
DIPLOMARBEIT

Herr Ing.
Matthias Thumer

Datenbankarchitektur zur Synchronisierung der Offline-Module

Mittweida, 2014

DIPLOMARBEIT

Datenbankarchitektur zur Synchronisierung der Offline-Module

Autor:

Herr Ing. Matthias Thumer

Studiengang:

Informationstechnik

Seminargruppe:

KI09wIA

Erstprüfer:

Prof. Dr.-Ing. Rolf Hiersemann

Zweitprüfer:

Dipl.-Ing. (FH) Roman Riegler

Einreichung:

Mittweida, 20.05.2014

Verteidigung/Bewertung:

Mittweida, 2014

Bibliografische Angaben:

Thumer, Matthias:

Datenbankarchitektur zur Synchronisierung der Offline-Module – Erweiterung der firmeninternen Verwaltungssoftware um ein XML-Interface und Beschreibung der dazu nötigen Datenbankarchitektur –2014. – vii, 53 S.
Mittweida, Hochschule Mittweida (FH), University of Applied Sciences,
Fakultät Elektro- und Informationstechnik, Diplomarbeit, 2014

Referat:

Die vorliegende Arbeit befasst sich mit der Datenbankarchitektur zur Implementierung der Synchronisierungsschnittstelle. Als Hauptziel sollen die Funktionalitäten einer firmeninternen Verwaltungssoftware mithilfe des Synchronisierungs-Interfaces auch den Offline-Modulen bereitgestellt werden können.

Durch die simultane Benutzung der internen Verwaltungssoftware und der Offline-Modulen muss es auch möglich sein einen bidirektionalen Datenaustausch über die Synchronisierungsschnittstelle zu machen. Zur Durchführung der Synchronisierung muss auch ein beidseitiges validiertes Zertifikat zur Kommunikation berücksichtigt werden. Zusätzlich soll über eine integrierte Benutzer-Authentifizierung der Datenaustausch für die entsprechenden Offline-Module eingeschränkt werden.

Inhalt

Inhalt.....	i
Abbildungsverzeichnis.....	iii
Tabellenverzeichnis.....	v
Abkürzungsverzeichnis.....	vi
0 Übersicht.....	1
0.1 Motivation	1
0.2 Zielsetzung	1
0.3 Kapitelübersicht	2
1 Grundlagen und Stand der Technik	3
1.1 Webservice.....	3
1.2 XML.....	4
1.2.1 XML-Syntax.....	4
1.3 Datenbanken	5
1.3.1 Datenbankarchitektur.....	6
1.3.2 Datenbankmodelle.....	8
1.3.2.1 Relationales Datenbankmodell	11
1.3.3 SQL	15
2 Einblick und Aufgabenstellung	19
2.1 Die Verwaltungssoftware	19
2.2 Die Offline-Module	22
2.2.1 Bank Account Manager	22
2.3 Aufgabenstellung.....	26
3 Systemkonzept	29
3.1 Konzept der Offline-Module	29
3.2 Installation bzw. Updatevorgang der Offline-Module	30
3.3 XML-Schnittstelle.....	31
3.4 Datenbankkonzept.....	33

4	Implementierung.....	36
4.1	<i>Erweiterung der MySQL-Datenbank</i>	36
4.1.1	Hinzufügen der nötigen Metadaten.....	38
4.1.2	Ergänzung der bestehenden Datensätze.....	40
4.2	<i>XML-Synchronisierungsschnittstelle</i>	41
4.2.1	Erstellen des PHP-Skripts	42
4.2.2	Implementierung in die Offline-Module	45
5	Ergebnisse und Ausblick.....	49
5.1	<i>Ergebnisse</i>	49
5.2	<i>Bewertung der Arbeit.....</i>	50
5.3	<i>Ausblick.....</i>	50
	Literatur.....	52
	Eidesstattliche Erklärung.....	53

Abbildungsverzeichnis

Abbildung 1-1: XML-Deklaration	4
Abbildung 1-2: XML-Beispiel.....	5
Abbildung 1-3: Nutzdaten und Metadaten	5
Abbildung 1-4: Zusammenspiel von DBMS und Datenbank.....	7
Abbildung 1-5: Ebenen des DBMS	8
Abbildung 1-6: Hierarchisches Datenbankmodell.....	9
Abbildung 1-7: Netzwerkdatenbankmodell	9
Abbildung 1-8: Objektorientierte Datenbankmodell	10
Abbildung 1-9: Dokumentenorientiertes Datenbankmodell.....	11
Abbildung 1-10: Beziehungstyp 1:1	12
Abbildung 1-11: Beziehungstyp 1:n	13
Abbildung 1-12: Beziehungstyp n:m.....	14
Abbildung 2-1: Ausschnitt vom Rechnungsbereich	20
Abbildung 2-2: Ausschnitt von den Offenen-Posten	21
Abbildung 2-3: Ausschnitt vom Bankkontenbereich	22
Abbildung 2-4: Oberfläche des Moduls Bank Account Manager.....	23
Abbildung 2-5: Anzeigemöglichkeiten	24
Abbildung 2-6: Import neuer Datensätze.....	24
Abbildung 2-7: Datensätze löschen	25
Abbildung 2-8: Bankkonto Einstellungen.....	25

Abbildung 3-1: Architektur der Verwaltungssoftware mit Offline-Module	30
Abbildung 3-2: Aufbau der zu generierenden XML-Datei	32
Abbildung 3-3: Graphische Darstellung des Datenaustausches.....	33
Abbildung 4-1: Übersicht von MySQL Workbench	36
Abbildung 4-2: Übersicht der Beziehungen wichtiger Tabellen.....	37
Abbildung 4-3: Tabellenentwurf mit Microsoft Access	39
Abbildung 4-4: Tabellenerweiterung in Microsoft Access	40
Abbildung 4-5: Darstellung von einem Assoziierten-Array.....	43
Abbildung 4-6: Anzeige vom letzten Import bzw. Export Vorgang	47
Abbildung 4-7: Fortschrittsanzeige mit dem Steuerelement ProgegssBar	48

Tabellenverzeichnis

Tabelle 1-1: Beispiel für Nutzdaten	7
Tabelle 1-2: Beispiel für Metadaten	7
Tabelle 1-3: Beispiel Tabelle für SQL-Befehle	15
Tabelle 3-1: Struktur der Tabelle tblxmlbericht.....	34

Abkürzungsverzeichnis

.NET	Dot NET Framework
ADO.NET	ActiveX Data Objects
C#	C Sharp
CSV	Comma Separated Values
DBMS	Datenbankmanagementsystem
DBS	Datenbanksystem
DDL	Data Definition Language
DML	Data Manipulation Language
GUI	Graphical User Interface
GPL	General Public License
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
ISO	International Organization for Standardization
LAN	Local Area Network
MD5	Message-Digest Algorithm 5
OQL	Object Query Language
ODL	Object Definition Language
ODBC	Open Database Connectivity
OLE DB	Object Linking and Embedding Database
PHP	Hypertext Preprocessor
PL/SQL	Procedural Language / Structured Query Language
SMTP	Simple Mail Transfer Protocol
SQL	Structured Query Language
SSL	Secure Sockets Layer

TCP/IP	Transmission Control Protocol/Internet Protocol
T/SQL	Transact Structured Query Language
W3C	World Wide Web Consortium
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
UTF-8	Universal Character Set Transformation Format 8-bit
VPN	Virtual Private Network
XML	eXtensible Markup Language
XSD	XML Schema Definition

0 Übersicht

Dieses Kapitel zeigt die Beweggründe bzw. Motivation zu diesem Thema und erläutert die Aufgabenstellung der Diplomarbeit. Gleichzeitig erfolgt ein kurzer Überblick zu den einzelnen Kapiteln.

0.1 Motivation

In der heutigen Zeit ist die Mobilität eines der größten Herausforderungen für die Informationstechnologie. Vor allem wenn es sich um unternehmensspezifische Anwendungen handelt. Um diese Herausforderungen zu lösen sind mittlerweile zahlreiche Lösungsansätze in Verwendung, also verschiedene Arten wie die benötigten Daten zur Verfügung gestellt werden.

Die Implementierung von sogenannten Synchronisierungsschnittstellen ist jedoch keine triviale Angelegenheit. Gründe wie unterschiedliche Betriebssysteme, Programmiersprachen und Datenbanken machen das Thema anspruchsvoll. Dabei ist vor allem zu achten und zu gewährleisten, dass durch das Synchronisierungs-Interface keine Daten beschädigt beziehungsweise manipuliert werden können.

0.2 Zielsetzung

Die vorliegende Arbeit befasst sich mit der Datenbankarchitektur zur Implementierung der Synchronisierungsschnittstelle. Dazu werden technische Grundlagen vorgestellt und zusätzlich ein Überblick über die aktuellen Softwarewerkzeuge gewährt.

Das Hauptziel ist aber, dass die Funktionalitäten einer firmeninternen Verwaltungssoftware mithilfe des Synchronisierungs-Interfaces und den Offline-Modulen bereitgestellt werden können. Durch die simultane Benutzung der internen Verwaltungssoftware und der Offline-Modulen muss es auch möglich sein einen bidirektionalen Datenaustausch über die Synchronisierungsschnittstelle zu betreiben.

Da der Betrieb der Offline-Module weitgehend außerhalb eines Intranets stattfinden wird soll beim Design der Anwendung und der Datenbankarchitektur ein möglichst einfaches Updateverfahren der Module eingeplant werden, um alle Offline-Module die in Verwendung sind am gleichen Stand zu halten.

Zur Durchführung der Synchronisierung muss auch ein beidseitiges validiertes Zertifikat zur Kommunikation berücksichtigt werden. Zusätzlich soll über eine integrierte Benutzer-Authentifizierung der Datenaustausch für die entsprechenden Offline-Module eingeschränkt werden.

0.3 Kapitelübersicht

Die Diplomarbeit besteht aus sechs Kapiteln.

Nach der allgemeinen Einleitung werden im **Kapitel 1** die Grundlagen und das Elementarwissen erläutert. Dieses Wissen soll die Diplomarbeit allgemein verständlicher machen.

Im **Kapitel 2** werden zum besseren Verständnis die Verwaltungssoftware und die Offline-Module vorgestellt sowie die grundlegenden Funktionen und die Aufgabenstellung präzisiert.

Anschließend wird im **Kapitel 3** das Systemkonzept für das Synchronisierungs-Interface und die Datenbankarchitektur für die Offline-Module sowie die Datenbank in der Verwaltungssoftware dargestellt.

Im **Kapitel 4** erfolgt die Umsetzung der Anforderungen. Dazu wird zuerst die XML-synchronisierungsschnittstelle implementiert und die nötigen Datenbankerweiterungen durchgeführt.

Abschließend werden im **Kapitel 5** die Resultate der Diplomarbeit noch einmal zusammengefasst. Hier werden auch die Leistungen des Diplomanden aus seiner Sicht skizziert und ein Ausblick auf mögliche Erweiterungen bzw. Verbesserungen gegeben.

1 Grundlagen und Stand der Technik

Es gibt viele verschiedene Definitionen in der Literatur. Jeder Entwickler bzw. Anwender hat möglicherweise eine etwas andere Definitionen und technisches Grundlagenwissen. Deswegen ist es von großer Bedeutung diese bewährten Technologien zu definieren bzw. zu beschreiben. So gesehen wird in diesem Abschnitt versucht diese Grundlagen zu definieren bzw. die wichtigsten Begriffe zu erläutern und beschreiben sowie auch bestimmte Architekturmodelle. Dieses Elementarwissen soll es ermöglichen die Diplomarbeit allgemein verständlicher zu machen bzw. als Grundlage dienen.

1.1 Webservice

Nach dem W3C¹ (World Wide Web Consortium) lautet die Definition:

“A Web Service is a software system identified by a URI² (Uniform Resource Identifier), whose public interfaces and bindings are defined and described using XML³ (eXtensible Markup Language). It's definition can be discovered by other software systems. These systems may then interact with the Web Service in a manner prescribed by it's definition, using XML based messages conveyed by Internet protocols.”

Daraus ergibt sich nach eigener Interpretation:

Ein Webservice ist ein Softwaresystem das durch ein URI identifiziert wird. Dessen öffentliche Schnittstellen und Bindungen mittels XML definiert und beschrieben werden. Seine Definition kann auch durch andere Software-Systeme entdeckt werden. Diese Systeme können dann über die vorher deklarierte Weise miteinander intera-

¹ W3C ist ein Konsortium zum Einwickeln von Web-Standards (siehe <http://www.w3.org/TR/wsa-reqs/>)

² Ein URI dient zur Identifizierung von abstrakten oder physischen Ressourcen (siehe <http://www.w3.org/Addressing/>)

³ XML ist eine Auszeichnungssprache

gieren unter Nutzung von XML-basierten Nachrichten über spezifische Internet-Protokolle wie HTTP⁴ (Hypertext Transfer Protocol), HTTPS⁵ (Hypertext Transfer Protocol Secure) oder SMTP⁶ (Simple Mail Transfer Protocol).

