der Tabelle und die Feldanzahl eintragen können.

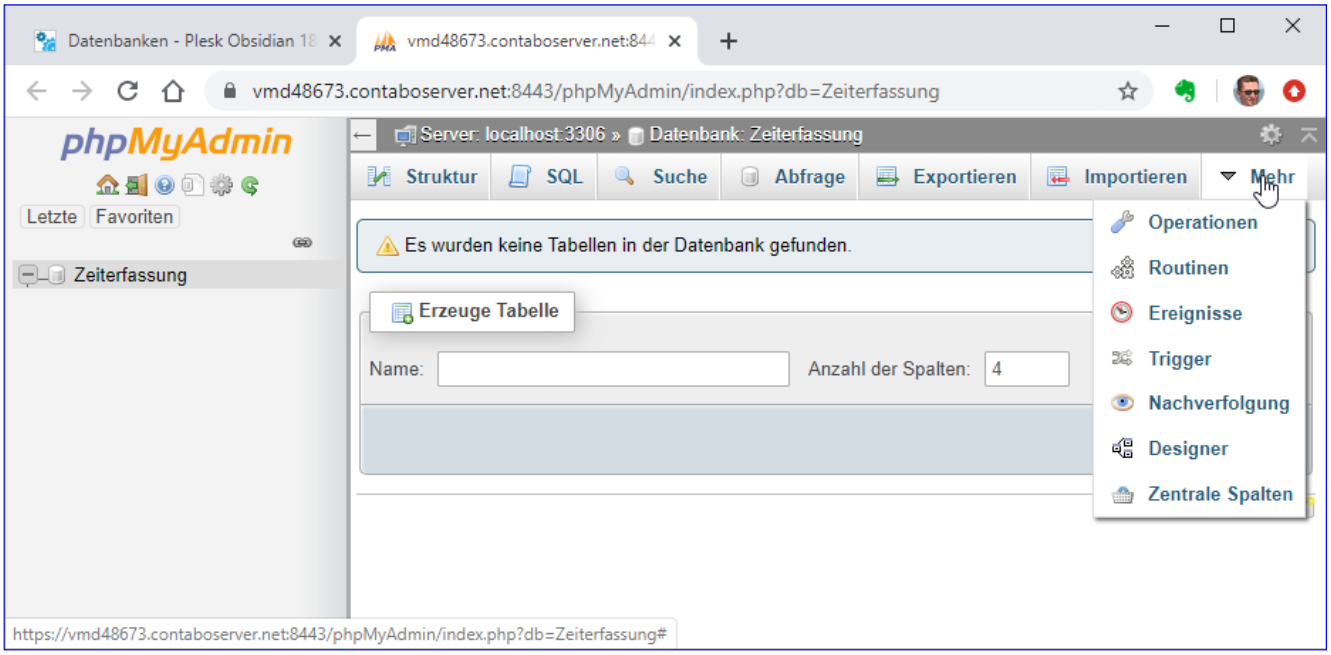

Bild 4: phpMyAdmin-Seite für die neue Datenbank

Tabelle erstellen

Hier geben wir dann den Namen der ersten Tabelle namens **tblMitarbeiter** ein und belassen die Anzahl der Spalten zunächst bei **4** (siehe Bild 5). Ein Klick auf die Schaltfläche **OK** zeigt dann die Entwurfsansicht der neuen Tabelle an. Hier tragen wir die Namen der gewünschten Felder und die übrigen benötigten Informationen ein.

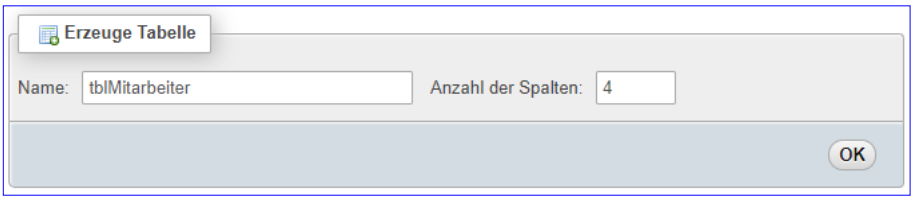

Bild 5: Anlegen einer neuen Tabelle

Felddatentypen

Hier ist eine Übersicht der wichtigsten Felddatentypen unter MySQL:

- **TINYINT**: Zahlen zwischen -128 und 127 oder unsigned zwischen 0 und 255. Wird beispielsweise zum Speichern von Boolean-Werten genutzt.
- **INT**: Zahlen von -2.147.483.648 bis 2.147.483.647 oder unsigned von 0 bis 4.294.967.295
- **DOUBLE**: Fließkommazahlen
- **VARCHAR**: Texte mit bis zu 65.535 Zeichen. Erfordert Angabe der maximalen Zeichenanzahl.
- **TEXT**: Texte bis 65.535 Zeichen. Angabe der Länge entfällt.
- **LONGTEXT**: Texte mit maximal ca. 4 Milliarden Zeichen
- **BLOB** (Binary Large Object): Speichern von Binärdaten wie beispielsweise von Bildern, bis 64 Kilobyte.
- **LONGBLOB**: Speichern von Binärdaten wie beispielsweise von Bildern, bis 4 Gigabyte.
- **DATE**: Abspeichern des Datums im Format **YYYY-MM-DD**.
- **TIME**: Abspeichern der Uhrzeit im Format **HH:MM:SS**.
- **DATETIME/TIMESTAMP**: Abspeichern des Datums und der Uhrzeit im Format **YYYY-MM-DD HH:MM:SS**

Primärschlüsselfeld anlegen

Das Primärschlüsselfeld der Tabelle soll schlicht **ID** heißen. Als Datentyp verwenden wir in der Spalte **Typ** den Wert **INT**. Da wir keine negativen Werte benötigen, können wir das Feld als unsigned markieren, was wir in der Spalte **Attribute** durch Auswahl des Eintrags **UNSIGNED** erledigen. Das Feld darf nicht den Wert **NULL** erhalten, also markieren wir die Option **Null** für dieses Feld der Tabelle nicht. Unter Index wählen wir den Wert **PRIMARY** aus. Damit öffnen wir einen Dialog namens **Index hinzufügen**, dessen Voreinstellungen wir beibehalten können (siehe Bild 6).

Bild 6: Eigenschaften des neuen Indexes

Hier finden wir allerdings keine Möglichkeit, das Feld als Autowert zu definieren. Dies schauen wir uns später an.

Weitere Felder definieren

Zunächst definieren wir noch die weiteren Felder, und zwar **Vorname** und **Nachname**, jeweils mit dem Typ **VARCHAR** und der Länge **255** und das Feld **AnredeID** mit dem Typ **INT**. Bevor wir die Tabelle speichern, sieht der Entwurf nun wie in Bild 7 aus.

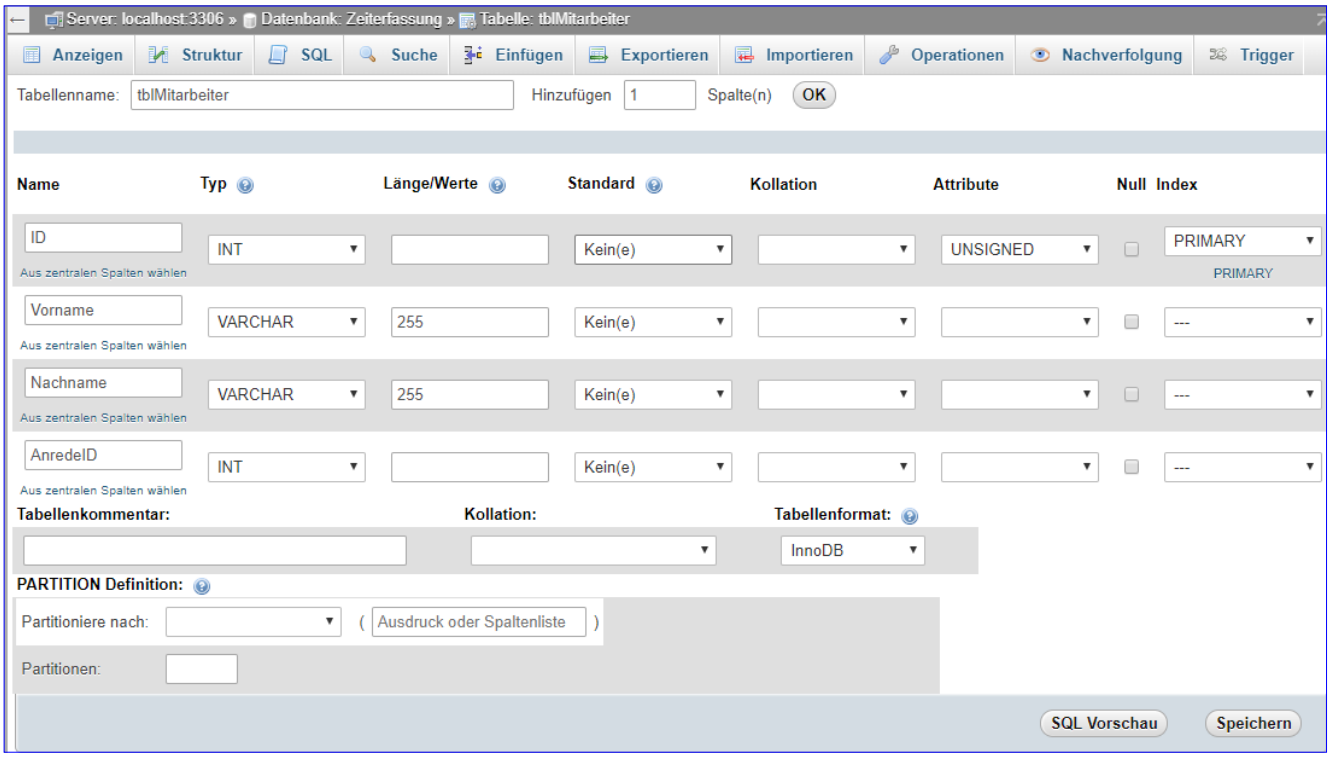

Bild 7: Anlegen der Felder der ersten Tabelle

Nun klicken wir auf die Schaltfläche **SQL Vorschau**, um uns anzusehen, wie die SQL-Anweisung zum Erstellen dieser Tabelle aussieht. Dies öffnet das Popup-Fenster aus Bild 8.

Die Definition sieht gut aus, allerdings sehen wir hier keinen Hinweis auf die Erstellung einer Autowert-Eigenschaft für das Feld **ID**. Kein Wunder: Den Autowert haben wir in der Entwurfsansicht nämlich übersehen.

Wir können diesen in der unscheinbaren Spalte mit der Überschrift **A_I** definieren. **A_I** steht für **Auto_Increment** ... Nachdem wir dies erledigt haben, sieht die SQL-Anweisung zum Erstellen der Tabelle schon besser aus:

```
CREATE TABLE `Zeiterfassung`.`tblMitarbeiter` ( `ID` INT UNSIGNED NOT NULL AUTO_INCREMENT , `Vorname` VARCHAR(255) NOT NULL , `Nachname` VARCHAR(255) NOT NULL , `AnredeID` INT NOT NULL , PRIMARY KEY (`ID`)) ENGINE = InnoDB;
```

Ein Klick auf die Schaltfläche **Speichern** legt nun die erste Tabelle der neuen Datenbank an.

Diese wird nun sowohl in der Liste im linken Bereich sowie in der Detailansicht im Hauptbereich angezeigt (siehe Bild 9).

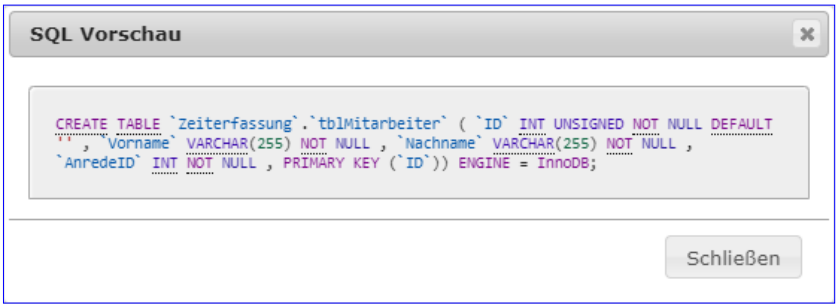

Bild 8: SQL-Anweisung zum Erstellen der Tabelle

Access-Datenbank zu MySQL migrieren

Wenn Sie die Tabellen einer Microsoft Access-Datenbank in eine MySQL-Datenbank migrieren wollen, haben Sie verschiedene Möglichkeiten. Sie können das von Hand erledigen, was eine Menge Arbeit ist, oder ein Tool zu einer mehr oder weniger automatischen Migration nutzen. MySQL selbst bietet dazu den MySQL Workbench Migration Wizard an, den wir uns in diesem Beitrag anschauen.

Download des Migration Wizards

Den **MySQL Workbench Migration Wizard** können Sie über den folgenden Link herunterladen.

Er ist Bestandteil der **MySQL Workbench**:

`https://dev.mysql.com/downloads/workbench/`

Für den Download benötigen Sie ein Konto bei Oracle, für das Sie sich gegebenenfalls noch registrieren müssen. Anschließend melden Sie sich mit diesem Konto an. Nach dem Download installieren Sie die MySQL Workbench, wobei Sie alle Standardvorgaben beibehalten können.

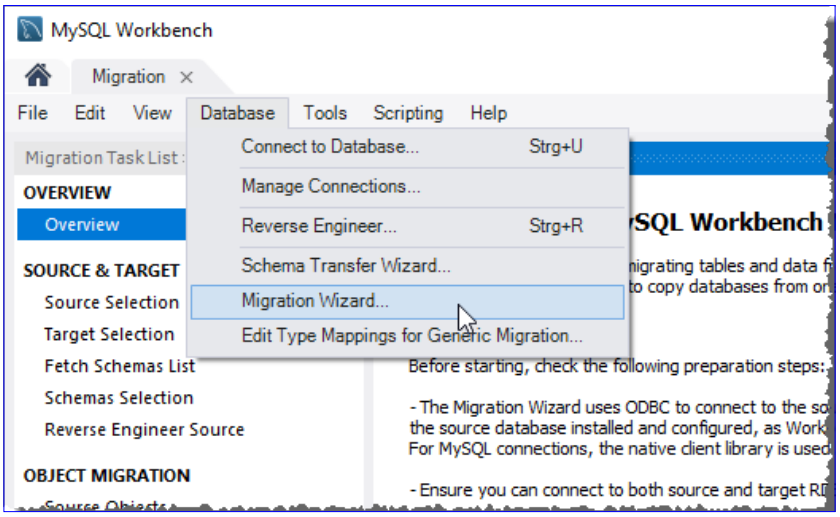

Bild 1: Die MySQL Workbench

Migration Wizard starten

Nach der Installation können Sie MySQL Workbench direkt starten. Dort finden Sie im Menü den Eintrag **Database|Migration Wizard** (siehe Bild 1).

Wählen Sie diesen Eintrag aus, erscheint der **MySQL Workbench Migration Wizard** (siehe Bild 2).

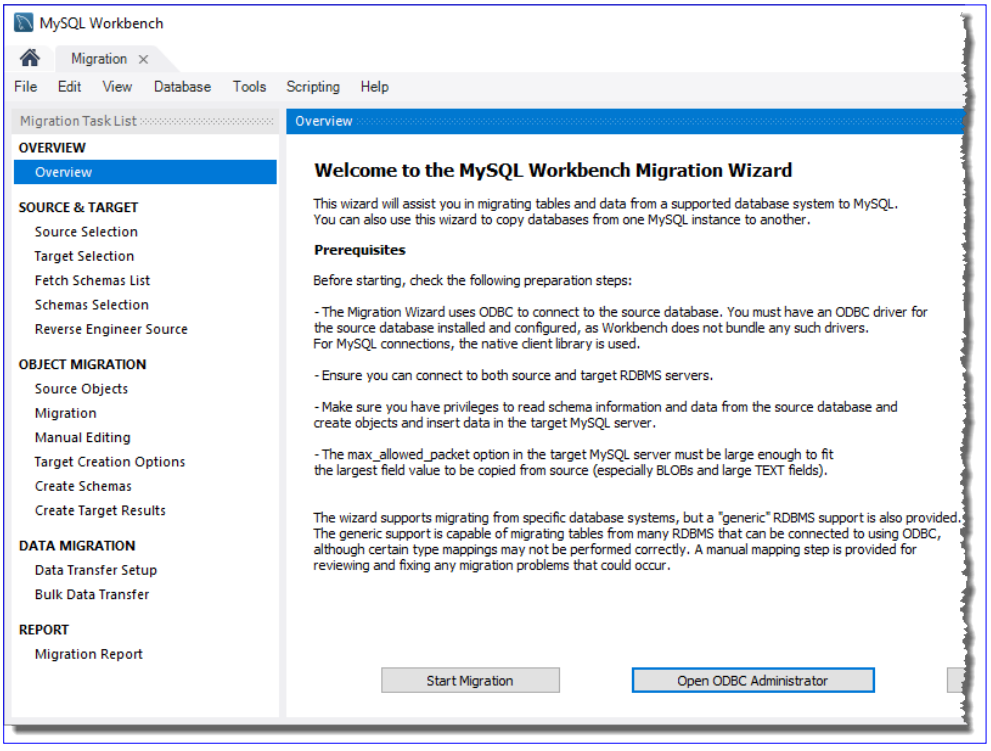

Bild 2: Der MySQL Workbench Migration Wizard

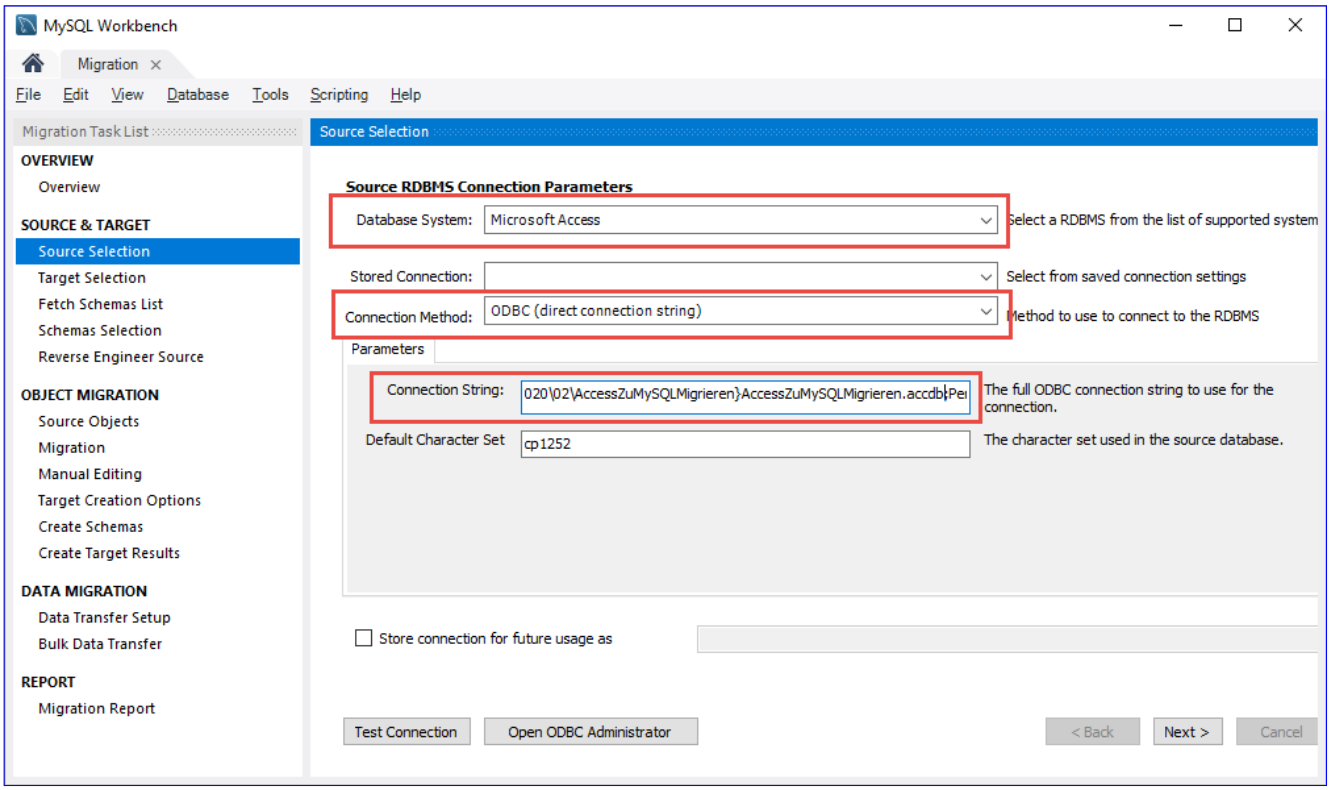

Bild 3: Versuch, mit einer direkt eingegebenen Verbindungszeichenfolge zu arbeiten

Der nächste Schritt (den Überblick über die einzelnen Schritte und den jeweils aktuellen Schritt sehen Sie im Bereich links) heißt **Source Selection** (siehe Bild 3). Unsere Quelle ist eine Microsoft Access-Datenbank. Wie sich in unserem Fall zeigte, kann man bei der Installation der MySQL Workbench durchaus etwas falsch machen: Sie müssen sich nämlich an einer Stelle entscheiden, die 32- oder die 64-Bit-Variante zu installieren. Für unseren Anwendungsfall, nämlich die Migration einer Access-Datenbank nach MySQL, müssen Sie die Version wählen, die zu der auf Ihrem System passenden Access-Version passt. Das heißt: Wenn Sie, wie wir, die 64-Bit-Version von MySQL Workbench installiert haben, aber mit der 32-Bit-Version von Access arbeiten, müssen Sie noch einmal einen Schritt zurückgehen. Wenn Sie bereits die zu Ihrer Access-Version passende Version von MySQL Workbench geladen haben, können Sie weiter unten unter **Neue Datenquelle für die Access-Datenbank anlegen** weiterlesen.

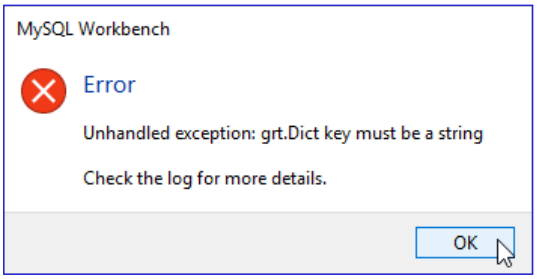

Bild 4: Fehlermeldung beim Test der Verbindung

Der harte Weg, Probleme mit der Version herauszufinden, ist das Ausprobieren mit einer nicht harmonierenden Kombination aus Microsoft Access und MySQL Workbench. Weil die anderen Einstellungen nicht sinnvoll erschienen, haben wir unter **Connection Method** den Eintrag **ODBC (direct connection string)** gewählt und unter **Connection String** eine Verbindungszeichenfolge wie die folgende angegeben:

```
Provider=Microsoft.ACE.
OLEDB.12.0;Data Source=C:\...\
AccessZuMySQLMigrieren\Access-
ZuMySQLMigrieren.accdb;Persist
Security Info=False;
```

Ein Klick auf die Schaltfläche **Test Connection** lieferte dann allerdings die interessante Fehlermeldung aus Bild 4.

Also haben wir uns dann einmal angesehen, welche Möglichkeiten wir erhalten, wenn wir auf die Schaltfläche **Open ODBC Administrator** klicken.

Dies öffnet den Dialog aus Bild 5. Wechseln wir hier zum Bereich **Treiber**, sehen wir nur verschiedene SQL Server-Treiber, aber keinen Access-Treiber. Dass oben in der Titelleiste der Hinweis auf 64-Bit steht, hat uns in die richtige Richtung geschoben. Also haben wir MySQL Workbench wieder deinstalliert.

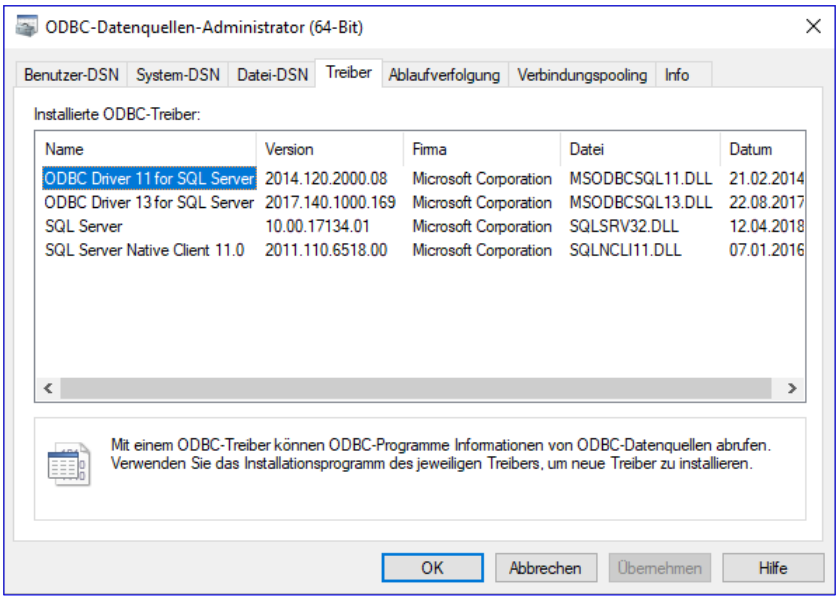

Bild 5: Der ODBC-Datenquellen-Administrator

Ein Blick auf die heruntergeladene Datei zeigt dann noch, dass wir offensichtlich die reine 64-Bit-Version erwischt haben (**mysql-workbench-community-8.0.19-winx64.msi**).

Die richtige Version finden wir schließlich nur in einer älteren Version im Archiv der Downloads:

`https://downloads.mysql.com/archives/workbench/`

Dies liefert die Seite aus Bild 6.