1.2 XML

Die Auszeichnungssprache XML wird zur Darstellung hierarchisch strukturierter Daten in Form von Textdaten verwendet. Es wird unter anderem für den plattform- und implementierungsunabhängigen Austausch von Daten zwischen Computersystemen eingesetzt, insbesondere über das Internet. Da diese Sprache sehr flexible ist und auch extrem verbreitet, gibt es fast in jeder Programmiersprache spezielle Funktionen zur Bearbeitung und Erstellung von XML-Dateien. Im Allgemeinen ist XML ein vom W3C eingeführter Standard zur Darstellung von Daten.

1.2.1 XML-Syntax

Wie in [HaMe2001] beschrieben, beginnt ein XML-Dokument nach W3C-Standard mit einem Kopf und dann kommen die eigentlichen Daten. Der Kopf sollte die XML-Deklaration enthalten in der Auskunft über die XML-Version und die Zeichencodierung des XML-Dokuments geben. Die folgende Abbildung zeigt einen Beispielkopf mit XML-Version und Zeichencodierung.

```
<?xml version="1.0" encoding="UTF-8"?>
```

Abbildung 1-1: XML-Deklaration

Eine XML-Datei wird mit sogenannten Strukturelementen zusammengesetzt. Die Namen für diese Elemente lassen sich frei wählen und können ganz unterschiedliche Daten enthalten und beschreiben.

Nach [HaMe2001] bestehen XML-Dokumente im Wesentlichen aus einer Aneinanderreihung von Wurzelementen mit Unterelementen, die die Nutzinformation beinhalten. Diese Elemente sind in spitzen Klammern eingefasst und beschreiben die Bedeutung der im nachfolgenden Text eingeschlossenen Daten.

⁴ HTTP ist ein Protokoll zur Übertragung von Daten (siehe <http://www.w3.org/Protocols/>)

⁵ HTTPS ist auch ein Protokoll und die Erweiterung von HTTP zur Übertragung verschlüsselter Daten

⁶ SMTP ist ein Protokoll zum Austausch von E-Mails in Computernetzen (siehe <http://www.ietf.org/rfc/rfc2821.txt>)

Zusätzlich können noch sogenannter Attribute einem Element zugeordnet werden wie zum Beispiel der Datentyp eines Elementes. Diese werden bei einem Wurzelement oder Unterelement hinzugefügt und per Name und Wert identifiziert.

Das nächste Listing (Abbildung 1-2) zeigt ein Beispiel eines XML-Dokuments, das eine Nachricht mit Datum und Text darstellt. Beim Element `note` gibt es zusätzlich das Attribut `id`, welches eine Nummerierung der Nachrichten darstellt. Zusätzlich ist beim Element `sent` das Attribut `type`, das den Datentyp beschreibt.

```
<?xml version="1.0" encoding="UTF-8"?>
<messages>
  <note id="1">
    <sent type="date">2013-01-01 12:05:32</sent>
    <body>Das ist der Text der Nachricht</body>
  </note>
</messages>
```

Abbildung 1-2: XML-Beispiel

1.3 Datenbanken

Wie auch bei anderen Begriffsdefinitionen gibt es auch für Datenbanken sehr viele Definitionen, aber nach [GeFr2011] lautet eine eher technische Definition:

„Eine Datenbank ist ein verteiltes, integriertes Computersystem, das Nutzdaten und Metadaten enthält. Nutzdaten sind die Daten, die Benutzer in der Datenbank anlegen und aus denen die Informationen gewonnen werden. Metadaten werden oft auch als Daten über Daten bezeichnet und helfen, die Nutzdaten der Datenbank zu strukturieren.“

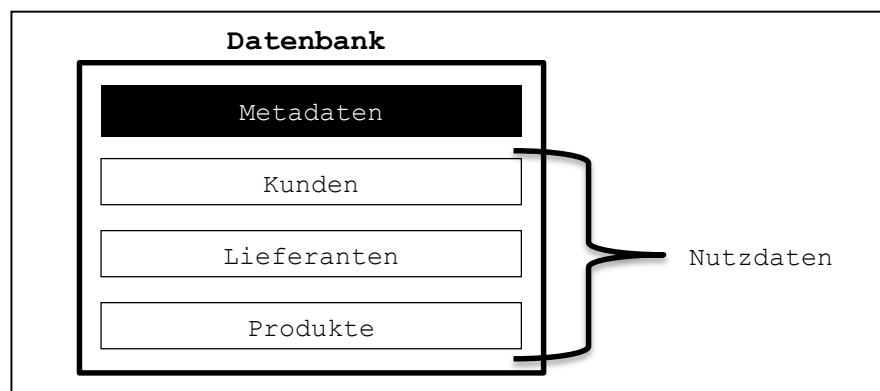


Abbildung 1-3: Nutzdaten und Metadaten

Diese Definition baut auf die Realisierung einer Datenbank als Computersystem. Dabei könnte man auch die Definition etwas theoretischer halten, da es Datenbanken nicht nur seit es Computer gibt existieren, sondern schon lange zuvor. Wie in [GeFr2011] beschrieben lautet eine allgemeinere Definition:

„Eine Datenbank ist eine geordnete, selbstbeschreibende Sammlung von Daten, die miteinander in Beziehung stehen.“

Mit beiden Definitionen bekommt man meiner Meinung nach ein sehr gutes Verständnis was man prinzipiell unter Datenbanken versteht und die Definitionen ergänzen sich hervorragend.

In dieser Arbeit werden wir uns im Wesentlichen auf Datenbanken als Computersystem konzentrieren bzw. einen genaueren Blick darauf werfen.

Vereinfacht könnte man auch sagen, dass eine Datenbank eine Sammlung von Daten ist, die nach bestimmten Regeln gespeichert werden.

Zusätzlich erfüllen Datenbanken noch weitere Funktionen, wie zum Beispiel Datensicherheit, Datenintegrität, Anfrageoptimierung oder Mehrbenutzerfähigkeit.

1.3.1 Datenbankarchitektur

Eine Datenbank oder auch DBS⁷ (Datenbanksystem) besteht aus zwei Teilen. Das System das sich um die Organisation der Daten kümmert und die Zugriffsrechte der Daten verwaltet wird in der Regel DBMS⁸ (Datenbankmanagementsystem) genannt und der Menge der zu verwaltenden Daten, der Datenbank im engeren Sinn.

Das Datenbankmanagementsystem kann bei Desktop-Datenbanksystemen aus einem einzelnen Programm bestehen wie zum Beispiel bei Microsoft Access der Fall ist oder aus mehreren zusammenarbeitenden Programmen. Diese Variante ist sehr verbreitet bei größeren server-gestützten Datenbanken. In Abbildung 1-4 ist der Zusammenhang zwischen DBMS, Metadaten und Nutzdaten zu sehen. Ein Benutzer stellt eine Abfrage an das Datenbankmanagementsystem, die dann die Daten aus der eigentlichen Datenbank heraussucht und dem Anwender zurückliefert.

⁷ DBS oder Datenbank ist ein System zur Datenverwaltung

⁸ DBMS ist die Verwaltungssoftware für die Datenbank und verantwortlich für Zugriffsrechte und die Organisation der Daten

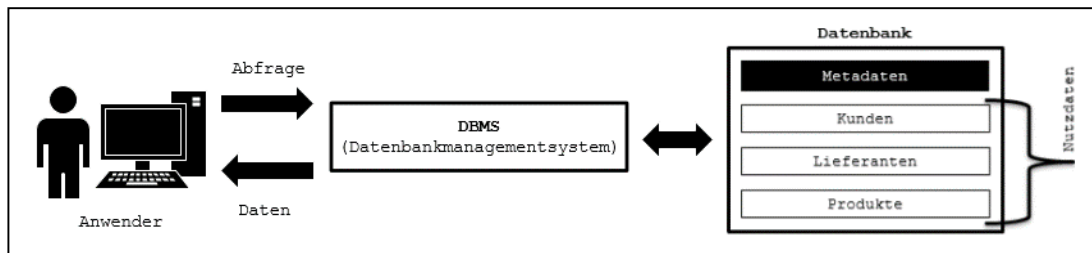


Abbildung 1-4: Zusammenspiel von DBMS und Datenbank

Zusätzlich kann man in der Abbildung 1-4 sehen, dass die Datenbank aus Metadaten und Nutzdaten besteht, wobei die Nutzdaten aus den Daten in den Tabellen Kunden, Lieferanten und Produkte bestehen. Anhand der Tabelle 1-1 mit der Tabelle *Kunde* wird dargestellt, dass die Nutzdaten der Inhalt der Tabelle ist, also die Datensätze der einzelnen Kunden.

Tabelle: Kunde

<i>KdNr</i>	<i>Vorname</i>	<i>Nachname</i>	<i>KundeSeit</i>
1	Matthias	Thumer	20/03/2013
2	Max	Mustermann	05/03/2013
3	John	Smith	12/12/2012

Tabelle 1-1: Beispiel für Nutzdaten

Bei den Metadaten handelt es sich nicht um die Daten in den Tabellen, sondern um die Struktur der Tabellen bzw. um den *Datentype* oder *Spaltennamen* wie man in der unteren Tabelle *Kunde* (Tabelle 1-2) sieht.

Tabelle: Kunde

<i>Spaltenname</i>	<i>Datentype</i>
KdNr	Integer
Vorname	Varchar(30)
Nachname	Varchar(30)
KundeSeit	Date

Tabelle 1-2: Beispiel für Metadaten

Betrachtet man das DBMS genauer, besteht dieses aus drei grundlegenden Ebenen wie man in Abbildung 1-5 sieht. Die unterste Ebene bildet die sogenannte Interne-Ebene in der alle systemspezifischen Merkmale des Datenbankmanagementsystems definiert sind. Diese Interne-Ebene ist weitgehend abhängig von dem jeweiligen Betriebssystem auf dem das Datenbanksystem läuft, wie zum Beispiel die Speicherstruktur und die verfügbaren Sprachschnittstellen.

Die Konzeptuelle-Ebene ist im Gegensatz zur internen Ebene vollständig unabhängig von dem verwendeten Betriebssystem. Diese Ebene stellt nur eine ausführliche Beschreibung der zugrunde liegenden Datenstruktur zur Verfügung, aber keine Informationen über die Implementierungsdetails der Datenbank.

In der letzten Ebene können die Anwender des Datenbankmanagementsystems zu greifen. Diese sogenannte Externe-Ebene gibt den Benutzern die Möglichkeit interaktiv den Datenbestand zu verwalten. Durch diese Aufbauweise kann man gewährleisten, dass der Anwender nichts an der Struktur und internen Implementierung der Datenbank modifizieren kann.

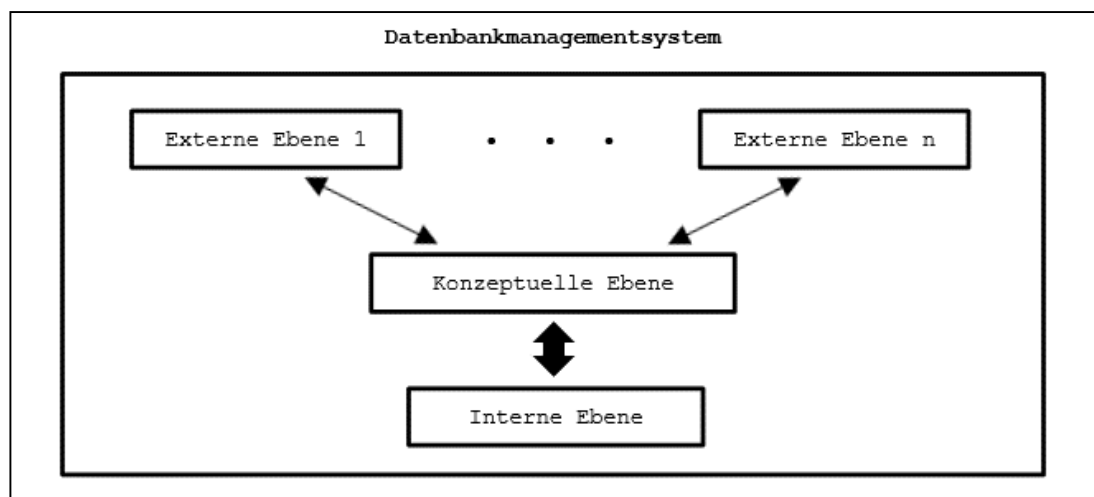


Abbildung 1-5: Ebenen des DBMS

1.3.2 Datenbankmodelle

In Laufe der Zeit sind viele verschiedene Datenbankmodelle entstanden. Wie nach [EISh2005] gibt es eine Vielzahl die sich auch miteinander vermischen haben, aber grundsätzlich kann man folgende aufzählen:

1. Hierarchisches Datenbankmodell
2. Netzwerkdatenbankmodell
3. Objektrelationales Datenbankmodell
4. Objektorientiertes Datenbankmodell
5. Dokumentenorientiertes Datenbankmodell
6. Relationales Datenbankmodell

Letzteres Modell, also das Relationale ist derzeit das am weitesten verbreitete und deswegen gehen wir auf dieses später noch genauer ein.