Hier klicken Sie auf den Eintrag **Windows (x86, 32-bit), MSI Installer**)

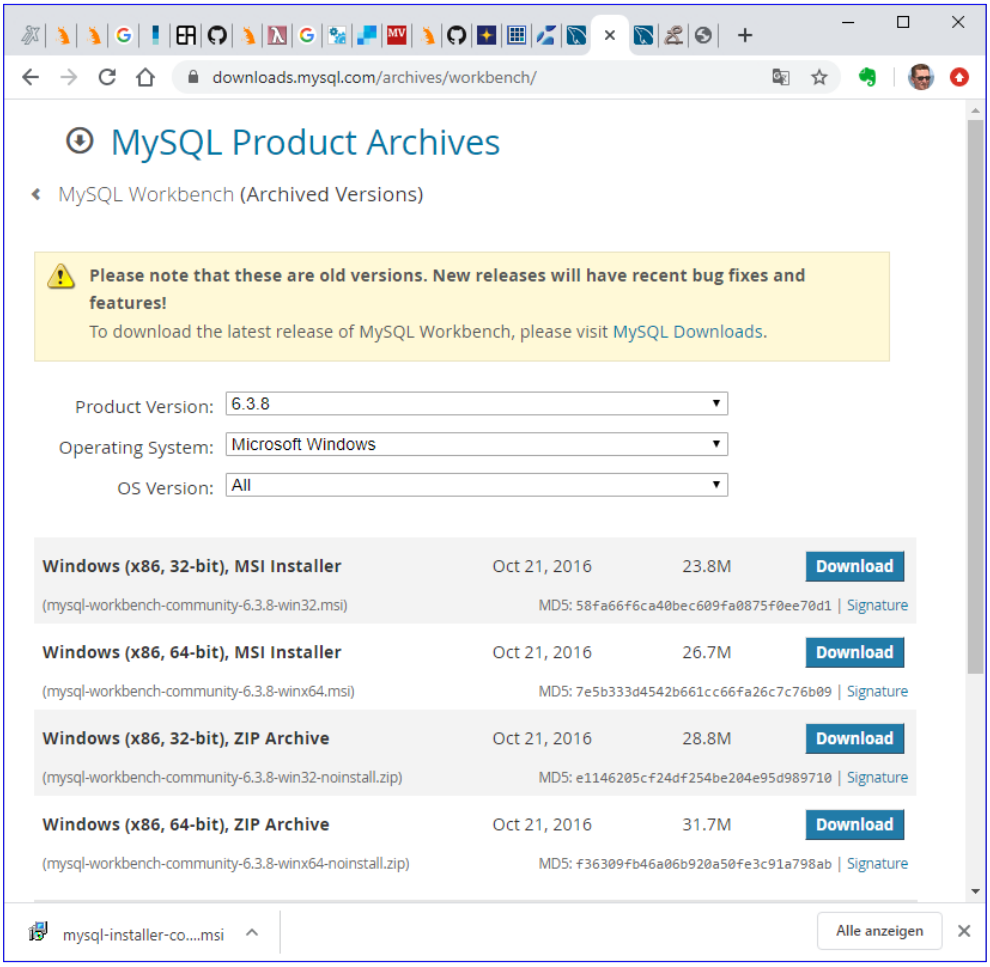

Bild 6: Download der 32-Bit-Version

und starten die Installation direkt nach dem Download.

32-bit-Workbench starten

Nach der Installation starten wir MySQL Workbench. Das Startfenster der älteren Version, die wir aus Kompatibilitätsgründen zu unserer 32-Bit-Access-Version nutzen, sieht ganz anders aus als das der neuen Version (siehe Bild 7).

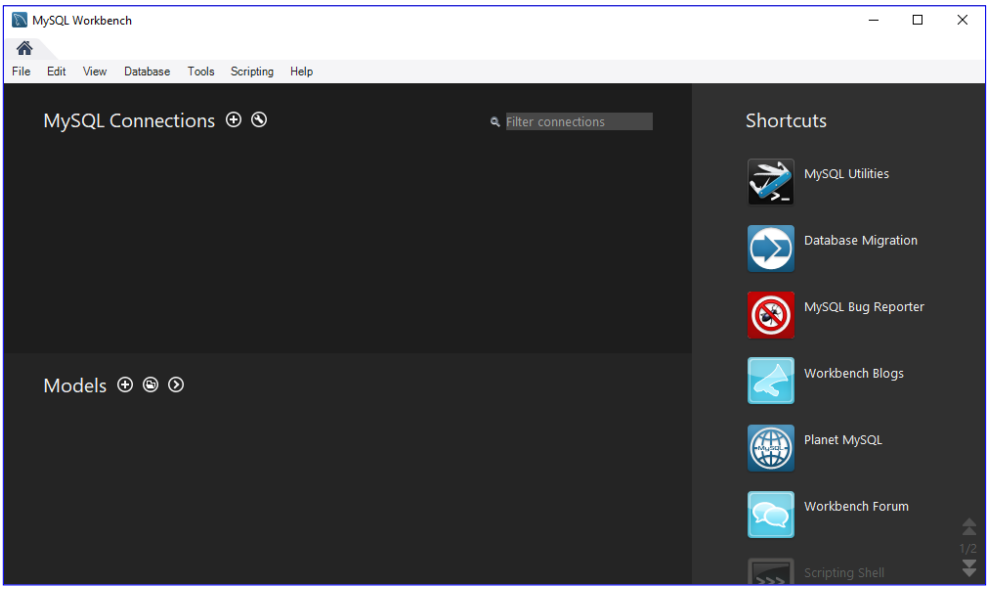

Bild 7: 32-bit-Version der MySQL Workbench

Wenn wir allerdings im rechten Bereich auf **Database Migration** klicken, finden wir uns gegenüber der aktuellen 64-Bit-Version schnell wieder zurecht.

Klicken wir dort auf die Schaltfläche **Open ODBC Administrator**, erscheint dieser in der 32-Bit-Version. Hier finden wir dann eher ein Überangebot an Treibern, und auch einige Access-Treiber (siehe Bild 8).

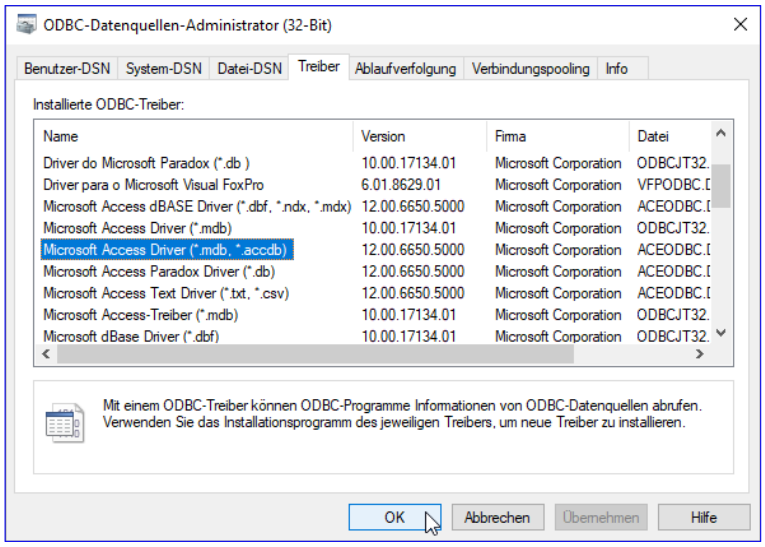

Bild 8: 32-Bit-Version des Datenquellen-Administrators

Neue Datenquelle für die Access-Datenbank anlegen

Wir wechseln zurück zur Seite **Benutzer-DSN** und klicken hier auf **Hinzufügen** (siehe Bild 9).

Dies öffnet einen weiteren Dialog namens **Neue Datenquelle erstellen** (siehe Bild 10). Hier müssen wir erst einmal den passenden Treiber finden. Wir entscheiden uns für den einzigen, der auch die Dateiendung **.accdb** in der Bezeichnung trägt.

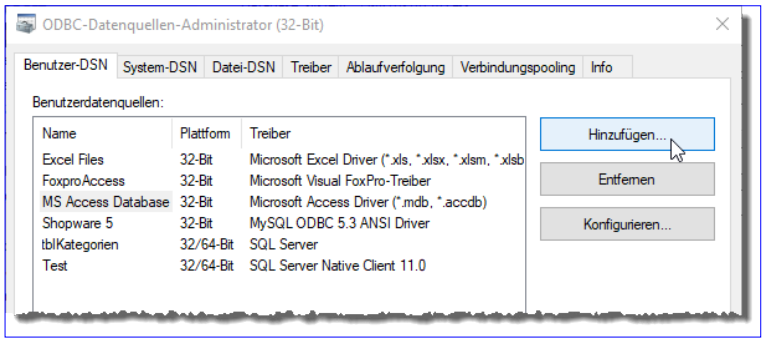

Bild 9: Neue Benutzer-DSN hinzufügen

Im nächsten Schritt nehmen wir die konkreten Einstellungen für die Datenquelle vor. Dazu geben wir einen Datenquellennamen an und

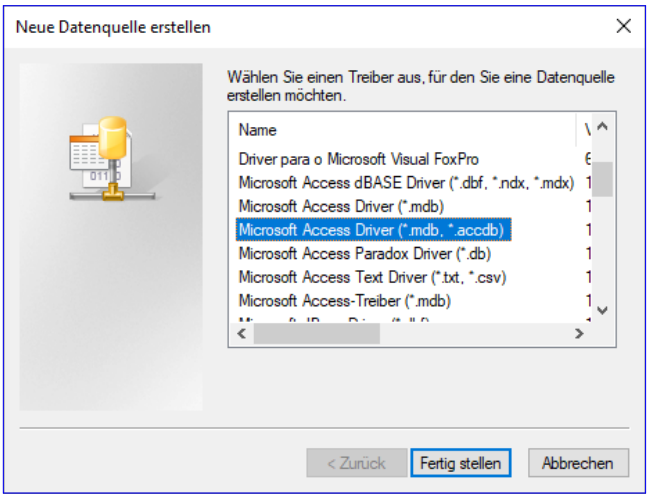

Bild 10: Auswählen des Treibers

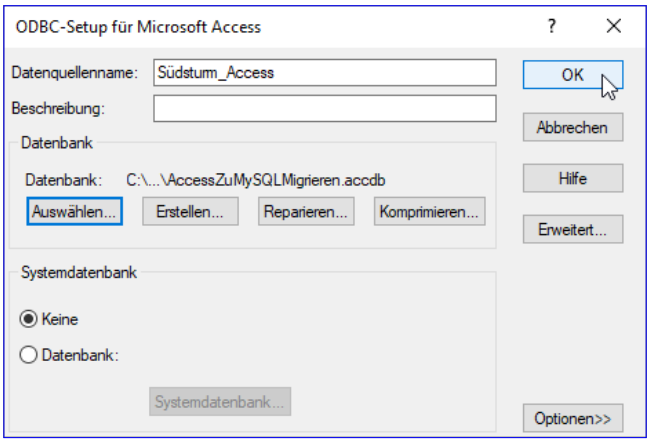

Bild 11: Einstellungen für die Datenquelle

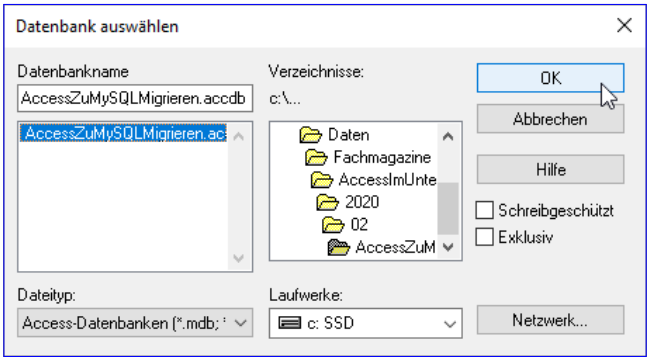

Bild 12: Auswahl der Datenbank

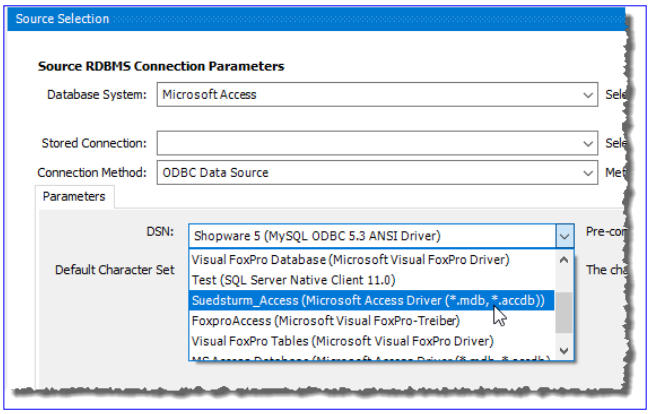

Bild 13: Auswahl der neuen DSN

wählen die Datenbank aus. Das Ergebnis finden Sie in Bild 11.

Für die Auswahl der Datenbank müssen wir einen Dateiauswahl-Dialog nutzen, der vermutlich noch aus der Windows 95-Zeit stammt (siehe Bild 12). Den neuen Eintrag wählen wir dann im ODBC-Datenquellen-Administrator aus und klicken auf **OK**, um den Dialog zu schließen.

Anschließend können wir in MySQL Workbench nach Hin- und Zurückwechseln der Verbindungsmethode (Auswahlfeld **Connection Method**) die soeben angelegte Datenquelle auswählen (siehe Bild 13). Damit klicken wir nun erneut auf die Schaltfläche **Test Connection** und erhalten endlich eine Erfolgsmeldung. Bevor wir fortfahren, nutzen wir die Möglichkeit, die Verbindung für spätere Verwendung zu speichern.

Zieldatenbank festlegen

Nun legen wir fest, in welche MySQL-Datenbank die Tabellen migriert werden sollen. Dies geschieht im Bereich **Target Selection** von MySQL Workbench. Dazu benötigen Sie einen laufenden MySQL-Server, entweder auf dem lokalen Rechner, auf einem Rechner im Netzwerk oder auch auf einem Internetrechner. Wir haben kein MySQL auf dem lokalen Rechner installiert, aber durchaus einen Internetserver, auf dem MySQL läuft. Müssen wir dort nun eine Datenbank anlegen oder wird diese bei der Migration automatisch angelegt? Wie der Dialog zum Eingeben der Parameter zeigt, ist das offensichtlich zunächst nicht der Fall. Hier brauchen wir nur die IP, den Port, den Benutzernamen und das Kennwort anzugeben. Das Kennwort geben Sie dabei ein, indem Sie auf **Store in Vault ...** klicken

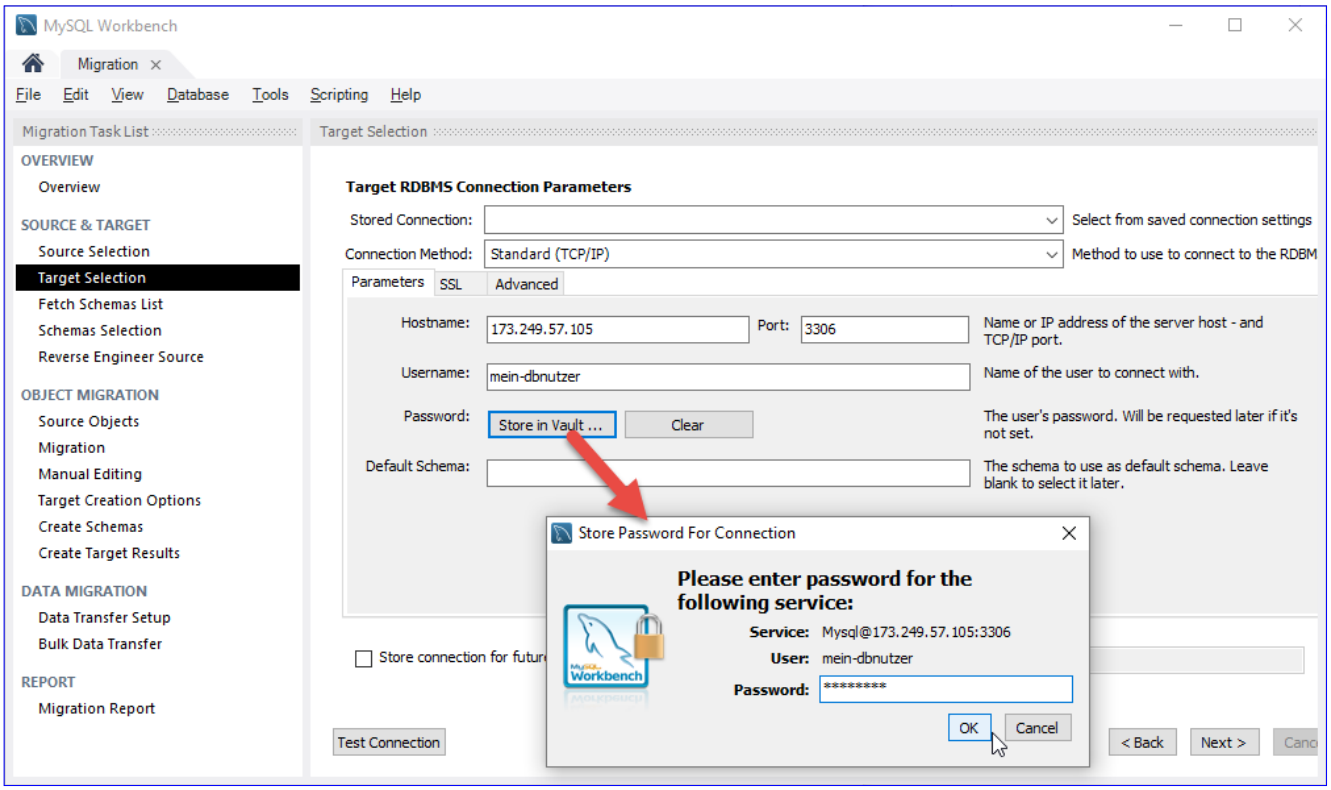

Bild 14: Angabe des Migrationsziels

und im Dialog **Store Password For Connection** das Kennwort eingeben (siehe Bild 14). Klicken Sie dann auf **Test Connection**, sollte bei korrekter Angabe aller Informationen eine Erfolgsmeldung erscheinen.

Für alle Fälle sichern wir die Verbindung für weitere Aufrufe, indem wir die Option **Store connection for future** aktivieren und eine entsprechende Bezeichnung eingeben.

Im nächsten Schritt greift MySQL Workbench auf den angegebenen Server zu und liest einige Informationen ein (siehe Bild 15).

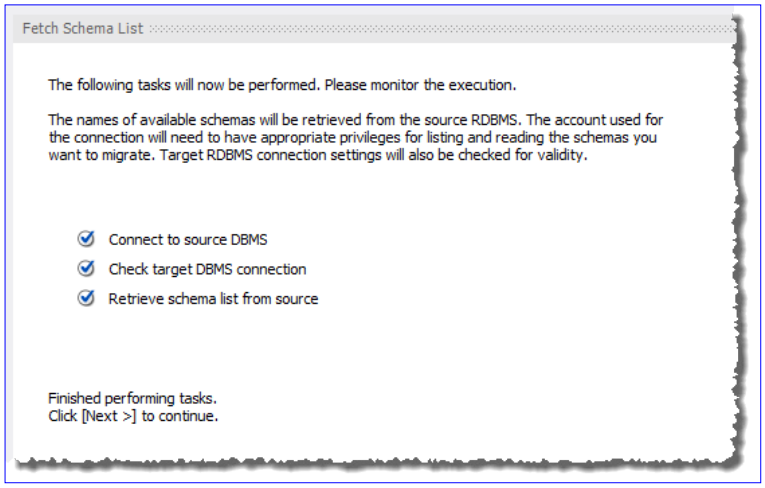

Bild 15: Sammeln von Informationen

Fehler beim Reverse Engineering

Danach folgt das Ermitteln des Datenmodells der zu migrierenden Datenbank. Dabei tritt, wie Bild 16 zeigt, ein Fehler auf. Dieser besagt, dass wir keinen lesenden Zugriff auf alle Systemtabellen der Datenbank festgelegt hätten.

Dieses Problem lösen wir, indem wir die Access-Datenbank öffnen, was üblicherweise mit dem Benutzer

MySQL im Web: Tools für das Access-Frontend, Teil 1

Wenn Sie eine Datenbank von mehreren Standorten aus nutzen, aber nicht auf den Komfort der Access-Programmierung verzichten wollen, dann gibt es nicht viele Möglichkeiten. Sie können entweder eine SQL Server-Datenbank oder eine MySQL-Datenbank auf einem Internetserver ablegen und für den Zugriff von einer Access-Datenbank aus verfügbar machen. Wie das gelingt, haben wir am Beispiel von MySQL in verschiedenen Beiträgen bereits dargestellt. Der vorliegende Beitrag schließt den Kreis und zeigt, wie Sie von einem Access-Frontend aus auf die MySQL-Datenbank auf dem Internetserver zugreifen und mit den Daten so arbeiten, als würden diese auf dem heimischen Rechner liegen. Wir starten mit einigen Tools, die den Zugriff erleichtern.

Formular zum Zusammenstellen einer Verbindungszeichenfolge

Um von Access aus auf eine MySQL-Datenbank zuzugreifen, benötigen Sie zuerst einmal eine Verbindungszeichenfolge. Um diese zusammenzustellen, wollen wir Ihnen ein praktisches Formular bereitstellen.

In dieses geben Sie die benötigten Daten wie Serveradresse, Datenbankname, Benutzername, Kennwort oder Port ein und prüfen, ob die Verbindung hergestellt werden kann. Damit das Formular komfortabel anzuwenden ist, benötigen wir einige Steuerelemente, einige Zeilen VBA-Code und auch Tabellen, in denen wir die ermittelten Daten speichern – und auch grundlegende Daten wie die Informationen über die zur Auswahl stehenden Treiber.

Treiber verwalten

Bevor Sie überhaupt von einer Access-Datenbank auf eine MySQL-Datenbank zugreifen können, benötigen Sie einen entsprechenden ODBC-Treiber.

Diesen erhalten Sie vom Datenbankhersteller, in diesem Fall von der folgenden Webseite:

```
https://dev.mysql.com/downloads/connector/odbc/
```

Hier finden Sie ODBC-Treiber für verschiedene Versionen vor (siehe Bild 1).

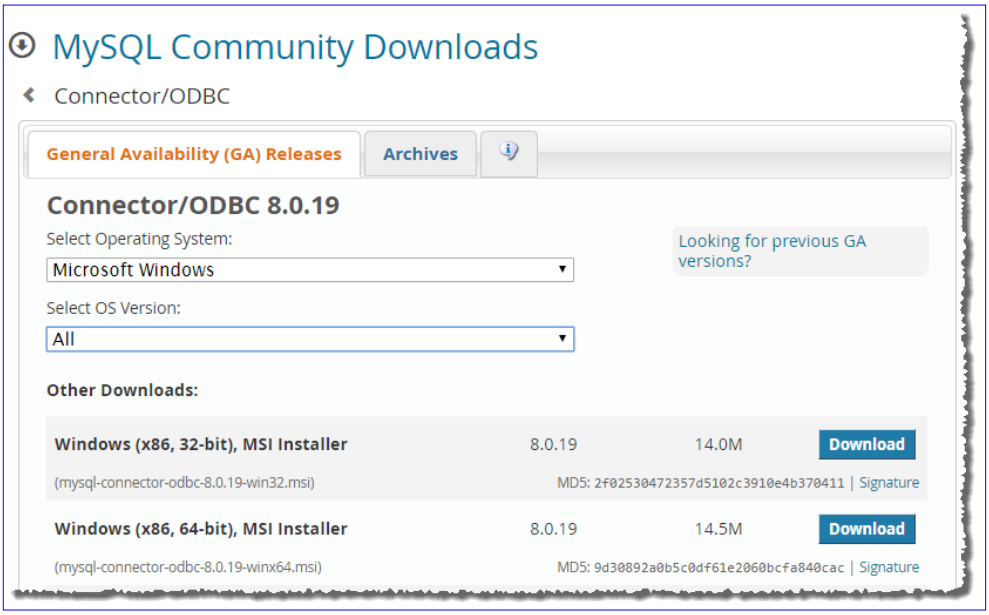

Bild 1: Download des ODBC-Treibers für MySQL

Wenn Sie bereits den Beitrag **Access-Datenbank zu MySQL migrieren** (**www.access-im-unternehmen.de/1229**) gelesen haben, kennen Sie schon eine Herausforderung, die auch bei der Auswahl des geeigneten Treibers auf Sie zukommt: Sie müssen zwischen der 32-Bit- und der 64-Bit-Version wählen.

Wer dabei meint, er müsse sich an der Windows-Version orientieren, die auf dem Zielrechner installiert ist, liegt falsch: Den MySQL-Treiber wählen Sie wie die MySQL Workbench-Version anhand der Variante des installierten Office-Pakets aus. Haben Sie Office in der 32-Bit-Version installiert, benötigen Sie also auch den entsprechenden ODBC-Treiber.

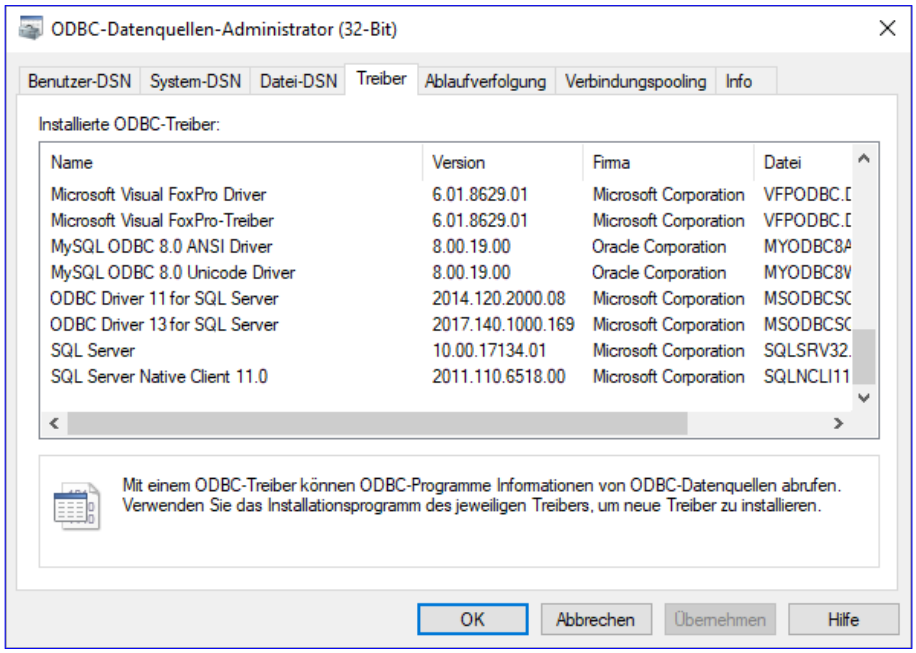

Bild 2: Prüfen, ob der richtige ODBC-Treiber installiert ist

Auf unserem Rechner ist die 32-Bit-Variante von Office installiert, also wählen wir von der obigen Seite den Download der Datei **mysql-connector-odbc-8.0.19-win32.msi**. Genau wie beim Download von MySQL Workbench benötigen Sie auch hier einen kostenlosen Zugang bei Oracle.

Nach dem Download installieren wir den Treiber auch direkt. Hierbei können Sie die Standardeinstellungen beibehalten. Beachten Sie, dass Microsoft Access während der Installation nicht gestartet sein darf.