Das Netzwerkmodell und das hierarchische Datenbankmodell kann man auch als historische Modelle bezeichnen, aber destotrotz gibt es sie noch und sie werden immer noch eingesetzt. Beim Netzwerkmodell spricht man auch von keinem strengen hierarchischen Schema und deswegen wird es auch als Erweiterung des hierarchischen Datenbankmodells angeführt. Zur besseren Veranschaulichung wird in Abbildung 1-6 und 1-7 jeweils ein Beispiel dargestellt.

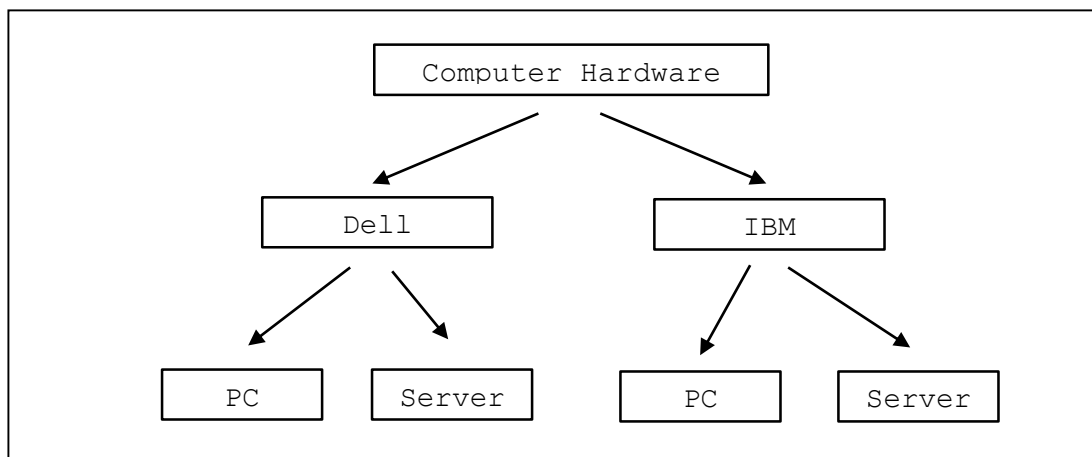


Abbildung 1-6: Hierarchisches Datenbankmodell

Wie man in Abbildung 1-6 sieht gibt es bei der auch genannten Baumstruktur nur die Möglichkeit einen Vorgänger per Eintrag zu haben mit der Ausnahme der Wurzel `Computer Hardware`. Dadurch kann es zu Redundanzen kommen in der Struktur.

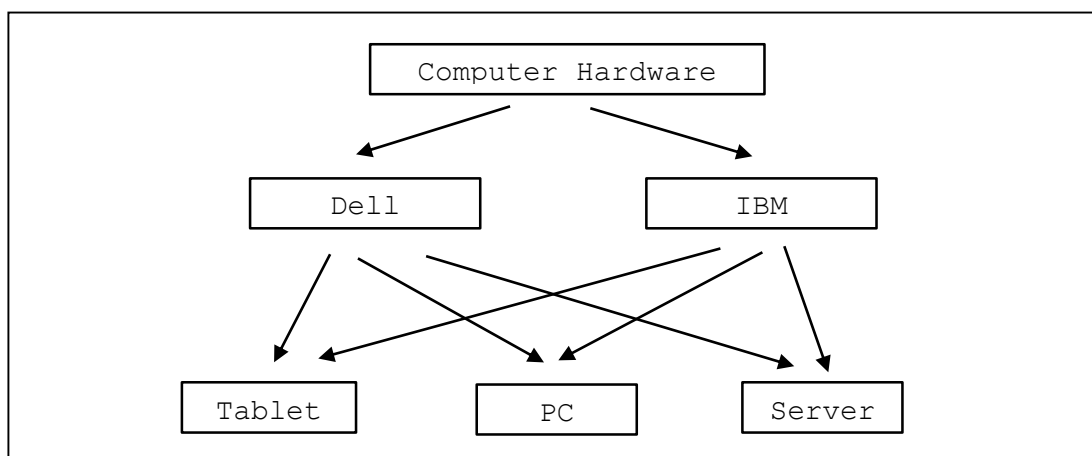


Abbildung 1-7: Netzwerkdatenbankmodell

Bei Abbildung 1-7 kann man sehen, wenn man diese Abbildung mit dem hierarchischen Datenbankmodell vergleicht, dass die Redundanzen vermieden werden können.

Das Netzwerkmodell wird hauptsächlich auf Großrechnern verwendet wobei es zunehmend verdrängt wurde vom relationalen Datenbankmodell, seit dem es vorgestellt wurde. Zusätzlich sieht man aber auch Mischformen dieser beiden Modelle auf Großrechnern in Verwendung.

Bei dem objektorientierten und objektrelationalen Datenbankmodell basieren Daten auf Objekte wie im Sinne der objektorientierten Programmiersprachen. Damit ist gemeint, dass die Objekte auf Begriffe der realen Welt basieren und Strukturen sowie Verhalten möglichst 1:1 erfassen. So gehört zum Beispiel die Größe und Anzahl der Zimmer einer Wohnung zum Objekt *Wohnung*. Die Attribute des Objekts *Wohnung* würden dieses Objekt dann näher beschreiben.

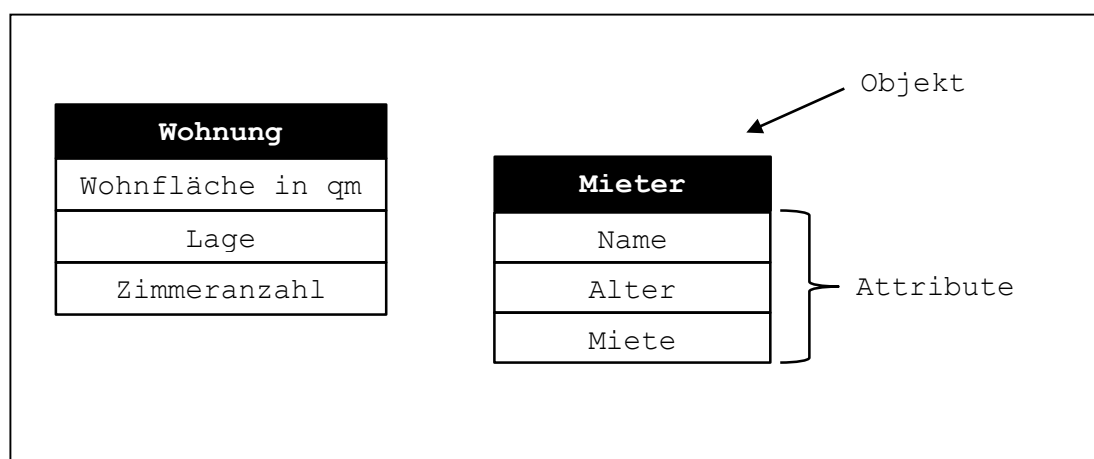


Abbildung 1-8: Objektorientierte Datenbankmodell

Das objektrelationale Datenbankmodell unterstützt den Anwender bei der Anfrage komplexer Strukturen, da diese Art der Datenbank die semantischen Zusammenhänge zwischen den Objekten bekannt sind. Deswegen gibt es auch eine entsprechende Abfragesprache genannt OQL⁹ (Object Query Language), der es sogar möglich ist ganze Objekte als Ergebnis einer Abfrage zurückzugeben. Als Datenmanipulationssprache wird die ODL¹⁰ (Object Definition Language) verwendet.

⁹ OQL ist die Abfragesprache für Objektdatenbanksysteme

¹⁰ ODL wird als Datenmanipulationssprache für Objektdatenbanksysteme verwendet

Die dokumentenorientierten oder auch sogenannten dokumentenbasierten Datenbanken speichern die Daten in Dateien ab. Dabei werden die Daten in Schlüsselfeldern mit dem entsprechend dazugehörigen Wert gespeichert. Die einzelnen Dateien haben keine Relationen zu den anderen Dateien und sind dabei als individuelle oder geschlossene Einheit zu betrachten. Deswegen kann auch das Schema jeder Datei unterschiedlich sein. Dies wiederum ermöglicht eine große Freiheit beim Aufbau der Dokumente. Diese Freiheit hat aber auch ihren Preis, da man dadurch auch den Aufwand hat sich selbst um Prüfmechanismen zu kümmern und nicht an die Datenbank selbst abgeben kann.

Bei den dokumentenbasierten Datenbanken gibt es auch keine standardisierte Abfragesprache oder Datenmanipulationssprache die den Anwender unterstützt. Da die Struktur der Datenbank dem Benutzer überlassen wird muss sich auch dieser um eine Art der Datenabfrage kümmern.

In Abbildung 1-9 wird nur eines von vielen möglichen dokumentenorientierten Datenbankmodellen dargestellt.

```
{  
  "Wohnung" : "Penthouse #341",  
  "Zimmeranzahl" : "4",  
  "Größe" : "160",  
  "Mieter" : "Max Mustermann",  
  "Alter" : "35",  
  "Miete" : "3000 EUR"  
}
```

Abbildung 1-9: Dokumentenorientiertes Datenbankmodell

1.3.2.1 Relationales Datenbankmodell

Wie schon erwähnt ist das relationale Datenbankmodell das am meisten verbreitet Modell. Dabei werden die Daten in ein oder mehreren Tabellen gespeichert die wiederum mit verschiedenen Beziehungen miteinander verbunden sind.

Die Tabellen bestehen aus einer vorher definierten Struktur die auch meistens gewisse Einschränkungen für die darin gespeicherten Datensätze vorschreibt. Durch diese Einschränkungen kann gewährleistet werden, dass die vorhanden Daten immer konstant sind und die gewünschten Kriterien erfüllen. Dies kann man auch an der früher präsentierten Tabelle 1-2 Beispiel für Metadaten sehen, dabei wird festgelegt, dass der Vorname bzw. der Nachname nur maximal aus 30 Zeichen bestehen darf.

Eine Tabelle besteht aus Spalten und Zeilen wobei die Zeilen auch als Datensätze bezeichnet werden und die tatsächlichen Daten enthalten. Die Spalten oder auch Attribute werden je nach Verwendung der Daten durch einen Spaltennamen und den passenden Datentyp definiert. Also der Datentyp ist immer auch etwas abhängig von dem verwendeten Hersteller der Datenbank und der zugrunde liegenden Architektur. Man deklariert damit ob es sich in dieser Spalte um Text, Nummer, Datum oder sonstige Daten handelt. Zusätzlich kann man eine Spalte oder mehrere Spalten als Primärschlüssel festgelegt werden um einen Datensatz eindeutig identifizieren zu können, auch wenn Datensätze gegebenenfalls gleich sind.

Die Beziehungen in der die Tabellen zueinander stehen können von verschiedener Art sein und auch sehr komplex aufgebaut sein. Der Zusammenhang zwischen den Beziehungen wird je nach Art der Beziehung mit dem Primärschlüssel oder Fremdschlüssel geschaffen. Man kann die Beziehungen in folgende drei Arten unterteilen:

1. Beziehung 1:1
2. Beziehung 1:n
3. Beziehung n:m

Zur besseren Veranschaulichung der verschiedenen Beziehungen und auch zur Erklärung der Primärschlüssel sowie Fremdschlüssel folgen einige Beispiele.