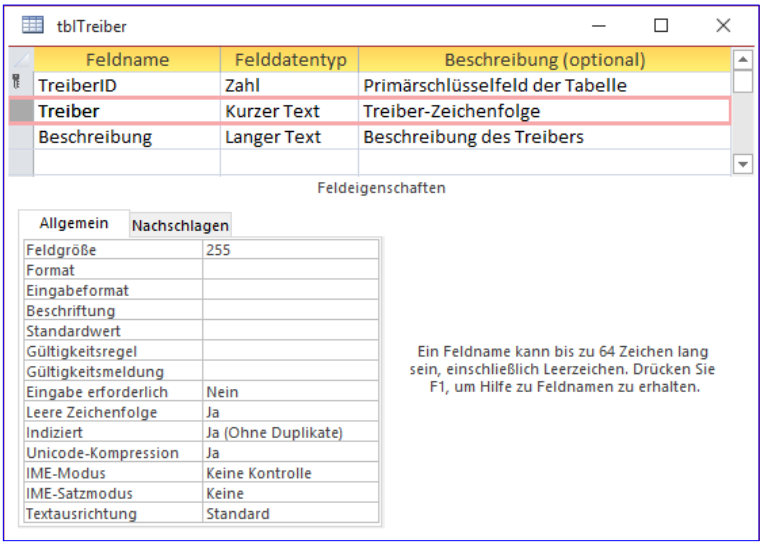

Bild 3: Entwurf der Tabelle zum Speichern der Treiberinformationen

Nach der Installation können Sie auf folgende Weise prüfen, ob Sie den richtigen Treiber installiert haben, je nachdem, ob Sie die 32-Bit- oder die 64-Bit-Version benötigen. Dazu geben Sie im Windows-Suchfeld **ODBC** ein und starten das dann erscheinende **ODBC-Datenquellen (32-Bit)** oder **ODBC-Datenquellen (64-Bit)**.

In dieser Anwendung wechseln Sie auf die Seite **Treiber** und prüfen, ob dort ein MySQL-ODBC-Treiber in der soeben installierten Version vorliegt – in unserem Fall die Version **8.0** (siehe Bild 2).

Speichern der Treiberinformationen

Die verschiedenen ODBC-Treiber haben verschiedene Zeichenfolgen, die in der ODBC-Verbindungszeichenfolge

zum Einsatz kommen. Damit wir diese nicht fest im Code verdrahten müssen, legen wir eine neue Tabelle namens **tblTreiber** an, mit der wir diese Zeichenketten speichern können.

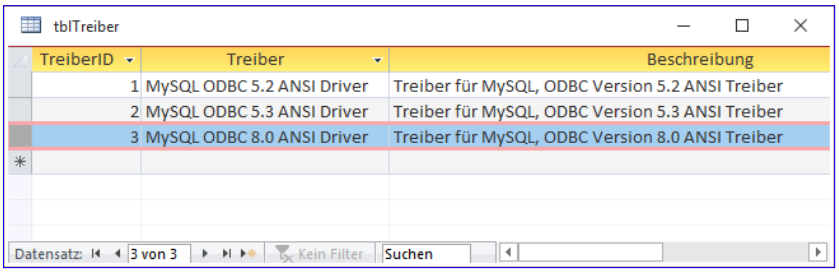

Bild 4: Einträge der Tabelle **tblTreiber**

Diese Tabelle enthält neben dem Primärschlüsselfeld noch ein Feld namens **Treiber**, um die Zeichenfolge aufzunehmen, und ein Beschreibungsfeld.

Den Entwurf sehen Sie in Bild 3. Für eine ältere Version des MySQL-Treibers lautet diese Zeichenfolge so:

```
MySQL ODBC 5.3 ANSI Driver
```

Wir haben nun allerdings soeben die Version 8.0 des Treibers installiert und die obige Zeichenfolge wird nicht mehr funktionieren. Ob man die 5.3 einfach durch die 8.0 ersetzen kann, werden wir gleich herausfinden. Wir fügen jedenfalls einfach mal einen Datensatz mit dieser Kombination hinzu (siehe Bild 4).

Formular zum Verwalten der Verbindungszeichenfolgen

Damit der Benutzer die Daten dieser Tabelle komfortabel verwalten kann, stellen wir ihm ein Formular zur Verfügung, das an die Tabelle **tblVerbindungszeichenfolgen** gebunden ist und wie in Bild 6 aussieht.

Das Kombinationsfeld oben zeigt die Bezeichnungen aus der Tabelle **tblVerbindungszeichenfolgen** an und stellt die gewählte Verbindungszeichenfolge ein. Dafür sorgt diese Ereignisprozedur:

```
Private Sub cboSchnellauswahl_AfterUpdate()
    Me.Recordset.FindFirst 7
```

Daten zu Verbindungszeichenfolgen speichern

Eine Verbindungszeichenfolge erfordert allerdings noch einige weitere Informationen wie etwa den Server, den Datenbanknamen, den Benutzernamen, das Kennwort oder den Port.

Diese wollen wir ebenfalls in einer Tabelle speichern, die wir **tblVerbindungszeichenfolgen** nennen und die im Entwurf wie in Bild 5 aussieht.

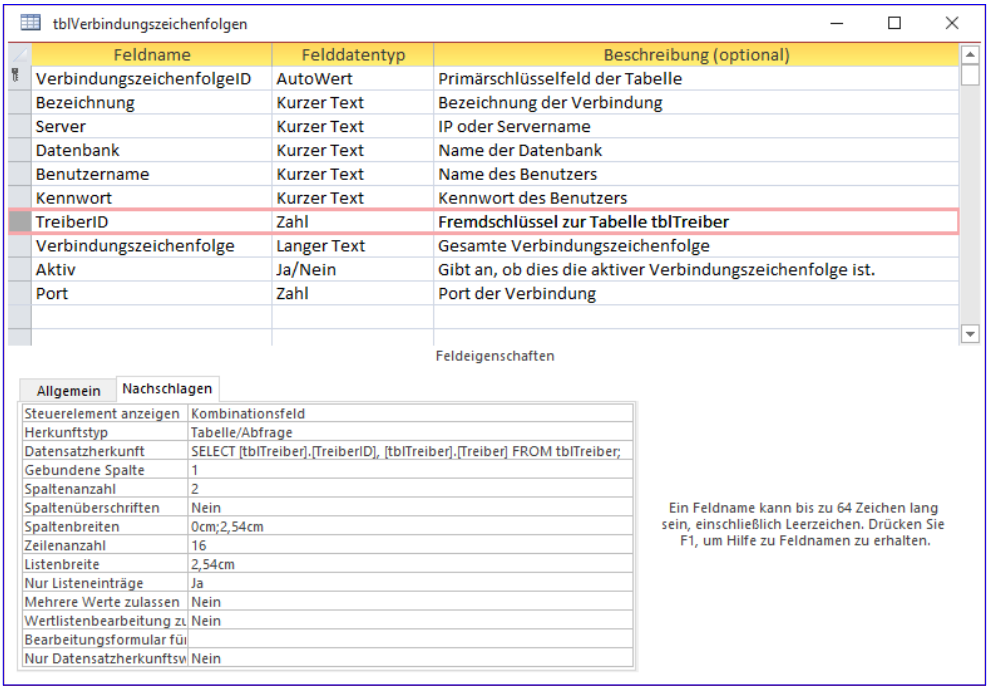

Bild 5: Entwurfsansicht der Tabelle **tblVerbindungszeichenfolgen**

```
        "VerbindungszeichenfolgeID = " _
            & Me!cboSchnellauswahl
End Sub
```

In die meisten der übrigen Steuerelemente gibt der Benutzer die gewünschten Werte einfach ein. Diese sind an die entsprechenden Felder der Datensatzquelle des Formulars gebunden.

Es gibt folgende Ausnahmen:

- **cboDatenbank**: Dient zur Auswahl einer der Datenbanken auf dem Server.
- **cboTreiberID**: Dient zur Auswahl des Treibers aus der Tabelle **tblTreiber**.

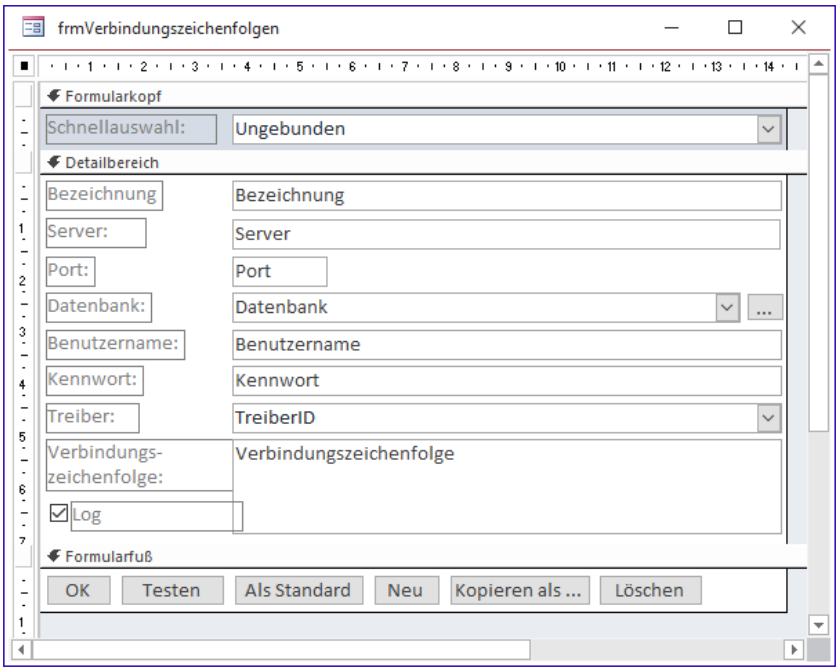

Bild 6: Entwurfsansicht des Formulars **frmVerbindungszeichenfolgen**

Datenbanken einlesen und auswählen

Das Kombinationsfeld **cboDatenbank** soll die Auswahl einer der Datenbanken auf dem im Feld **Server** angegebenen Server ermöglichen.

Dies ist allerdings erst möglich, wenn die Datenbanken eingelesen wurden. Dazu betätigt man die Schaltfläche rechts neben dem Kombinationsfeld.

Die Schaltfläche heißt **cmdDatenbankenEinlesen** und löst die Prozedur aus Listing 1 aus.

Diese fragt ab, ob der Benutzer bereits Daten in die vier Steuerelemente **txtServer**, **cboTreiber**, **txtBenutzername** und **txtKennwort** eingegeben hat.

Fehlen die Daten in einem dieser Felder, erscheint eine Meldung, welches

```
Private Sub cmdDatenbankenEinlesen_Click()
    If Len(Nz(Me!txtServer)) = 0 Then
        MsgBox "Bitte geben Sie den Namen des MySQL-Servers ein."
        Me!txtServer.SetFocus
        Exit Sub
    End If
    If Len(Nz(Me!cboTreiberID)) = 0 Then
        MsgBox "Bitte wählen Sie einen Treiber aus."
        Me!cboTreiberID.SetFocus
        Exit Sub
    End If
    If Len(Nz(Me!txtBenutzername)) = 0 Then
        MsgBox "Bitte geben Sie einen Benutzernamen ein."
        Me!txtBenutzername.SetFocus
        Exit Sub
    End If
    If Len(Nz(Me!txtKennwort)) = 0 Then
        MsgBox "Bitte geben Sie ein Kennwort ein."
        Me!txtKennwort.SetFocus
        Exit Sub
    End If
    Set Me!cboDatenbank.Recordset = DatenbankenEinlesen
End Sub
```

Listing 1: Die Prozedur **cmdDatenbankenEinlesen_Click**

```
Private Function DatenbankenEinlesen() As Recordset
    Dim strSQL As String
    Dim qdf As DAO.QueryDef
    Dim strVerbindungszeichenfolge As String
    strVerbindungszeichenfolge = Me!txtVerbindungszeichenfolge
    strSQL = "SHOW DATABASES"
    Set qdf = QueryDefErstellen(strSQL, strVerbindungszeichenfolge)
    On Error Resume Next
    DoCmd.Hourglass True
    Set DatenbankenEinlesen = qdf.OpenRecordset
    DoCmd.Hourglass False
    If Not Err.Number = 0 Then
        MsgBox "Die Verbindung konnte nicht hergestellt werden:" & vbCrLf & Err.Number & " " & Err.Description
    End If
End Function
```

Listing 2: Die Prozedur **DatenbankenEinlesen**

Feld noch gefüllt werden muss, und das entsprechende Steuerelement erhält den Fokus. Dann wird die Prozedur beendet. Erst, wenn alle Felder gefüllt sind, wird die Funktion **DatenbankenEinlesen** aufgerufen und das Ergebnis der Eigenschaft **Recordset** des Kombinationsfeldes **cboDatenbank** zugewiesen.

Die Funktion **DatenbankenEinlesen** finden Sie in Listing 2. Die Funktion schreibt die Verbindungszeichenfolge, die zuvor mit der Funktion **VerbindungszeichenfolgeDatenbanken** zusammengestellt wurde, in die Variable **strVerbindungszeichenfolge**.

In die Variable **strSQL** füllen wir eine für Access- und SQL Server-gewöhnte Augen etwas ungewöhnliche Abfrage, nämlich die folgende:

```
SHOW DATABASES
```

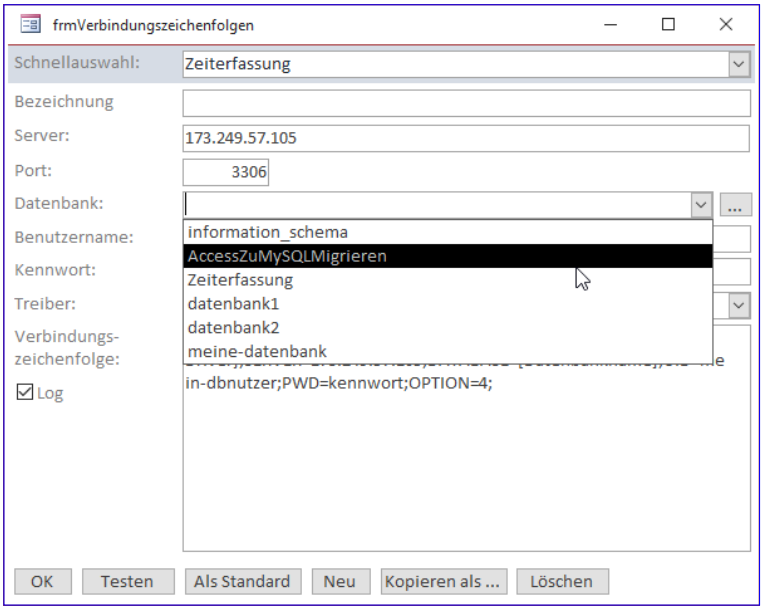

Bild 7: Auswahl der Datenbank

Damit können Sie unter MySQL eine Liste aller Datenbanken des aktuellen Servers ausgeben lassen. Mit **strVerbindungszeichenfolge** und **strSQL** füllen wir dann über die Funktion **QueryDefErstellen** ein neues **QueryDef**-Objekt namens **qdf**.

Dieses führen wir mit der **OpenRecordset**-Methode aus und geben das Ergebnis als Funktionswert zurück.

Die Funktion **VerbindungszeichenfolgeDatenbanken** liefert eine Verbindungszeichenfolge, die aus den Inhalten der Steuerelemente **txtServer**, **cboTreiber**, **txtBenutzername** und **txtKennwort** zusammenstellt wird:

Rechnungen mit ZUGFeRD 1.0, Teil 1: .NET

ZUGFeRD ist das Akronym für »Zentraler User Guide des Forums elektronische Rechnung Deutschland«. Außerdem ist es ein elektronisches Rechnungsdatenformat für den Austausch von Rechnungen und damit ein Lösungsansatz für die Frage, wie man die Daten einer Rechnung auf einfache Weise für die Durchführung der Überweisung des berechneten Betrags verarbeiten kann. Genau genommen nutzt man hier eine Kombination aus einem auf spezielle Weise hergestellten PDF-Dokuments und eines XML-Dokuments, das die in der Rechnung enthaltenen Daten einmal optimiert für das menschliche Auge und einmal optimiert für die Datenverarbeitung enthält. In diesem ersten Teil einer kleinen Beitragsreihe zeigen wir, wie Sie die Open-Source-Bibliothek von Konik in einer .NET-Lösung nutzen, um daraus eine DLL für den Zugriff auf Access heraus zu erstellen.

ZUGFeRD-Basics

Grundidee hinter ZUGFeRD ist es, ein einheitliches Format für den Austausch von Rechnungsdaten zu schaffen, das sowohl vom Menschen als auch vom Computer optimal gelesen werden kann.

Die üblichen Rechnungen sind durchaus so strukturiert, dass der Mensch die Daten für die zu tätigende Überweisung entnehmen kann – wenngleich hier eine weitere Vereinheitlichung bezüglich der Position der verschiedenen Informationen wünschenswert wäre.

Wenn Sie die Rechnung computergestützt verarbeiten wollen, wird es ohne Standardisierung umso schwieriger – Algorithmen brauchen halt genaue Informationen, wo sie was einlesen sollen, damit sie ihre Arbeit fehlerfrei verrichten. Das ist bei gedruckten und eingescannten Rechnungen oder auch bei Rechnungen im PDF-Format nicht zuverlässig möglich.

Also haben sich die Menschen hinter **FeRD**, dem **Forum elektronische Rechnung Deutschland**, etwas ausgedacht: Sie verwenden ein spezielles PDF-Format, nämlich das Format PDF/A-3, mit dem ein PDF-Dokument, das mit herkömmlichen PDF-Readern angezeigt werden kann, um ein XML-Dokument erweitert werden kann. Dieses enthält die gleichen Daten, die auch schon im PDF-Dokument angezeigt werden, nochmal im XML-Format – also in einem Format, das von Computern besser verarbeitet werden kann als ein einfaches PDF-Format.

Die Konik-Bibliothek

Die Firma Konik bietet unter **https://konik.io/** eine Open-Source-Bibliothek für Java und .NET an. Als Access-Entwickler schauen wir erstmal wieder in die Röhre, denn es gibt keine direkte Möglichkeit für uns, direkt auf die Bibliothek zuzugreifen. Wir müssen uns also selbst eine DLL bauen, welche diese Bibliothek nutzt, und auf diese von Access aus zugreifen. Wir werden dies aber in diesem Teil der Beitragsreihe und in den folgenden Teilen detailliert darstellen.

ZUGFeRD-Versionen

Es gibt verschiedene Versionen von ZUGFeRD. Die erste Version 1.0 stammt von 2014. Das ist die Version, die wir im vorliegenden Beitrag beschreiben.

Die neueste Version 2.01 liegt zum Zeitpunkt der Erstellung dieses Beitrags noch nicht in Form einer Open Source-Bibliothek vor.

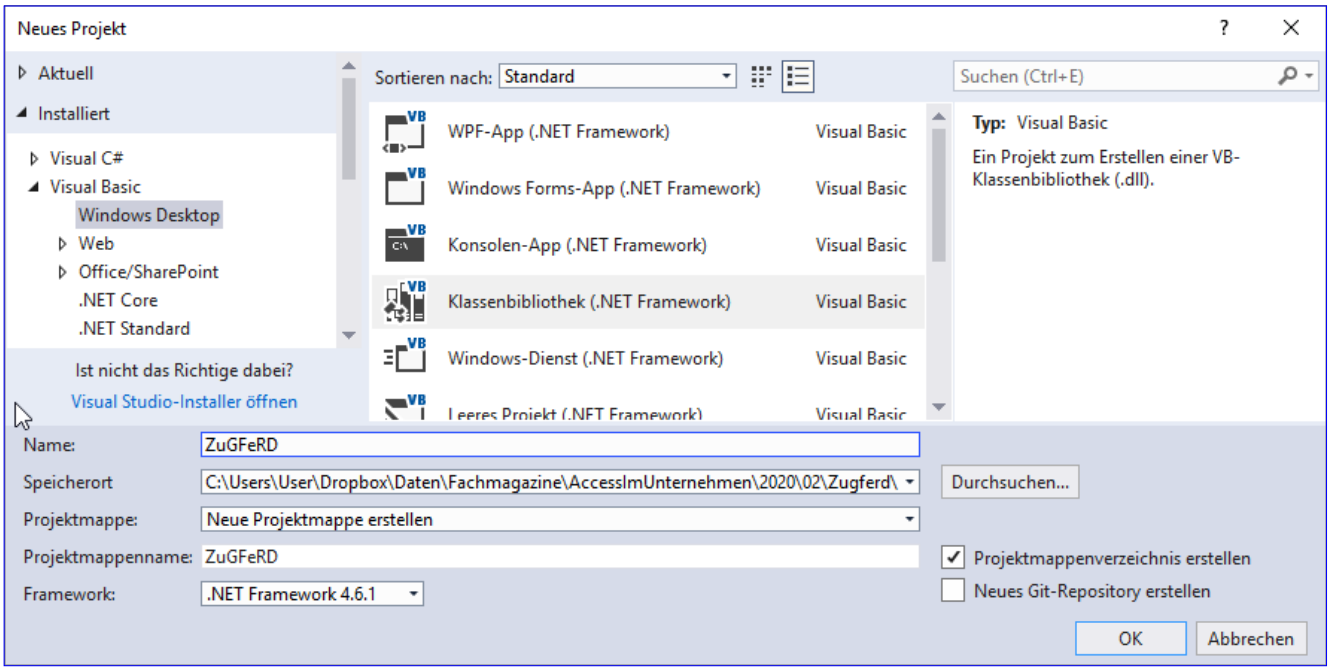

Bild 1: Anlegen der Klassenbibliothek in Visual Studio

Visual Studio installieren

Wenn Sie nicht bereits Visual Studio installiert haben, sollten Sie dies nachholen, sofern Sie die Beispiele dieses Beitrags nachvollziehen wollen. Sie können aber auch die hier erstellte DLL von Access aus verwenden.

Visual Studio können Sie in der aktuellen Version zumindest für private Zwecke und Testzwecke in der Community Edition kostenlos herunterladen und nutzen (weitere Informationen hierzu erhalten Sie in den jeweiligen Lizenzbedingungen).

Visual Studio als Administrator starten

Da die DLL nicht nur erstellt, sondern auch registriert werden muss, benötigen Sie später Administratorrechte. Daher starten Sie Visual Studio gleich mit diesen Rechten.

Dazu klicken Sie mit der rechten Maustaste auf den Eintrag für Visual Studio und wählen dort den Befehl **Als Administrator öffnen** aus. Dass Visual Studio mit Administratorrechten gestartet wurde, erkennen Sie daran, dass in der Titelzeile **Administrator** in Klammern ausgegeben wird.

DLL erstellen

Um eine geeignete DLL zu erstellen, öffnen Sie mit dem Menüpunkt **Datei|Neu|Projekt...** den Dialog **Neues Projekt**. Hier wählen Sie den Namen und den Speicherort des zu erstellenden Projekts aus.

Am wichtigsten ist jedoch die Wahl des richtigen Projekttyps, in diesem Fall **Visual Basic|Windows Desktop|Klassenbibliothek (.NET Framework)**. Vor dem Erstellen des Projekts sieht der Dialog wie in Bild 1 aus.

Vorbereitungen der DLL

Als Erstes benennen wir die automatisch hinzugefügte Klasse **Class1** um, und zwar in **ZUGFeRD_NET**. Das erledigen Sie, indem Sie den entsprechenden Eintrag im Projektmappen-Explorer von Visual Studio ändern.

Wenn Sie das tun, fragt Visual Studio automatisch, ob Sie auch Verweise auf das Codeelement **Class1** umbenennen

wollen, was Sie bejahen sollten. Dadurch wird auch die Klasse entsprechend umbenannt:

```
Public Class ZUGFeRD_NET

End Class
```

Danach erweitern wir den Inhalt der Datei **ZUGFeRD_NET.vb** wie folgt um einen Verweis auf den Namespace **System.Runtime.InteropServices** und einer Zusatzinformation:

```
Imports System.Runtime.InteropServices

<ClassInterface(ClassInterfaceType.AutoDual)>
Public Class ZUGFeRD_NET

End Class
```

Sichtbarmachen der Klassen und Methoden

Damit wir die Klasse und die nachfolgend definierten Methoden im VBA-Editor finden können, müssen wir noch weitere Schritte durchführen.

Dazu klicken Sie doppelt auf den Eintrag **My Project** im Projektmappen-Explorer und zeigen so die Eigenschaften des Projekts an.

Hier wechseln Sie zum Bereich **Anwendung**, sofern dieser noch nicht angezeigt wird, und klicken dort auf die Schaltfläche **Assemblyinformationen...** (siehe Bild 2).

Danach erscheint der Dialog **Assemblyinformationen**. Hier finden Sie im unteren Bereich den Eintrag **Assembly COM-sichtbar machen**, den Sie aktivieren (siehe Bild 3).

Fehlt noch ein Schritt, den Sie in den Eigenschaften im Bereich **Kompilieren** erledigen –

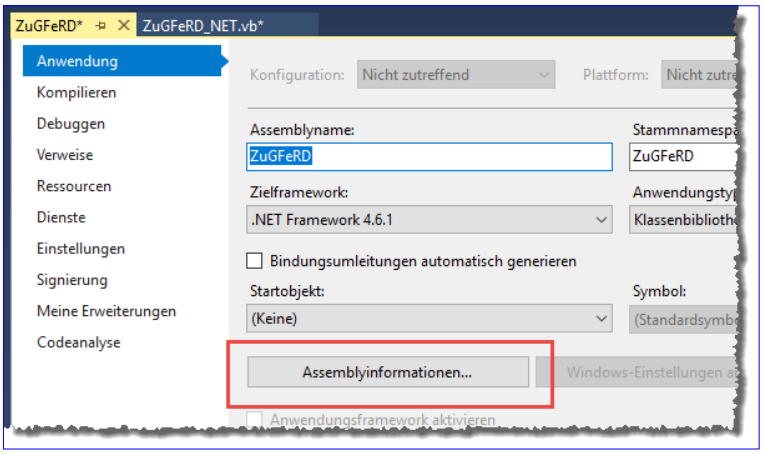

Bild 2: Anzeigen der Assemblyinformationen

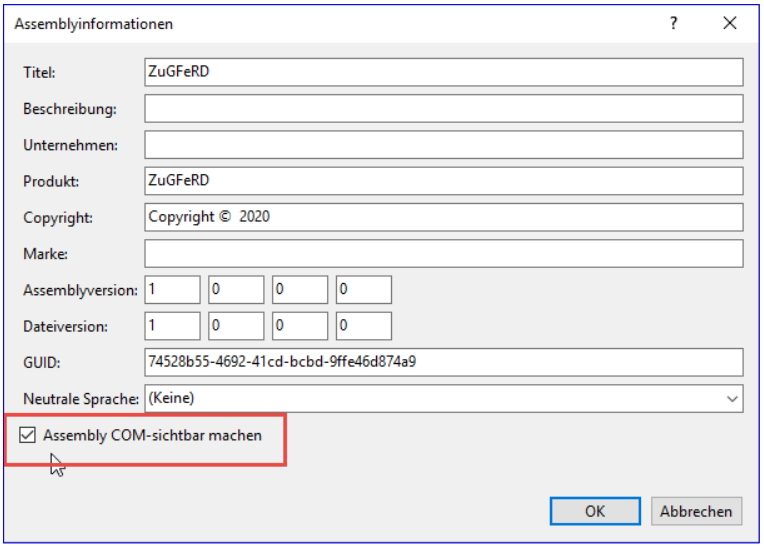

Bild 3: Assembly sichtbar für COM machen

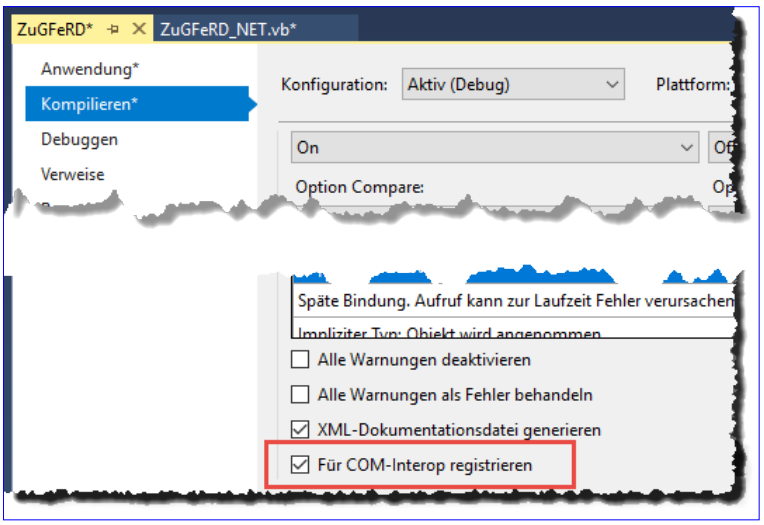

Bild 4: Registrieren für COM-Interop

hier muss die Option **Für COM-Interop registrieren** aktiviert werden (siehe Bild 4).