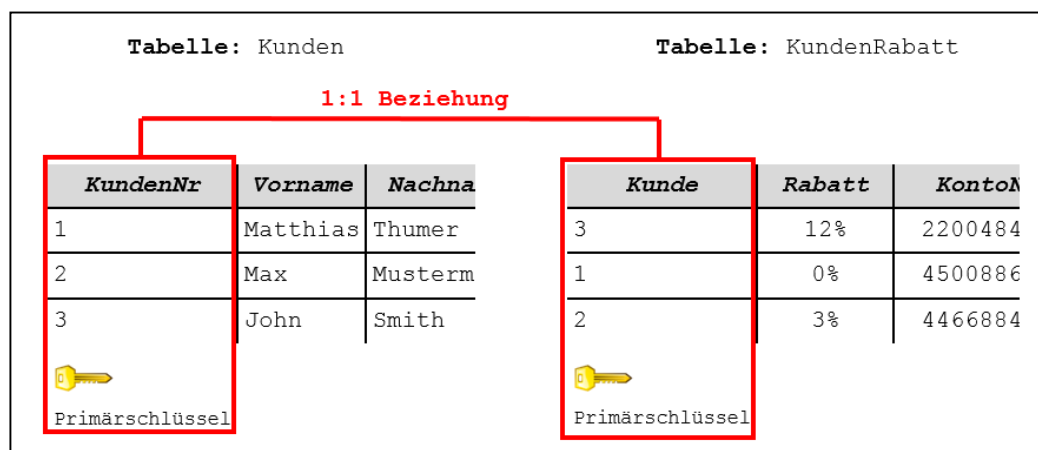


Abbildung 1-10: Beziehungstyp 1:1

Die Beziehung 1:1 ist in der Praxis durchaus in Verwendung, wobei man sagen muss, dass die Information einer solchen Beziehung meistens auch in einer Tabelle zusammengefasst werden können. In der Abbildung 1-10 ist ein solches praxisnahes Beispiel dargestellt, da Aufgrund der verschiedenen Zugriffsrechte die Daten auf zwei Tabellen aufgeteilt wurden.

Grundsätzlich werden die Beziehungen durch den Primärschlüssel, der genau einen individuellen Datensatz in der Tabelle repräsentiert, und dem Fremdschlüssel erstellt. Der Fremdschlüssel dient zur Verknüpfung von den Tabellen wie man in Abbildung 1-11 sieht und ist in diesem Beispiel die Spalte *Kunde* in der Tabelle *Vorgänge*, der Primärschlüssel ist dann die Spalte *KundenNr* der Tabelle *Kunden*.

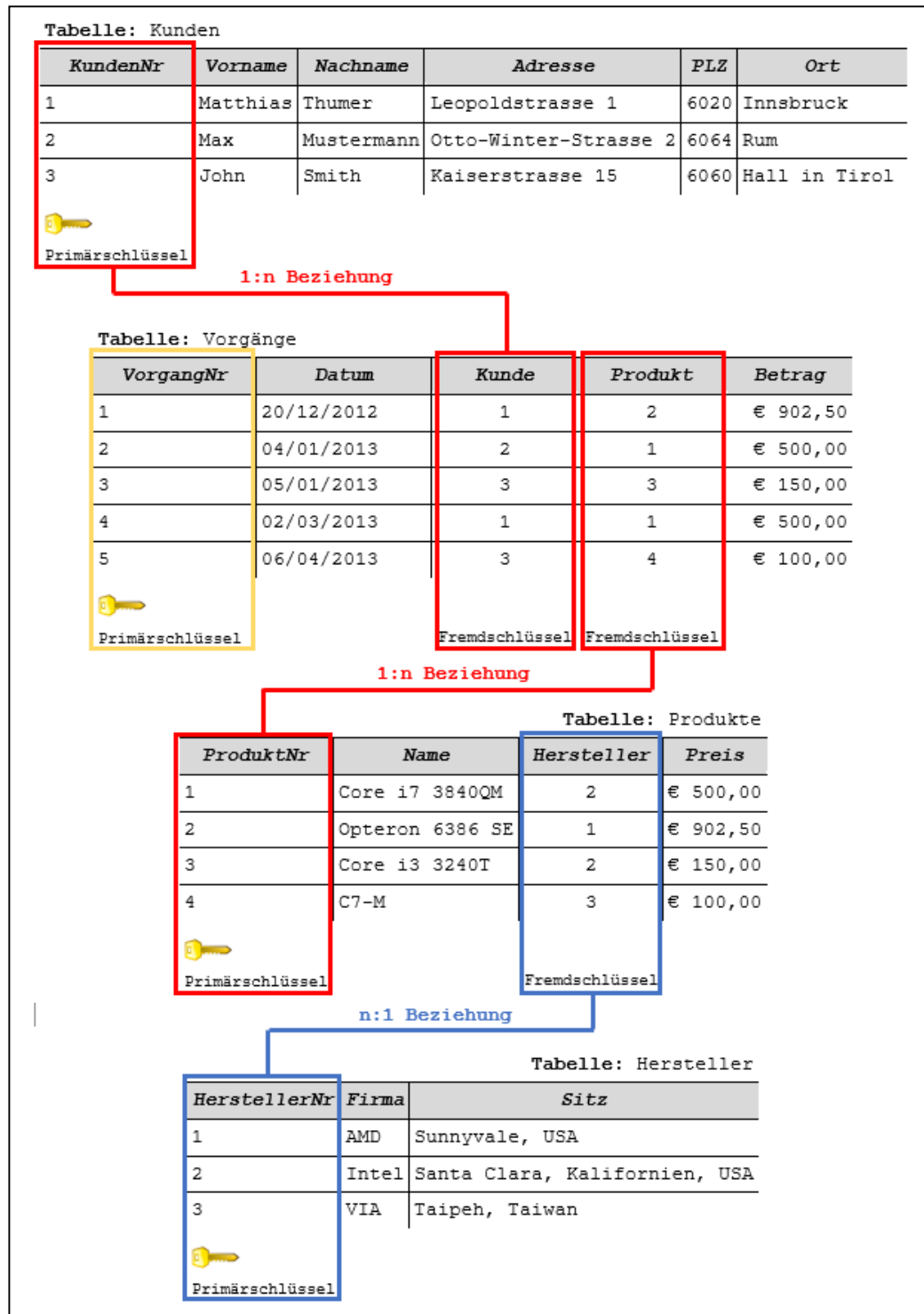


Abbildung 1-11: Beziehungstyp 1:n

In dem dargestellten Beispiel (Abbildung 1-11) kann man eine 1:n Beziehung sehen, damit ist gemeint, dass ein Datensatz in der ersten Tabelle mit einem oder mehreren Datensätzen in der zweiten Tabelle zusammenhängt. Bei genaueren Betrachtung stellt man auch fest, dass es auch eine umgedrehte 1:n Beziehung, also eine n:1 Beziehung, gibt mit dem gleichen Prinzip.

Die 1:n Beziehung oder auch die umgekehrte n:1 Beziehung sind die am häufigsten verwendeten Beziehungen bei relationalen Datenbanken.

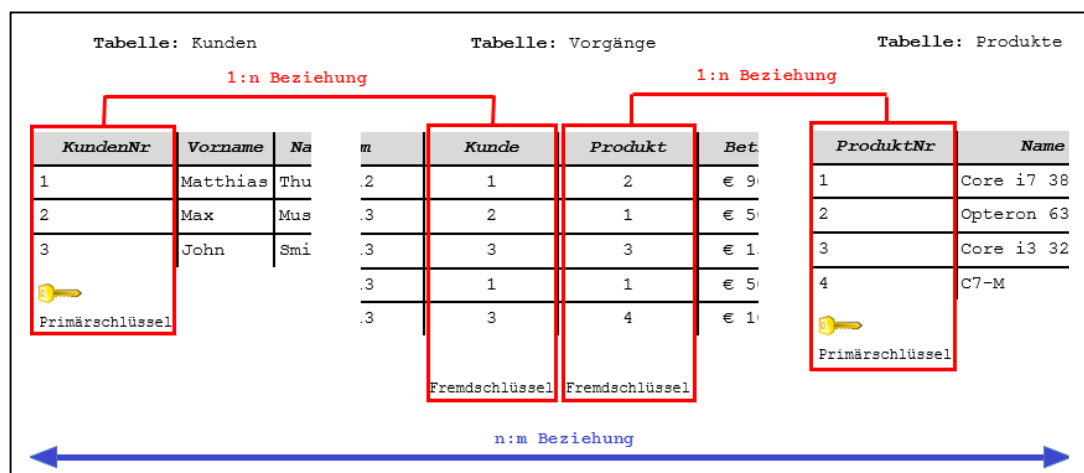


Abbildung 1-12: Beziehungstyp n:m

Die Beziehung n:m in Abbildung 1-12 ist nur möglich mit einer sogenannten Zwischentabelle die in diesem Beispiel die Tabelle *Vorgänge* ist. Also nur durch Verwendung der zwei 1:n Beziehungen kann eine n:m Beziehung erreicht werden.

Vorteil durch Relationen zwischen Tabellen ist auch zugleich ein Nachteil bei komplexeren Beziehungen, da es dabei zu Performance Beeinträchtigungen kommen kann. Dies ist natürlich jeweils abhängig von der verwendeten Hardware bzw. eingesetzten Datenbank. Bei Verwendung von einer MySQL Datenbank in Kombination mit PHP¹¹ (Hypertext Preprocessor) kann es sehr schnell zu Performance Minderungen kommen wie auch in [BoAx2006] beschrieben.

¹¹ PHP ist eine Skriptsprache die hauptsächlich zur Erstellung dynamischer Webseiten oder Webanwendungen verwendet wird

1.3.3 SQL

Bei Datenbanksystemen wird SQL¹² (Structured Query Language) hauptsächlich verwendet für die Daten-Abfrage bzw. Datenmanipulation mit SQL-Befehlen. Zusätzlich kann man mit SQL auch die Datenbankstruktur erstellen, also damit ist gemeint, dass man die Tabellen erstellen kann.

Nach [BaTh2010] lautet die Definition:

„SQL dient in Datenbanksystemen zur Definition von Daten und zur Informationsgewinnung.“

Folglich ist es notwendig für den effektiven Umgang mit SQL-Datenbanken, dass man die SQL-Befehlssyntax beherrscht. Dabei stellt man auch fest, dass die SQL-Ausdrücke sich in zwei Gruppen unterteilen lassen. Es handelt sich einmal um die Befehle zur Datenbearbeitung, die sogenannte DML¹³ (Data Manipulation Language), und um die Kommandos zum Tabellen definieren bzw. ändern, die DDL¹⁴ (Data Definition Language).

Zu der sogenannten DML-Kategorie gehören die SQL-Befehle wie folgt:

1. SELECT
2. INSERT
3. UPDATE
4. DELETE

Der SELECT-Ausdruck dient zur Abfrage von Daten einer oder mehreren Tabelle. Dabei kann ein ganzer Datensatz oder nur ein Feld ausgegeben werden.

Tabelle: Protokoll

<i>Nr</i>	<i>Vorname</i>	<i>Nachname</i>	<i>Zeit</i>	<i>KommenGehen</i>
1	Matthias	Thumer	01/03/2013 12:03:32	K
2	Max	Mustermann	05/03/2013 03:45:01	G
3	John	Smith	12/03/2013 19:33:00	G
4	Matthias	Thumer	20/03/2013 20:54:55	G

Tabelle 1-3: Beispiel Tabelle für SQL-Befehle

¹² SQL dient in Datenbanksystemen zur Definition von Daten und zur Informationsgewinnung

¹³ DML werden die SQL-Befehle zur Datenbearbeitung vereint

¹⁴ DDL sind die SQL-Kommandos für das Datenbankdesign

Mit der Beispiel-Tabelle `Protokoll` werden die SQL-Befehle anhand von sogenannten SQL-Queries also Abfragen oder Befehlen erläutert.

Für den `SELECT`-Befehl könnte eine Beispiel Abfrage wie folgt lauten:

- `SELECT 'Vorname' FROM 'Protokoll' WHERE 'KommenGehen' = 'G'`

Mit dieser Abfrage würden alle Vornamen in der Tabelle `Protokoll` bis auf den Vornamen `Matthias` zurückgegeben.

Das Kommando `INSERT` würde man verwenden wenn man einen neuen Datensatz der Tabelle hinzufügen will und könnte wie folgt sein:

- `INSERT INTO 'Protokoll' ('Nr', 'Vorname', 'Nachname', 'Zeit', 'KommenGehen') VALUES (5, 'John', 'Smith', '2013-03-21 01:23:12', 'K')`

Wenn man aber nicht einen neuen Eintrag machen will sondern nur einen bestehenden Datensatz ändern will verwendet man den SQL-Befehl `UPDATE`. Mit folgendem Befehl würde man die Zeit vom Datensatz `Max Mustermann` ändern auf `30/03/2013 00:00:00`:

- `UPDATE 'Protokoll' SET 'Zeit' = '2013-03-30 00:00:00' WHERE 'Nr' = 2`

Der SQL-Befehl `UPDATE` kann wie im Beispiel für einen Datensatz gelten oder man kann auch mehrere Datensätze gleichzeitig abändern indem man das Kriterium für die betroffenen Datensätze erweitert.

Mit `DELETE` kann man einen oder mehrere Datensätze einer Tabelle löschen, wie man im folgenden SQL-Befehl sehen kann:

- `DELETE FROM 'Protokoll' WHERE 'KommenGehen' = 'G'`

Mit diesem Befehl werden insgesamt drei Datensätze in der Tabelle `Protokoll` gelöscht, alle welche den Wert `G` in der Spalte `KommenGehen` besitzen.