Schließlich fügen wir der Klasse **ZUGFeRD_NET** die folgende Funktion hinzu:

```
Public Function ZUGFeRDRechnungErstellen() _
                                    As Boolean
    Return True
End Function
```

Damit können wir das Projekt nun erstellen, indem wir den Menübefehl **Erstellen|Projektmappe erstellen** von Visual Studio aufrufen. Dadurch wird die DLL erstellt und die Registry wird um die Informationen ergänzt, die gleich beim Öffnen von Access eingelesen werden, damit dort die neue Bibliothek als Verweis zur Verfügung steht.

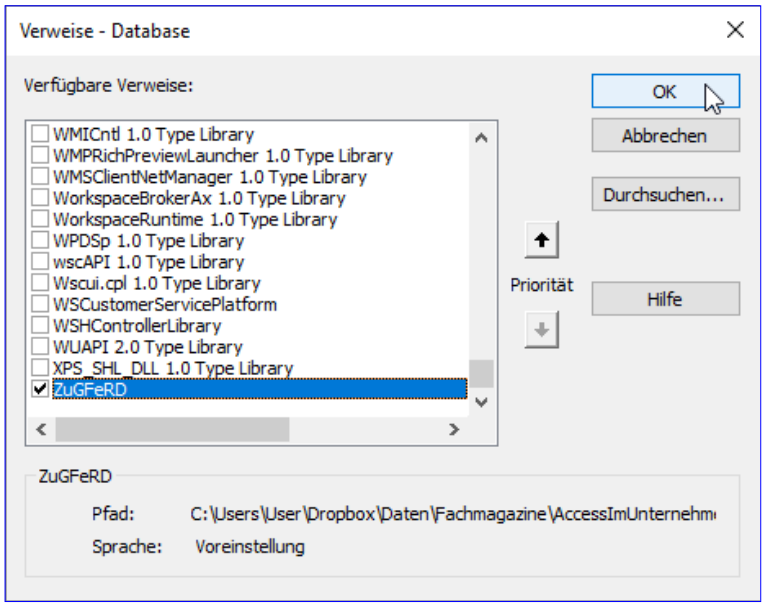

Bild 5: Verweis auf die neue DLL hinzufügen

Verweis auf die neue ZUGFeRD-Bibliothek einbinden

Um den Verweis zu einem VBA-Projekt einer Access-Datenbank hinzuzufügen, öffnen Sie zunächst die Access-Datenbank, in der Sie die DLL verwenden wollen.

Dann zeigen Sie den VBA-Editor an, am schnellsten mit der Tastenkombination **Strg + G**, und öffnen dann mit dem Menüeintrag **Extras|Verweise** den **Verweise**-Dialog.

Hier finden Sie nun ganz unten den Eintrag **ZUGFeRD**, den Sie anhaken, bevor Sie den Dialog wieder schließen (siehe Bild 5).

Wenn dies geschehen ist, können Sie ausprobieren, ob die einzige bisher angelegte Funktion bereits wie erwartet arbeitet. Diese sollte beim Aufruf der folgenden Prozedur den Wert **True** im Direktfenster ausgeben:

```
Public Sub ZUGFeRDTest()
    Dim objZUGFeRD As ZUGFeRD.ZUGFeRD_NET
    Set objZUGFeRD = New ZUGFeRD.ZUGFeRD_NET
    Debug.Print objZUGFeRD.ZUGFeRDRechnungErstellen
End Sub
```

Dies gelingt wie erwartet, sodass wir uns nun der Programmierung zuwenden können.

ZUGFeRD-Paket zum VB-Projekt hinzufügen

Die Open Source-Bibliothek **ZUGFeRD.NET** der Firma Konik kommt als NuGet-Paket. Solche Pakete können Sie einfach zu einem VB-Projekt unter Visual Studio hinzufügen.

Dazu wählen Sie im Kontextmenü des Projekt-Elements im Projektmappen-Explorer unter Visual Studio den Eintrag **NuGet-Pakete verwalten** aus. Dies öffnet den **NuGet-Paket-Manager**. Hier geben Sie als Suchbegriff **ZUGFeRD** ein und wechseln dann zum Bereich **Durchsuchen**.

Wir suchen den Eintrag **ZUGFeRD.NET** der Firma Konik und fügen das entsprechende Paket durch einen Klick auf die Schaltfläche **Installieren** zum aktuellen VB-Projekt hinzu (siehe Bild 6).

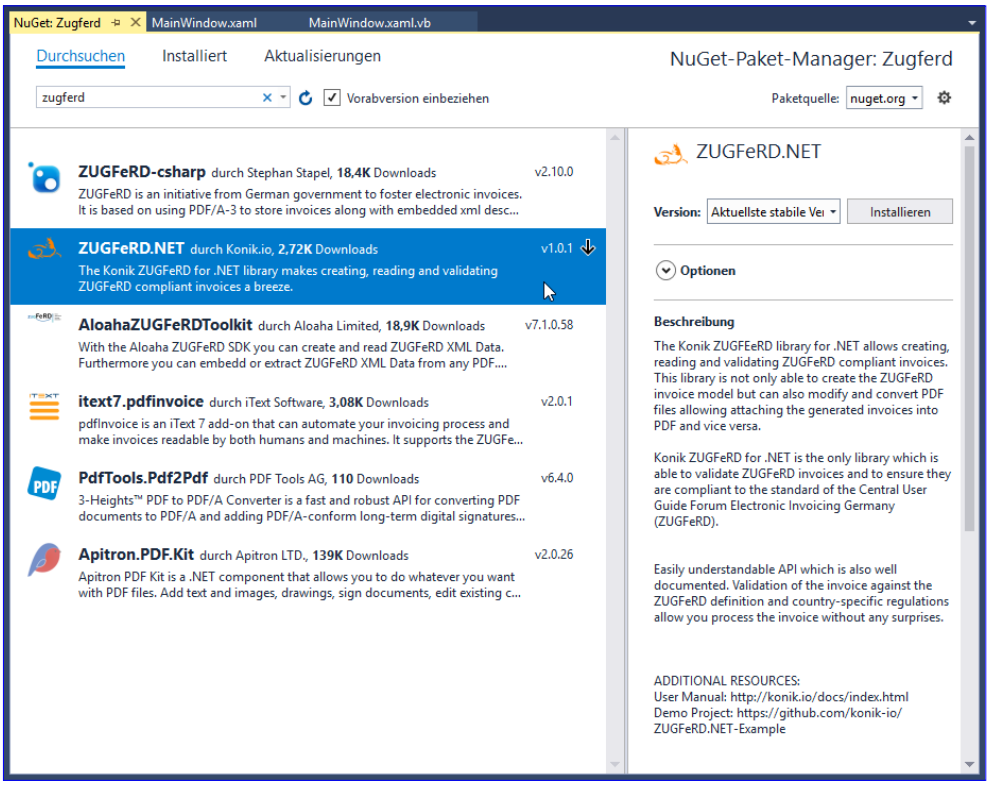

Bild 6: Hinzufügen des NuGet-Pakets

Namespace-Verweise

Im Modul **ZUGFeRD_NET.vb** fügen Sie nun zunächst die folgenden Verweise auf die benötigen Namespaces hinzu:

```
Imports System.Runtime.InteropServices
Imports java.io
Imports org.apache.commons.lang3.time
Imports com.neovisionaries.i18n
Imports io.konik
Imports io.konik.ZUGFeRD
Imports io.konik.ZUGFeRD.entity
Imports io.konik.ZUGFeRD.entity.trade
Imports io.konik.ZUGFeRD.entity.trade.item
Imports io.konik.ZUGFeRD.profile
Imports io.konik.ZUGFeRD.unece.codes
Imports io.konik.ZUGFeRD.unqualified
Imports java.math

Imports Country = com.neovisionaries.i18n.CountryCode
Imports Currency = com.neovisionaries.i18n.CurrencyCode
```

Verschiedene Profile

Bevor wir in die Programmierung einsteigen, noch ein Hinweis auf die verschiedenen Profile für ZUGFeRD-Dateien.

Diese sehen wir folgt aus:

- **BASIC**: Für einfachste Rechnungen. Hier werden nur die relevantesten Daten im XML-Dokument berücksichtigt. Einzelne Rechnungspositionen etwa entfallen. Informationen wie die einzelnen Positionen werden in Textform im PDF-Teil der Rechnung übermittelt.
- **COMFORT**: Bietet mehr Möglichkeiten als das BASIC-Profil, aber nicht alle Möglichkeiten.
- **EXTENDED**: Ermöglicht die vollständige Abbildung des strukturierten Rechnungsaustauschs.

Eine bessere Unterscheidung ermöglicht die Betrachtung der verschiedenen Rechnungsarten.

Das **BASIC**-Profil unterstützt diese Rechnungsarten:

- Handelsrechnung (Rechnung für Waren und Dienstleistungen) mit dem Code **380**
- Bescheide (z. B. Zahlungsaufforderung von Behörden) mit dem Code **380**
- Kaufmännische Gutschrift (zum Beispiel Rechnungskorrektur/Storno) mit negativen Werten (Code **380**)

Das **COMFORT**-Profil bietet folgende zusätzlichen Rechnungsarten:

- Wertbelastung/Wertrechnung ohne Warenbezug (Code **84**)
- Wertgutschrift ohne Warenbezug mit negativen Werten (Code **84**)
- Mit dem **EXTENDED**-Profil können außerdem diese Rechnungsarten verarbeitet werden:
- Selbst ausgestellte Rechnung (Steuerrechtliche Gutschrift/Gutschriftsverfahren, Code **389**)
- Selbst ausgestellte Gutschrift mit negativen Werten (Code **389**)

Die hier genannten Codes werden einer der Eigenschaften zugeordnet, die wir gleich über das Objektmodell der Bibliothek dem XML-Dokument zuweisen, das mit dem PDF-Dokument zusammengeführt werden soll.

Öffentliche Eigenschaften der DLL

Damit wir die im XML-Teil der Rechnung zu berücksichtigenden Informationen von Access aus an die DLL übergeben können, erstellen wir einige öffentliche Eigenschaften.

Deren Programmierung im Modul **ZUGFeRD_NET.vb** besteht aus zwei Teilen – der privaten Deklaration einer Variablen zum Speichern der zugewiesenen Werte sowie einer öffentlichen Eigenschaft mit Getter und Setter.

Die Namen der privaten Variablen beginnen allesamt mit dem Unterstrich und werden wie folgt deklariert:

```
Private _QuellPDF As String
Private _ZielPDF As String
Private _Rechnungsnummer As String
Private _Verkaeufer_Name As String
Private _Verkaeufer_PLZ As String
Private _Verkaeufer_Strasse As String
Private _Verkaeufer_Ort As String
Private _Verkaeufer_Land As CountryCode
Private _Bemerkung As String
Private _Verkaeufer_UstIDNr As String
Private _Kaeufer_Name As String
Private _Kaeufer_PLZ As String
Private _Kaeufer_Strasse As String
Private _Kaeufer_Ort As String
Private _Kaeufer_Land As CountryCode
Private _Kaeufer_UstIDNr As String
Private _Verkaeufer_IBAN As String
Private _Verkaeufer_BIC As String
Private _Nettobetrag As BigDecimal
Private _Aufschlaege As Long
Private _Abschlaege As Long
Private _Mehrwertsteuerbetrag As Long
Private _Bruttobetrag As Long
Private _Zahlungsreferenz As String
Private _Produkt As String
Private _AnzahlProdukt As Long
```

Für jede der Variablen (mit wenigen Ausnahmen) fügen wir eine öffentliche Eigenschaft hinzu, die von außen gelesen und geschrieben werden kann.

Für die Variable **_ZielPDF** sieht dieses Konstrukt wie folgt aus:

```
Public Property ZielPDF As String
    Get
        Return _ZielPDF
    End Get
    Set(value As String)
        _ZielPDF = value
    End Set
End Property
```

Für die übrigen Variablen haben wir ähnliche Konstrukte erstellt. In einigen Fällen benötigen wir alternative Getter und Setter, zum Beispiel für das Land. Hier müssen wir

Rechnungen mit ZUGFeRD 1.0, Teil 2: Access-DLL

Im ersten Teil der Beitragsreihe haben wir gezeigt, wie Sie eine DLL erzeugen können, mit der Sie von außen übergebene Daten plus eine Rechnung im PDF-Format zu einer ZUGFeRD-konformen Rechnung zusammenführen können. Das heißt, dass die PDF-Rechnung in eine Rechnung des Formats PDF/A-3 umgewandelt wird und ein XML-Dokument mit den Rechnungsdaten im computerlesbaren Format in das PDF-Dokument integriert wird. Der zweite Teil der Beitragsreihe zeigt nun, wie Sie die für das XML-Dokument benötigten Informationen an die DLL übergeben und so die ZUGFeRD-Rechnung erstellen.

DLL bereitstellen

Die DLL, die wir im ersten Teil der Beitragsreihe programmiert haben, steht automatisch auf Ihrem Rechner zur Verfügung, wenn Sie das VB-Projekt auf diesem Rechner erstellt haben. Dies führt automatisch zur Registrierung der DLL für die Nutzung von Access aus. Wenn Sie die DLL auf einem anderen Rechner einsetzen wollen, ist nach dem Kopieren der DLL auf den Zielrechner noch ein weiterer Schritt nötig. Sie müssen dann die Eingabeaufforderung mit Administratorrechten starten und den folgenden Befehl eingeben:

```
RegAsm.exe c:\\....\\ZUGFeRD.dll /codebase
```

Die Datei **RegAsm.exe** finden Sie in der Version für 32-Bit im Verzeichnis **C:\\Windows\\Microsoft.NET\\Framework\\v4.0.30319**, für 64-Bit im Verzeichnis **C:\\Windows\\Microsoft.NET\\Framework64\\v4.0.30319**. Achtung: 32-Bit oder 64-Bit hängt davon ab, welche Office-Version die DLL nutzen soll.

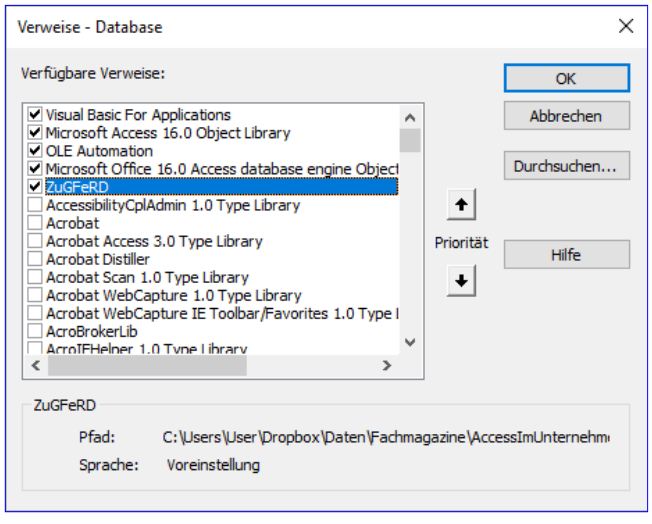

Bild 1: Verweis auf die **ZUGFeRD**-DLL

Verweis hinzufügen

Anschließend können Sie den Verweis über den **Verweise**-Dialog des VBA-Editors wie in Bild 1 hinzufügen.

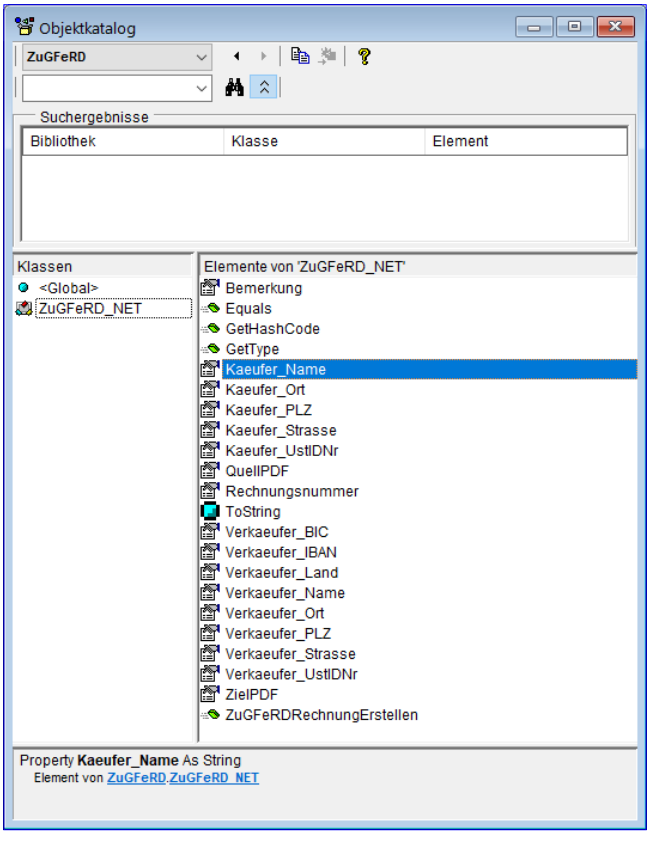

Bild 2: Die **ZUGFeRD**-Befehle im Objektkatalog

Wenn Sie dann den Objektkatalog öffnen und die Bibliothek **ZUGFeRD** auswählen, finden Sie dort alle Befehle vor, die wir im ersten Teil mit der Visual Basic-Lösung in Visual Studio programmiert haben (siehe Bild 2).

Achtung: Verweis erneuern

Wenn Sie die DLL in Visual Studio hinsichtlich der öffentlichen Schnittstelle geändert haben, müssen Sie den Verweis erneuern. Dazu entfernen Sie den Verweis, schließen den **Verweise**-Dialog, öffnen diesen wieder und fügen den Verweis erneut hinzu.

Das Erneuern des Verweises können Sie allerdings auch über eine kleine VBA-Prozedur erledigen. Diese durchläuft die Verweise, bis es einen namens ZUGFeRD findet, merkt sich den Pfad, entfernt den Verweis und füge diesen erneut hinzu:

```
Public Sub VerweisAktualisieren()
    Dim objRef As Reference
    Dim strPath As String
    For Each objRef In References
        If objRef.Name = "ZUGFeRD" Then
            strPath = objRef.FullPath
            References.Remove objRef
            Exit For
        End If
    Next objRef
    References.AddFromFile strPath
End Sub
```

```
Public Sub ZUGFeRDTest()
    Dim objZUGFeRD As ZUGFeRD.ZUGFeRD_NET
    Set objZUGFeRD = New ZUGFeRD.ZUGFeRD_NET
    With objZUGFeRD
        .Bemerkung = "Zahlbar innerhalb von zehn Tagen."
        .Kaeufer_Name = "Klaus Müller"
        .Kaeufer_Ort = "Oberhausen"
        .Kaeufer_PLZ = "46001"
        .Kaeufer_Strasse = "Centro-Allee 1"
'        .Kaeufer_UstIDNr = ""
        .QuellPDF = CurrentProject.Path & "\RechnungOhneXML.pdf"
        .Rechnungsnummer = "12345-1"
        .Verkaeufer_BIC = "COBADEFFXXX"
        .Verkaeufer_IBAN = "DE12121212121212121212"
        .Verkaeufer_Name = "André Minhorst Verlag"
        .Verkaeufer_Land = "DE"
        .Verkaeufer_Ort = "Duisburg"
        .Verkaeufer_PLZ = "47137"
        .Verkaeufer_Strasse = "Borkhofer Str. 17"
        .Verkaeufer_UstIDNr = "DE123123123"
        .Abschlaege = 0
        .AnzahlProdukt = 1
        .Aufschlaege = 0
        .Bruttobetrag = 119
        .Mehrwertsteuerbetrag = 19
        .Nettobetrag = 100
        .Produkt = "Access im Unternehmen"
        .Zahlungsreferenz = "12345-1"
        .ZielPDF = CurrentProject.Path & "\RechnungMitXML.pdf"
    End With
    Debug.Print objZUGFeRD.ZUGFeRDRechnungErstellen
End Sub
```

Listing 1: Erstellen einer ZUGFeRD-konformen Rechnung

PDF-Rechnung in ZUGFeRD-konforme Rechnung umwandeln

Ein Beispiel für eine Prozedur, mit der Sie eine PDF-Rechnung etwa namens **RechungOhneXML.pdf** mit Rechnungsdaten im XML-Format ausstatten, finden Sie in Listing 1. Hier deklarieren wir zunächst ein Objekt des Typs **ZUGFeRD.ZUGFeRD_NET** und erstellen dieses dann.

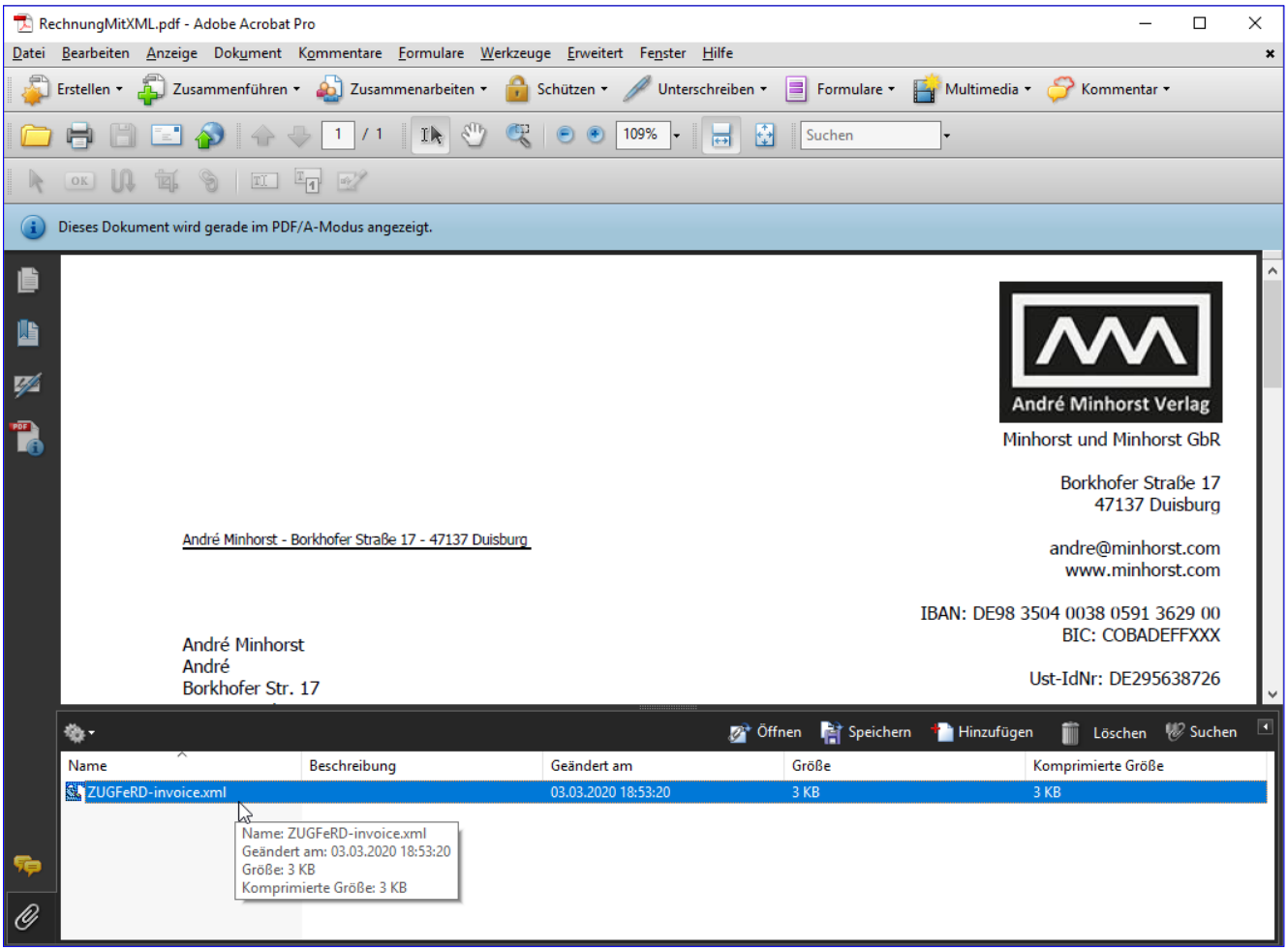

Bild 3: ZUGFeRD-konforme Datei

Danach weisen wir den einzelnen Eigenschaften des Objekts die entsprechenden Werte zu.

Nach dem Starten der Prozedur dauert es wenige Sekunden, bis die Antwort des Aufrufs im Direktbereich des VBA-Editors landet. Danach steht die PDF-Rechnung unter dem angegebenen Pfad bereit und Sie können diese im PDF-Reader öffnen.

PDF untersuchen

Adobe Acrobat liefert dann zunächst eine sichtbare Besonderheit, nämlich dass oben in einer eigenen Zeile der folgende Text erscheint:

```
Dieses Dokument wird gerade im PDF/A-Modus angezeigt.
```

Außerdem entdecken wir links im Navigationsbereich ein Büroklammer-Symbol, das nach dem Anklicken einen eigenen Bereich preisgibt (siehe Bild 3). Dieser sieht wie die Detailansicht im Windows Explorer aus und zeigt die eingebundene Datei namens **ZUGFeRD-invoice.xml** an. Wenn wir diese doppelt anklicken, finden wir das XML-Dokument in der mit der Endung **.xml** verknüpften Anwendung vor. Wir haben den Code formatiert und den ersten Teil in Listing 2 abgebildet. Hier finden wir im oberen Bereich im Element **ram:ID** zunächst die Information, dass es sich um das **BASIC**-Format von ZUGFeRD handelt. Darunter folgen dann die allgemeinen Rechnungsinformationen wie die Rechnungsnummer, der Name, der Type-Code und das Ausgabedatum sowie der Hinweis auf die Zahlungsfrist.