In die DDL-Kategorie der SQL-Kommandos würden folgende Befehle gehören:

1. CREATE
2. ALTER
3. DROP

Wenn wir mit der Beispiel-Tabelle `Protokoll` fortfahren und mit dem SQL-Kommando `CREATE` diese Tabelle erstellen würden, würde der Befehl wie folgt aussehen:

- ```
CREATE TABLE IF NOT EXISTS `Protokoll` (
 `Nr` INT(11) NOT NULL,
 `Vorname` VARCHAR(40) NOT NULL,
 `Nachname` VARCHAR(40) NOT NULL,
 `Zeit` DATETIME NOT NULL,
 `KommenGehen` VARCHAR(1) NOT NULL,
 PRIMARY KEY (`Nr`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

Mit dem SQL-Befehl `CREATE` kann man also eine ganze Tabelle erstellen sowie die Richtlinien für die einzelnen Spalten festlegen. Damit ist gemeint, dass der Wert in der Spalte `Vorname` nur aus maximal 40 Zeichen bestehen darf und alle Spalten auch nicht leer bleiben dürfen, also jede Spalte muss einen Wert haben.

Zum Anpassen einer Spalte in einer Tabelle verwendet man `ALTER` und ein Beispiel dazu wäre wie folgt:

- ```
ALTER TABLE `Protokoll` CHANGE `Nachname` `Nachname` VARCHAR(60) NOT NULL
```

Der obere Befehl verändert die Richtlinie der maximal erlaubten Zeichen für die Spalte `Nachname` von 40 auf 60 Zeichen.

Zuletzt kommt noch ein Beispiel um eine ganze Tabelle zu löschen und dies ist möglich mit dem Befehl `DROP` und im folgenden Beispiel dargestellt:

- ```
DROP TABLE `Protokoll`
```

Wichtig ist noch, dass es noch mehrere SQL-Ausdrücke gibt wobei die Erwähnten die wichtigsten sind. Man kann auch verschiedenen SQL-Ausdrücke verschachteln und mit Vergleichsoperatoren (`<`, `>`, `<=`, `>=`, `<>`, `LIKE`, `EXISTS`, `IN`) und Logischen Operatoren (`NOT`, `AND`, `OR`) ausstatten.

Alle Beispiel-Befehle basieren auf der Annahme, dass es sich dabei um eine MySQL Datenbank handelt. Da die SQL-Befehle leicht unterschiedlich sein können bei verschiedenen Datenbankherstellern wobei der grundlegende Aufbau des Befehls durchaus gleich bleibt. Änderungen sind vor allem bei Spalten die mit einem Datum- oder einen Zeitwert versehen sind, da man je nach Hersteller ein anderes Format verwenden muss.

Es sollte auch erwähnt werden, dass es auch Hersteller abhängige Erweiterungen für SQL gibt, wie zum Beispiel PL/SQL<sup>15</sup> (Procedural Language / SQL) für Oracle Datenbanken und T/SQL<sup>16</sup> (Transact SQL) für Microsoft SQL Server Datenbanken. PL/SQL ist dabei zu sehen wie eine eigene Programmiersprache für Datenbanken mit der Programme beziehungsweise Funktionen in SQL-Abfragen oder SQL-Befehle integriert werden können wie auch in [FePri2002] beschrieben. Natürlich bietet auch die T/SQL Erweiterung diese Funktionalität.

---

<sup>15</sup> PL/SQL ist eine proprietäre Programmiersprache der Firma Oracle und verbindet die Abfragesprache SQL mit einer prozeduralen Programmiersprache, bei der die Syntax sehr stark an die Programmiersprache Ada angelehnt ist.

<sup>16</sup> T/SQL ist eine proprietäre Programmiersprache und Erweiterung des SQL-Standards durch die Firma Microsoft.

## 2 Einblick und Aufgabenstellung

In diesem Kapitel erfolgt ein kurzer Einblick in die bestehende Software. Dazu wird für das Aufgabenverständnis der Verwaltungssoftware und die Offline-Module kurz erläutert sowie auch ihre Funktionen. Zum Abschluss erfolgt die Zusammenstellung der Aufgabenstellung.

### 2.1 Die Verwaltungssoftware

Die ansteigende Globalisierung der Firmen führt zu neuen Absatzmärkten, Beschäftigten und Dienstleistern. Zur gleichen Zeit wird der Konkurrenzdruck auf die Unternehmen vergrößert. Damit ein Betrieb wettbewerbsfähig bleiben kann, müssen bestehende Ressourcen effizienter genutzt und optimiert werden. An dieser Stelle setzt die Verwaltungssoftware an.

Die Verwaltungssoftware ist eine firmeninterne Entwicklung zur Unterstützung der Mitarbeiter. Mit ihr werden Angebote für Kunden angefertigt, Rechnungen erstellt, wirtschaftliche Berichte automatisiert, Bankkonten dargestellt, Zahlungseingänge erfasst und Offene-Posten angezeigt.

Die Entwicklung der Verwaltungssoftware wurde im Laufe der Zeit immer weitergeführt und ist auch dementsprechend umfangreich geworden. Man hat die Software komplett online entwickelt als eine sogenannte Web-App. Die verwendete Programmiersprache ist PHP, wobei mit der Version 4.3.0 gestartet wurde und mittlerweile man bei Version 5.2.17 angelangt ist. Durch die Größe der Software ist es auch immer schwieriger geworden auf eine neuere PHP Versionen umzusteigen, da man immer wieder Code Segmente an die neuen Versionen anpassen musste. Als Datenbank wird die sehr verbreitete relationale Datenbank MySQL verwendet mit der Version 5.1.48.

Grundsätzlich ist die Verwaltungssoftware in fünf Bereiche unterteilt, wie folgt:

1. Allgemeine Übersichten
2. Stammdaten wie Kunden, Lieferanten und Produkte
3. Dokumente
4. Buchhaltung
5. Einstellungen

Diese Bereiche hängen durch unterschiedlichen Bedingungen zusammen oder voneinander ab. So wie zum Beispiel: Erst wenn ein Auftrag angelegt ist und ein Lieferschein gedruckt wurde, kann die Rechnung gebucht werden. Dann scheint diese Rechnung unter den Offenen-Posten auf bis die Zahlungseingänge vermerkt bzw.

hinterlegt werden. Solche Zusammenhänge gibt es noch zahlreiche in der Software, die es dem Anwender sehr einfach machen die gewünschten Aufgaben in der richtigen Reihenfolge zu erledigen ohne wichtige Schritte auszulassen. Damit werden die Abläufe in der Firma standardisiert.

Im Rechnungsbereich kann der Anwender, der die dazu nötigen Rechte hat, letzte Änderungen am Auftrag durchführen, sowie die Rechnung erstellen. Dabei werden das Rechnungsdatum und die Rechnungsnummer vergeben, die nachträglich nicht mehr änderbar sind. Zusätzlich können Zahlungsbedingungen auf der Rechnung angezeigt werden wie zum Beispiel: Zahlung sofort nach Rechnungserhalt ohne Abzug. Anzumerken ist auch, dass die Verwaltungssoftware automatisch die Rechnung, aber auch die anderen Dokumente, immer in dreifacher Ausführung erstellt. Also in allen benötigten Sprachen: Deutsch, Spanisch und Englisch.

The screenshot displays the 'COMTRAQ Administration' web interface. On the left is a navigation menu with categories like 'General', 'Kunden', 'Lieferanten', 'Produkte', 'Bestellungen', 'Rechnungen', 'Lieferscheine', 'Auftragsbestätigungen', 'Buchhaltung', and 'Einstellungen'. The main area is titled 'Druck' and 'Comtraq GmbH' with a user role of 'Administrator Administrator'. It features tabs for 'Rechnung', 'Lieferschein', and 'Auftragsbestätigung'. The 'Rechnung' tab is active, showing a form with sections: 'Rechnung Anschrift' (Firma: Matthias Thumer, Comtraq ATU 62; Datum: 15.07.2011; Vorname, z.Hd., Nachname, Email: matthias@thumer.com), 'Rechnung Adresse' (Strasse: An-der-Lanstrasse 28b; Land: Austria; PLZ: A 6020; Ort: Innsbruck), 'Rechnung Daten' (Rechnung Nr.: 41100010; Rechnung Datum: 15.07.2011; Lieferung vom: N/A; Bestellnummer; Auftragsnummer: 71100010), and 'Produkte und Leistungen'.

Abbildung 2-1: Ausschnitt vom Rechnungsbereich

Sobald die Rechnung erstellt ist, scheint diese auch in dem Bereich offene Posten auf. In diesem Bereich bekommt man eine Übersicht aller offenen ausgehenden Rechnungen. Dabei werden die wichtigsten Eckdaten des Auftrags / der Rechnung angezeigt, wie das Rechnungsdatum, der Betrag der Rechnung inklusive der Steuern, der offene Betrag und die Anzahl der Tage seit die Rechnung erstellt wurde.

Zusätzlich wird auch der letzte Zahlungseingang mit dem Datum und dem Betrag des Zahlungseinganges angezeigt.

Auch vorgefertigte Zahlungserinnerungen sowie Mahnungsschreiben können in diesem Punkt angefertigt werden und an den Empfänger per Email oder per Post, indem man das PDF Dokument druckt, versendet werden. Diese Schreiben werden dann mit den Daten des Auftrags sowie der Rechnung und Zahlungseingänge automatisch gefüllt, sodass die Schreiben eine persönliche Note bekommen. Dies ist auch erforderlich um die Schreiben, falls notwendig vor Gericht zu verwenden.

| Datum                                                                                                                        | RE Nr.   | Betrag       | offener Betrag      | offen seit |
|------------------------------------------------------------------------------------------------------------------------------|----------|--------------|---------------------|------------|
| 2011-07-15                                                                                                                   | 41100010 | EUR 1.443,60 | EUR 1.137,60        | 23 Tage    |
| <b>Details:</b> Es sind bereits EUR 306,00 eingegangen. Der letzte Zahlungseingang war am 27.07.2011 in Höhe von EUR 100,00. |          |              |                     |            |
| 2011-07-28                                                                                                                   | 41100011 | EUR 120,00   | EUR 95,00           | 10 Tage    |
| <b>Details:</b> Es sind bereits EUR 25,00 eingegangen. Der letzte Zahlungseingang war am 31.07.2011 in Höhe von EUR 5,00.    |          |              |                     |            |
| <b>Gesamtbetrag offen</b>                                                                                                    |          |              | <b>EUR 1.232,60</b> |            |

Abbildung 2-2: Ausschnitt von den Offenen-Posten

Der Bankkontenbereich ermöglicht dem Anwender eine genaue Übersicht über den Zustand des Firmenbankkontos. Dabei werden auch alle einzelnen Buchungen die am Konto durchgeführt werden angezeigt. Auch der aktuelle Saldo wird angeführt, sowie die Summe der Gutschriften und der Lastschriften jedes Monats. Dadurch ist es möglich auch den Saldo von früheren Monaten einzusehen um mögliche Tendenzen der Kontobewegungen zu erkennen.

Die Bankdaten werden in die Verwaltungssoftware importiert indem man in der online Banking Software der Bank alle Datensätze in eine CSV<sup>17</sup> (Comma-separated values)-Datei exportiert. Das ist sehr wichtig, da die Software der Bank diese Daten elektronisch nur für ein Jahr zur Verfügung stellt. Deswegen wurde auch dieser Bereich entwickelt um die Möglichkeit zu haben bis auf den ersten Datensatz zurück zu gehen.

<sup>17</sup> CSV ist ein Dateiformat mit dem Tabellen oder eine Liste unterschiedlich langer Listen abgebildet werden (siehe <http://tools.ietf.org/html/rfc4180>)

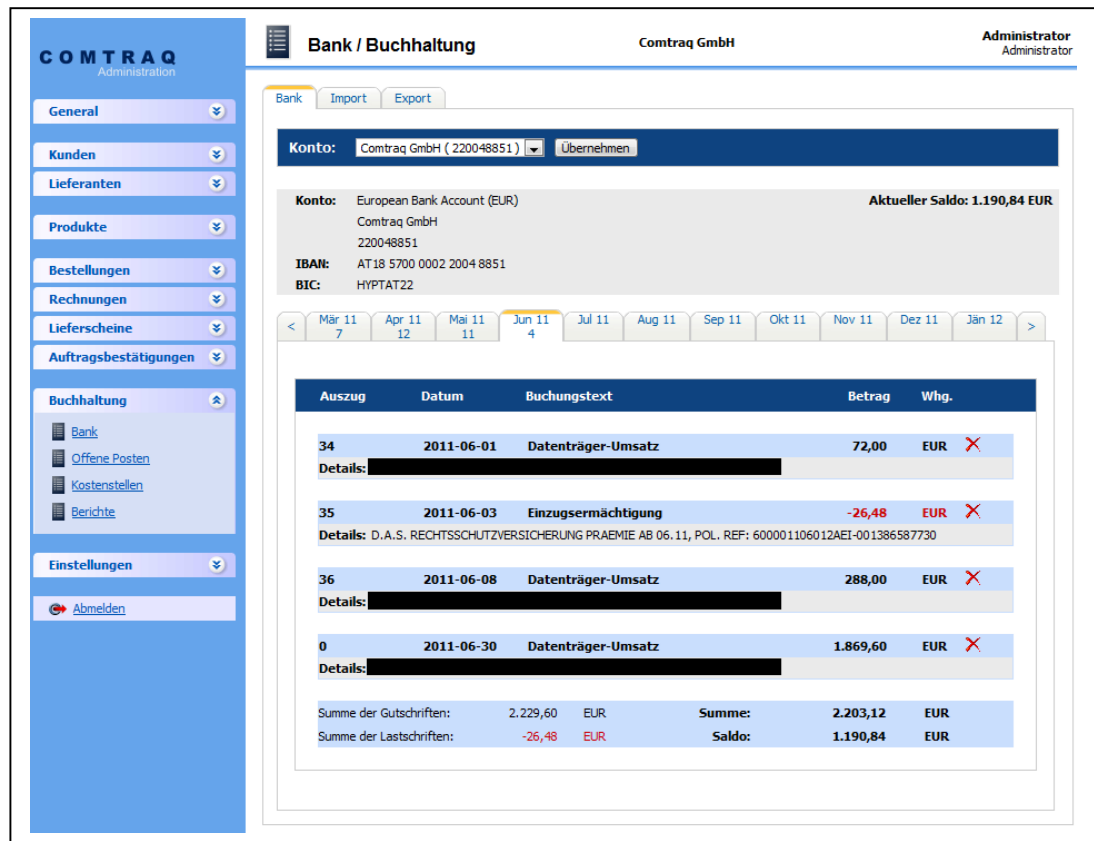


Abbildung 2-3: Ausschnitt vom Bankkontenbereich

## 2.2 Die Offline-Module

Die Offline-Module sind prinzipiell von der Funktionsfähigkeit gleich wie die einzelnen Bereiche der Verwaltungssoftware. Als Grundlage für die GUI<sup>18</sup> (Graphical User Interface) der Module dienen auch die Oberflächen der bereits bestehenden Verwaltungssoftware.

Jedes Modul ist komplett unabhängig von der Verwaltungssoftware und von den anderen Modulen bedienbar. Alle Datensätze bzw. Änderungen werden in einer lokalen Microsoft Access Datenbank gespeichert. Entwickelt wurden die Offline-Module in C#<sup>19</sup> (C Sharp).

### 2.2.1 Bank Account Manager

Die Oberfläche dieses Moduls wurde ähnlich aufgebaut wie in der Verwaltungssoftware (Abbildung 2-3). Durch diese Vorgehensweise hat der jeweilige Anwender der

<sup>18</sup> GUI ist eine Software-Komponente, die dem Anwender eines Computers die Interaktion mit der Maschine über graphische Symbole erlaubt.

<sup>19</sup> C# ist eine vom Softwarehersteller Microsoft entwickelte Programmiersprache

Software es erheblich leichter sich umzustellen, da alles schon vertraut ist, und braucht dadurch eine weitaus kürzere Zeit zur Einarbeitung bzw. wenn er wechselt zwischen online und offline Variante.

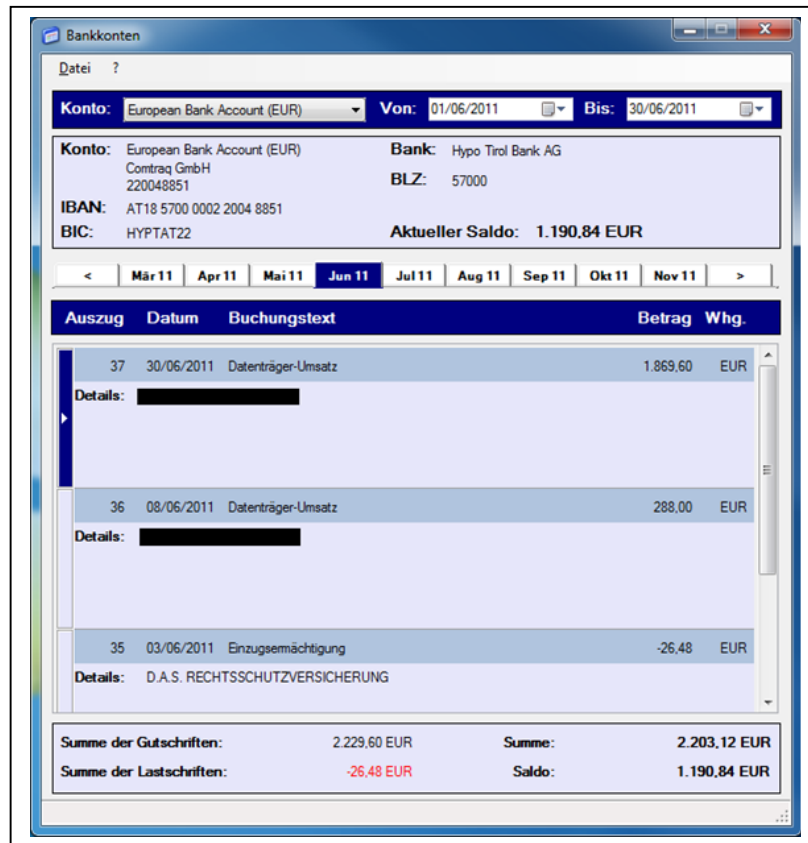


Abbildung 2-4: Oberfläche des Moduls Bank Account Manager

Die Daten wie der aktuelle Saldo, die Summe der Gutschriften und der Lastschriften sowie die Monatssumme und der Saldo bis zum ausgewählten Zeitpunkt werden immer sofort aus den Datensätzen in der Moduldatenbank generiert.

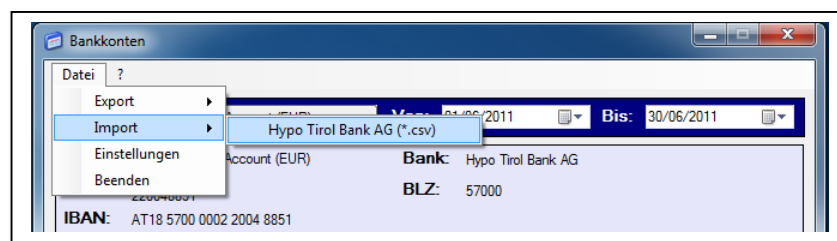
Für die Anzeige der ganzen Buchungen gibt es zwei Möglichkeiten für den Anwender. Man kann mit dem Tab-Menü, auf dem die Monate stehen, jeweils durch die verschiedenen Monate navigieren oder man verwendet ganz oben neben dem Kontonamen die Felder **Von** und **Bis**. Dabei wählt man genau den gewünschten Bereich aus und sobald man im Feld **Bis** das Datum festgelegt hat werden die neuen Datensätze mit der entsprechenden Eingrenzung angezeigt.



**Abbildung 2-5: Anzeigemöglichkeiten**

Die Datensätze die im Bankkontomodul angezeigt werden, importiert man indem man die CSV-Datei unter dem Menü Datei und dann auf Import -> Hypo Tirol Bank AG (\*.csv) drückt und auswählt.

Man bekommt die CSV-Datei von der Online Banking Software der Bank. Das einzige Problem oder die einzige Unannehmlichkeit dabei ist, dass die Online Banking Software diese Funktion immer wieder ändert. Damit ist gemeint, dass das Format bis jetzt schon fünfmal geändert wurde. Die Änderungen des Formats beziehen sich hauptsächlich auf die Felder Datum und Uhrzeit, aber manchmal auch das ganze Format. Also bei dem Datumfeld wird das Datum dann zum Beispiel: einmal 2011-12-30, 30/12/2011 oder 2011-30-12 verwendet. Bei den restlichen Textfeldern wird auf einmal am Beginn das Zeichen „,“ gesetzt und auch am Ende. Dadurch muss man immer wieder die Importfunktion im Modul anpassen, deswegen wurde diese Funktion im Programm auch mit einer eigenen C#-Klasse umgesetzt. Folglich ist es auch wesentlich einfacher den Import Prozess anzupassen.



**Abbildung 2-6: Import neuer Datensätze**

Wenn man beim Import doppelte Datensätze hat weil man sie ausversehen zweimal importiert hat kann man sie ganz einfach wieder löschen. Dabei muss man nur auf den gewünschten Datensatz mit der rechten Maustaste drücken und dann löschen betätigen.

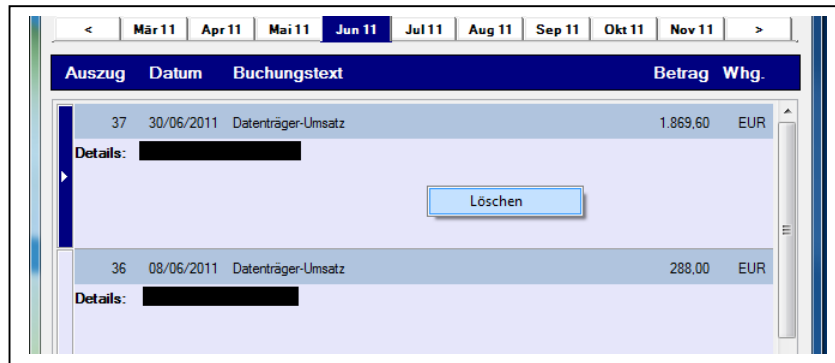


Abbildung 2-7: Datensätze löschen

Zusätzlich muss man einmal das Bankkonto im Programm einrichten. Das ist möglich unter dem Dateimenü ganz oben, dabei öffnet sich ein neues Fenster indem man auch den Datenbankpfad festlegen kann. Wenn man auf die drei Punkte rechts neben dem Bankkonto drückt, dann öffnet sich wieder ein Fenster indem man alle Bankdaten einpflegen kann. Darüber hinaus kann man auch unter Einstellungen das Standardkonto festlegen, also wenn man mehr als ein Konto hat welches beim Start gleich ausgewählt ist.

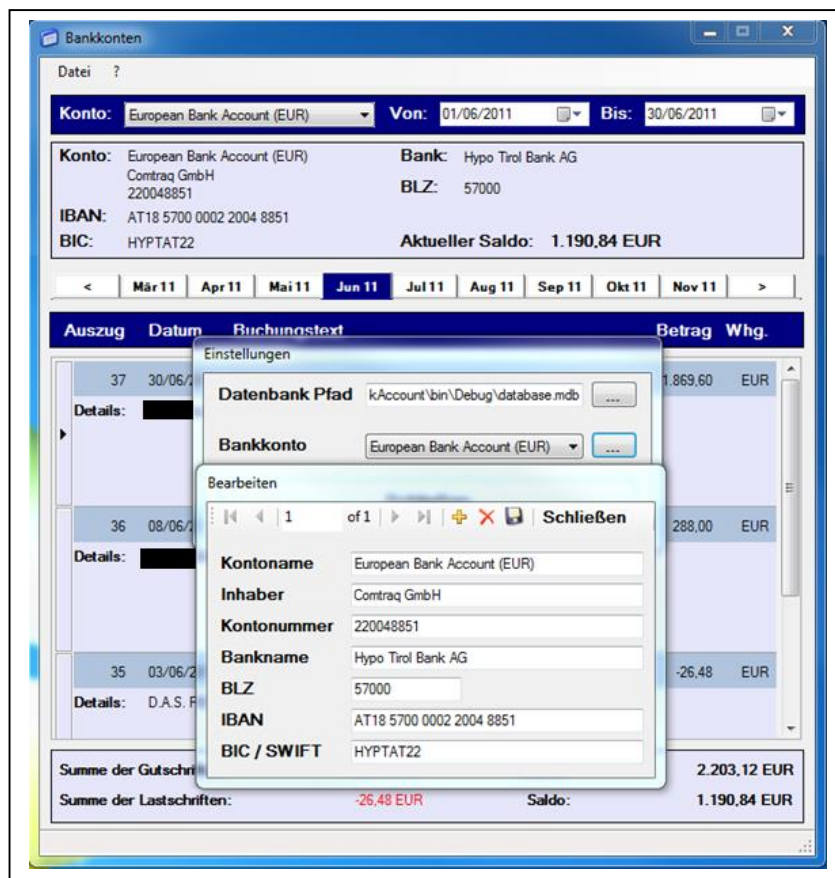


Abbildung 2-8: Bankkonto Einstellungen

## 2.3 Aufgabenstellung

Das Hauptaugenmerk ist die Implementierung der XML-Synchronisierungsschnittstelle mit den vorhandenen Datenbanken. Damit ist gemeint, dass die MySQL Datenbank mit der Microsoft Access Datenbank, der Offline-Module, Daten austauscht mit Verwendung der XML-Schnittstelle.

Dabei soll nur minimal auf die zugrundeliegende Architektur der online Verwaltungssoftware eingegriffen werden um mögliche Fehlfunktionen der bestehen Software zu vermeiden.

Also in die MySQL Datenbank sowie die bestehenden Datensätze sollen so wenig wie möglich abgeändert werden um die Schnittstelle zu integrieren. Somit wird es sogar erwünscht weitere Tabellen hinzuzufügen um eventuelle Änderungen an bestehenden Tabellen zu vermeiden.

Wenn man aber dennoch bestehende Datensätze erweitern muss sollte diese Erweiterung automatisch durch die Synchronisierungsschnittstelle geschehen und nicht manuell. Damit meint man, dass die Datensätze erst bearbeitet werden bzw. erweitert werden wenn diese von einem Offline-Module angefordert werden oder aktualisiert werden. Und nicht einfach alle bereits gespeicherten Datensätze in der MySQL Datenbank bearbeitet werden müssen um die Funktionalität der neuen Module zu integrieren.

Der Datenzugriff auf die MySQL Datenbank von den Offline-Modulen soll ausschließlich über die XML-Synchronisierungsschnittstelle stattfinden. Auf keinem Fall darf eine direkte Verbindung von den Offline-Modulen mithilfe von .NET Bibliotheken aufgebaut werden, wie zum Beispiel ADO.NET<sup>20</sup> (ActiveX Data Objects), OLE DB<sup>21</sup> (Object Linking and Embedding Database) oder auch ODBC<sup>22</sup> (Open Database Connectivity).

Um mögliche Fehlfunktionen und eventuelle Probleme in der Synchronisierung zu verstehen bzw. auch nachvollziehen zu können sollen alle Vorgänge die mittels der Schnittstelle durchgeführt werden auch protokolliert werden. Dieser Ansatz steuert nicht nur der Problemlösung bei sondern auch der Sicherheit. Dadurch will man sicherstellen, dass man absichtliche Datenmanipulationen einerseits aufdecken kann aber auch dann wieder die ursprünglichen original Daten wieder herstellen kann wenn nötig.

---

<sup>20</sup> ADO.NET ist eine Sammlung von Klassen, die den Zugriff auf relationale Datenbanken gewährleisten und Teil der von Microsoft entwickelten .NET-Plattform

<sup>21</sup> OLE DB ist eine von Microsoft entwickelte Programmierschnittstelle für einen standardisierten Zugriff auf unterschiedlichste Datenquellen und wurde als Nachfolger von ODBC entwickelt

<sup>22</sup> ODBC ist eine standardisierte Datenbankschnittstelle, die SQL als Datenbanksprache verwendet

Klarerweise muss jeglicher Datenaustausch verschlüsselt sein und man soll auch die Option haben einzelne Anwender zu sperren bzw. ganze Module zu blockieren. Dies setzt natürlich voraus, dass man den Datenaustausch einzelner Anwender genau zuweisen können muss und auch dass der Datenaustausch der spezifischen Offline-Module segmentiert sein muss.

Aufgrund der laufenden Anpassung und auch Funktionserweiterung der online Verwaltungssoftware ist es notwendig, dass die Struktur der Datenbank aber auch die Funktionalität sowie Logik der XML-Schnittstelle relative einfach geändert werden kann wenn nötig.

Daraus ergibt sich auch dass die Offline-Module eine recht einfach änderbare Datenbank brauchen. Zusätzlich muss bei den Offline-Modulen auch die Datenbankstruktur durch den Updatevorgang anpassbar sein um die eventuellen neuen Funktionen zu unterstützen.

Auf die beim Anwender der Offline-Module gespeicherten Daten soll nur mit der Anwendung zugegriffen werden können. Damit ist gemeint, dass man nicht direkt auf die Rohdaten, die durch die Synchronisierung in die lokale Datenbank beim Benutzer gespeichert werden zu greifen können soll. Auf diese Weise möchte man erreichen, dass allfällige Daten beim Anwender nutzlos gemacht werden können im Falle dass man diesen sperren will.

Letztendlich sollte die Bedienung um die Daten zu synchronisieren ganz einfach und intuitiv sein. Also man sollte ersten einmal auch einen Überblick haben wie weit die Synchronisierung ist und auch die Durchführung selbst zu starten sollte kinderleicht sein. Natürlich sollte auch der ganze Synchronisierungsprozess zügig abgeschlossen werden.

Wenn möglich sollten die Funktionen bei den Offline-Modulen soweit wie möglich gleich sein um die Bedienung für die Anwender zu erleichtern. Damit können die Benutzer auch mit den anderen Modulen schneller zurecht, da grundlegende Funktionen bzw. Vorgehensweisen immer unverändert bleiben.

Darüber hinaus sollte der Prozess eine gewisse Intelligenz besitzen, infolgedessen sollten nur Daten ausgetauscht werden die auch wirklich nötig sind auch wenn der Anwender zum Beispiel mehrmals die Synchronisierung startet obwohl keine neuen Änderungen gemacht wurden, dann sollte auch kein unnötiger Datenaustausch durchgeführt werden.

Zusammenfassend sind folgende Kriterien zu beachten:

1. Es wird erwünscht neue Datenbanktabellen hinzuzufügen statt bestehende umfangreich zu ändern in der MySQL Datenbank
2. Änderungen an den Datensätzen erst bei Anforderung bzw. Aktualisierung der Offline-Module

3. Kein direkter Datenzugriff auf die MySQL Datenbank von den Offline-Modulen
4. Möglichkeit der Nachvollziehbarkeit der Änderungen durch die Synchronisierung
5. Verschlüsselter Datenaustausch und die Möglichkeit der Sperrung einzelner Offline-Module und Anwender
6. Einfach adaptierbare Datenbankstruktur und gänzlich updatebare Offline-Module
7. Kein Zugriff auf lokale Rohdaten beim Anwender
8. Einheitliche bzw. übersichtliche Bedienung und Statusanzeige des Fortschrittes
9. Synchronisierung sollte rasch abgeschlossen werden und unnötiger Datenaustausch sollte vermeiden.

## 3 Systemkonzept

In diesem Kapitel wird das Systemkonzept für die XML-Schnittstelle, die Architektur der Offline-Module, Installation bzw. Updatevorgang der Offline-Module und das Datenbankkonzept vorgestellt.

### 3.1 Konzept der Offline-Module

Aufgrund der Gegebenheiten in der Firma, die die Offline-Module schlussendlich verwendet, und der verfügbaren Werkzeuge war es von Beginn des Projektes klar, dass man die Anwendung auf .NET<sup>23</sup> (dot NET Framework) aufbaut. Durch die Verwendung der .NET Plattform bekommt man als Entwickler eine große Auswahl an Klassen und Bibliotheken die die Programmierung wesentlich erleichtern.

.NET stellt weiteres auch noch eine Reihe von Werkzeugen zur Verfügung, allen voran Visual Studio, eine mehrsprachenfähige Softwareentwicklungsumgebung mit Debugger und GUI-Designer, die optimal auf Aufgaben wie das Erstellen von Webservices oder Windows Form Applikationen ausgerichtet ist.

Die Verwendung der Entwicklungsumgebung Visual Studio, die normalerweise mit erheblichen Kosten verbunden ist, hat sich ergeben, da die Firma bereits Microsoft Partner ist und dadurch schon einen jährlichen Beitrag an Lizenzgebühren zahlt in denen auch die Kosten der Entwicklungsumgebung schon inkludiert sind. Mit der Verwendung von Visual Studio ist es möglich den Entwicklern eine gemeinsame Umgebung bereitzustellen, auch wenn die Software bzw. die Offline-Module in Zukunft in einer anderen .NET-Sprache weiter entwickelt werden.

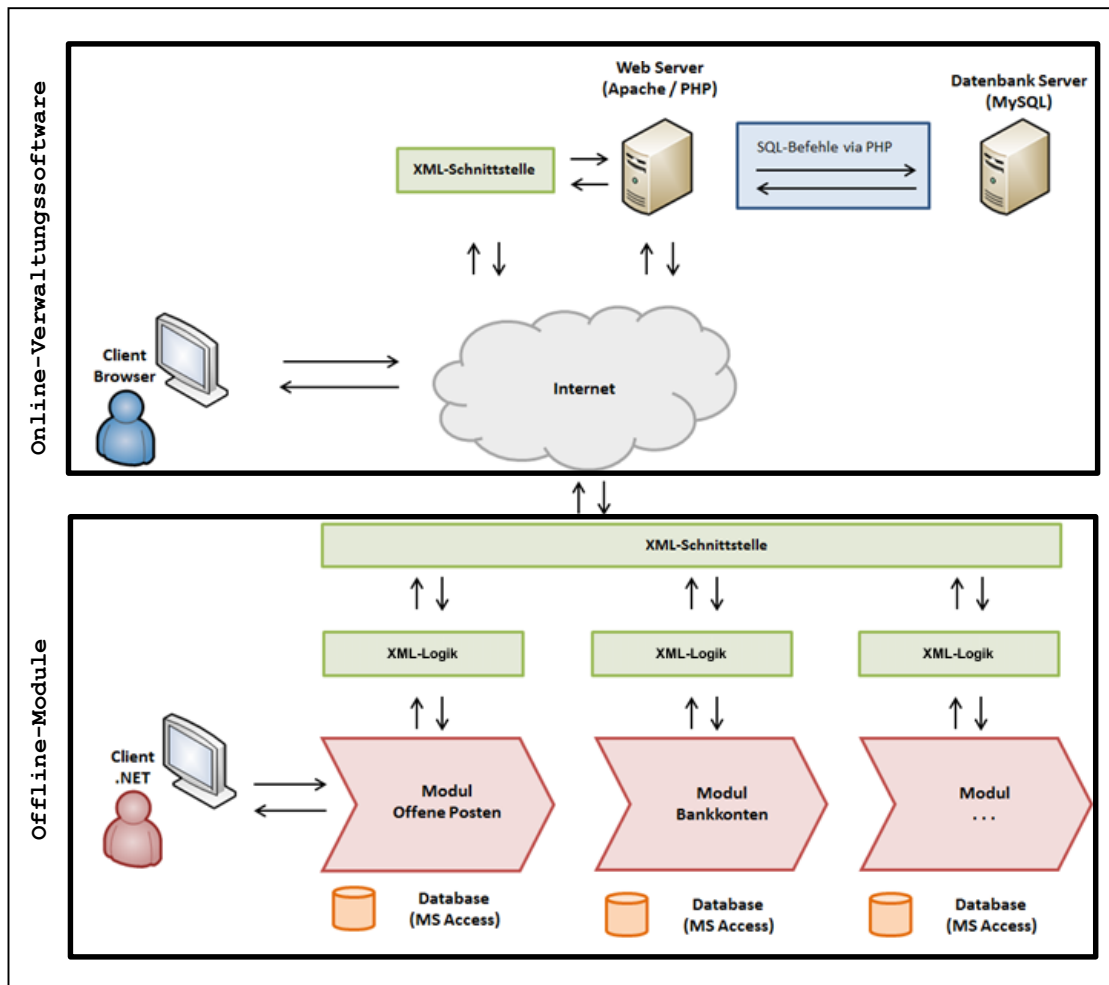
Ein weiterer Punkt für die Verwendung von .NET und der Entwicklung der Offline-Module in C# war, dass die Anwender nur Windows Computer einsetzen. Dadurch war die Kompatibilität zu den bereits vorhandenen Systemen gewährleistet.

Wichtig ist auch, dass jedes Offline-Module weitgehen unabhängig von den anderen Modulen bleibt, um bei dem Anwender nur die wirklich notwendigen Module zu installieren zu können.

In Abbildung 3-1 sieht man einen Überblick über die Verwaltungssoftware die sich in die Online-Verwaltungssoftware und die jeweiligen Offline-Module unterteilen lässt, sowie über den Aufbau der Module.

---

<sup>23</sup> .NET bezeichnet eine von Microsoft entwickelte Software-Plattform zur Entwicklung und Ausführung von Anwendungsprogrammen



**Abbildung 3-1: Architektur der Verwaltungssoftware mit Offline-Module**

Als Datenbank für die einzelnen Module wird eine Microsoft Access Datenbank verwendet, da man dazu nur eine Datei braucht und auch die Handhabung bei Problemen sehr einfach ist. Damit ist gemeint, dass wenn ein Anwender ein Problem hätte mit einem Offline-Module, könnte man sich einfach die Datenbank schicken lassen und dann mit dem lokalen Computer auf Ursachensuche gehen.

Um die Module auch klein zu halten werden auch nicht einfach alle Daten der Online-Verwaltungssoftware in das jeweilige Module geladen, da es sich ja dabei um Daten von mehreren Jahren handelt, die nicht notwendig sind in der Offline-Variante. Deswegen werden nur Datensätze bis zu einem Alter von zwei Monaten gespeichert. Durch diese Maßnahme wird die Datenmenge enorm reduziert bei den Anwendern der Offline-Module und die Anwendung kann auch viel schneller agieren.

### 3.2 Installation bzw. Updatevorgang der Offline-Module

Grundsätzlich muss sichergestellt sein, dass alle Offline-Module aller Anwender immer am gleichen und letzten Stand sind. Dazu wird bei jedem Start der Anwendung

geprüft ob es eine neue Version gibt oder nicht. Im Fall, dass es eine neuere Version gibt wird diese dann auch gleich eingespielt.

Seit Visual Studio 2005 bietet Microsoft auch die Verteilung mit der ClickOnce-Technologie an wie auch in [KüAn2011] beschrieben. Während eine herkömmliche Windows-Anwendung auf dem lokalen Rechner installiert wird, kann man eine ClickOnce veröffentlichte Anwendung auf einem Server zur Verfügung stellen. Dabei kann es sich um einen Webserver oder eine Netzlaufwerkfreigabe handeln, sowie einen Datenträger.

Die Installation bzw. das Update der Offline-Module muss möglich sein ohne Administrator-Rechte. Dies ist mit der ClickOnce-Technologie gewährleistet wobei man bei der Entwicklung und bei dem Aufbau der Anwendung auf gewisse Einschränkungen Rücksicht nehmen muss. Es dürfen keine Einträge oder Änderungen an der Windows-Registrierungsdatenbank und am Desktop des Anwenders gemacht werden.

ClickOnce bietet auch die Möglichkeit, Anwendungen so einzurichten, dass der Benutzer sowohl online als auch offline mit der Anwendung arbeiten kann. Im Offline-Modus wird ein Programmpunkt im Startmenü angelegt, der die Anwendung startet.

Für die Offline-Module wird ClickOnce auf einem Webserver im LAN<sup>24</sup> (Local Area Network) installiert und wird eingerichtet, dass die Anwendung am Benutzer-Computer auch offline funktioniert. Zusätzlich wird ein VPN<sup>25</sup>-Zugang (Virtual Private Network) eingerichtet für jeden Anwender der Offline-Module um auch ein Update zu machen wenn der Anwender länger unterwegs ist bzw. schon seit längerem nicht mehr im Büro war. Dadurch wird auch der interne Webserver, auf dem die ClickOnce-Anwendung installiert ist, auch von außerhalb erreichbar insofern der Anwender Zugang zu einem Internet-Anschluss hat.

Zusätzlich wird durch die Verwendung von ClickOnce auch sichergestellt, dass am zu installierenden Computer alle notwendigen .NET Plattformen oder Erweiterungen die für die Offline-Module verwendet wurden, bei der Programmierung, automatisch zur Verfügung gestellt und installiert werden.

### 3.3 XML-Schnittstelle

Für den Datenaustausch mit der XML-Schnittstelle werden natürlich zwei Teilnehmer gebraucht. Ein Teilnehmer ist der PHP Webserver auf dem die online Verwal-

---

<sup>24</sup> LAN ist ein Rechnernetz, das die Ausdehnung von Personal Area Networks übertrifft und ist dabei in seiner Ausdehnung ohne Zusatzmaßnahmen auf 500 Meter beschränkt

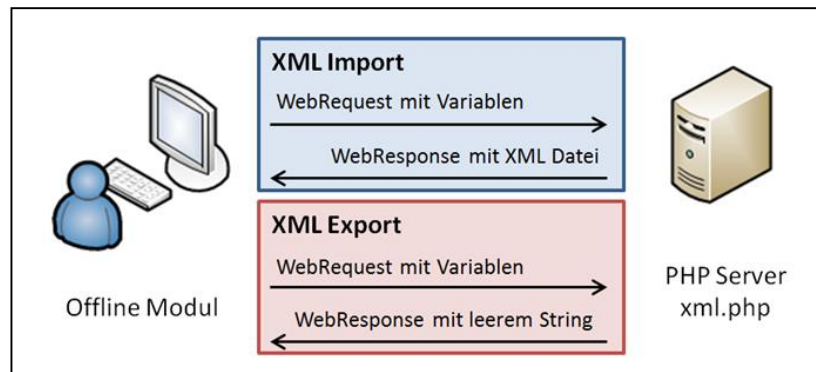
<sup>25</sup> VPN dient dazu, Teilnehmer eines privaten (in sich geschlossenen) Netzes an ein anderes privates Netz zu binden

tungssoftware läuft und der andere Teilnehmer ist der Benutzer eines Offline-Moduls. Gestartet wird der Datenaustausch indem der Anwender im C# Programm entweder auf XML-Import oder XML-Export drückt und dadurch eine Anfrage an den PHP Server per SSL stellt. Die Anfrage wird direkt an die zuständige Datei `xml.php` geschickt. Diese Anfrage schickt je nachdem welche Funktion verwendet wurde vier oder fünf Variablen via HTTPS POST an das Skript `xml.php`. Die Variablen sind die folgenden: Benutzername, Kennwort, Offline-Modul, Art der Anfrage und die generierte XML-Datei. Die XML-Datei die dabei erstellt wird würde dann folgenden Aufbau haben:

```
<?xml version="1.0" encoding="utf-8"?>
<xmlsync>
 <document>
 <renr>41100010</renr>
 <kdnr>10700001</kdnr>
 <datum>2011-07-15</datum>
 <betrag>1200</betrag>
 <firma>Matthias Thumer, Comtraq (ATU 6215 6008)</firma>
 <vorname>Matthias</vorname>
 <nachname>Thumer</nachname>
 <zhd>IT Abteilung</zhd>
 <email>max@beispiel.com</email>
 <landkurz>A</landkurz>
 <plz>6020</plz>
 <ort>Innsbruck</ort>
 <strasse>An-der-Lanstrasse 28b</strasse>
 <landlang>Austria</landlang>
 <zahlung>
 <renr>41100010</renr>
 <nr>1</nr>
 <datum>2011-07-26</datum>
 <art>Überweisung</art>
 <wert>200</wert>
 <xmlid>1311700499</xmlid>
 <xmldatum>2011-07-26 17:14:59</xmldatum>
 </zahlung>
 <zahlung>
 <renr>41100010</renr>
 <nr>2</nr>
 <datum>2011-07-27</datum>
 <art>Überweisung</art>
 <wert>100</wert>
 <xmlid>1311700499</xmlid>
 <xmldatum>2011-07-27 11:03:15</xmldatum>
 </zahlung>
 <aktion>
 <befehl>delete</befehl>
 <datum>2011-07-25 11:03:15</datum>
 <detail>1311737599</detail>
 </aktion>
 </document>
</xmlsync>
```

Abbildung 3-2: Aufbau der zu generierenden XML-Datei

Der Datenaustausch wird immer von den Offline-Modulen aus angestoßen durch betätigen einer der XML-Funktionen in der Applikation. Dabei handelt es sich um XML-Import oder XML-Export, dies wird in Abbildung 3-3 graphisch dargestellt.



**Abbildung 3-3: Graphische Darstellung des Datenaustausches**

Als Erstens prüft das PHP Skript ob der Benutzername und das Kennwort stimmen, wenn sie nicht stimmen, dann würde das Skript gar nichts zurückgeben. Natürlich wird der Benutzername und das Kennwort nicht in Klartext übermittelt, diese werden zusätzlich mit MD5<sup>26</sup> (Message-Digest Algorithm 5) verschlüsselt. Als zweiter Schritt kommt die Prüfung der Anfragenart, also ob es eine XML-Import (Server2Client) oder eine XML-Export (Client2Server) Anfrage ist, sowie die Festlegung um welches Offline-Modul es sich handelt um die entsprechende Logik für den Datenaustausch zu verwenden. Bei dem XML-Import generiert das PHP Skript eine XML Datei die das Offline-Modul dann verarbeitet. Wenn es sich um eine XML-Export Anfrage handelt dann schickt der Client auch die generierte XML Datei mit der Anfrage mit und die wird dann vom PHP Skript verarbeitet.

### 3.4 Datenbankkonzept

Grundsätzlich sollte die Datenbankstruktur gleich bleiben, da die Datensätze von der Online-Verwaltungssoftware in die Offline-Module synchronisiert werden. Um nicht eine große Änderung in der Struktur der Datenbank in der bereits bestehenden Verwaltungssoftware durchzuführen. Also die neuen Offline-Module sollten mit möglichst geringen Aufwand in das bestehende System integriert werden.

Um eine vernünftig funktionierende Synchronisierung zustande zu bringen müssen eigene Felder bzw. Spalten in den Tabellen der Datenbank hinzugefügt werden. Diese Felder müssen in der bereits bestehenden Verwaltungssoftware und in der Microsoft Access Datenbank der Offline-Module hinzugefügt werden.

Da man gleichzeitig auf beiden Softwareprodukten arbeiten kann, also in der online und offline Software, kann es klarerweise vorkommen, dass der gleiche Datensatz bearbeitet wird. Dieses Problem muss die Synchronisierungsschnittstelle bewältigen können. Deswegen wird das Feld `xmldatum` eingeführt, indem bei jeder Änderung

<sup>26</sup> MD5 ist eine weit verbreitete kryptographische Hashfunktion, die aus einer beliebigen Nachricht einen 128-Bit-Hashwert erzeugt

ein sogenannter `timestamp` hinterlegt wird. Damit kann man immer genau feststellen welcher Datensatz der Neueste ist.

Zusätzlich verwenden beide Datenbanken, MySQL und Microsoft Access, als Primärschlüssel eine Integer-Nummer die von den jeweiligen Datenbanken von selbst vergeben wird, wenn man neue Datensätze hinzufügt. Deswegen hat jeder Datensatz ein anderes System für die Nummerierung bzw. eine andere Identifikationsnummer für den jeweiligen Datensatz. Um dann genau zu wissen welcher Datensatz in der einen Datenbank zu vergleichen ist muss noch ein Feld `xmlid` in beiden Datenbanken hinzugefügt werden. Damit wird in jeder Datenbank für eine unverwechselbare Identifikation für einen Datensatz vergeben und die Synchronisierungsschnittstelle kann diese verwenden um die Datensätze zu vergleichen bzw. zu identifizieren.

Darüber hinaus muss die Synchronisierungsschnittstelle auch das Problem lösen wenn ein Datensatz gelöscht wird und dieser aber erst zu einem späteren Zeitpunkt synchronisiert wird. Deswegen wird eine neue Tabelle `tblxmlbericht` in beiden Softwarevarianten eingeführt.

In dieser Tabelle werden alle Synchronisierungen protokolliert und auch die Datensätze eingetragen die jeweils gelöscht werden durch die XML-Schnittstelle.

So wird einfach bei jedem Löschvorgang, von einem Datensatz, in diese Tabelle ein Eintrag geschrieben mit dem aktuellen Datum in das Feld `datum`, in das Feld `aktion` kommt der Text `delete`, die `xmlid` wird in das Feld `detail` geschrieben und in das Feld `dokumentid` wird hinterlegt zu welchem Dokument der Löschvorgang gehört.

Die Tabelle `tblxmlbericht` hat folgende Struktur:

**Tabelle:** `tblxmlbericht`

<i>Spaltenname</i>	<i>Datentype</i>	<i>Extra</i>
ID	int(11)	auto_increment
aktion	varchar(50)	
datum	datetime	
detail	varchar(255)	
dokumentid	varchar(50)	

**Tabelle 3-1: Struktur der Tabelle `tblxmlbericht`**

Diese Daten werden dann beim Generieren der XML-Datei in das Element `<document>` mit dem eigenem Element `<aktion>` und den Daten eingefügt, wie auch in der vorangehenden Abbildung 3-2 dargestellt wurde.

## 4 Implementierung

In diesem Kapitel werden die Schritte erläutert, die für die Implementierung der XML-Synchronisierungsschnittstelle und der Offline-Module sowie die nötigen Datenbankerweiterungen.

### 4.1 Erweiterung der MySQL-Datenbank

Zur Implementierung und Erweiterung bzw. Analyse der MySQL Datenbank wurde freiverfügbare Software MySQL Workbench 6.1 verwendet. Diese Software wird direkt vom Datenbankhersteller unter der GPL<sup>27</sup> (General Public License) zur Verfügung gestellt, damit wird ermöglicht, dass die Endnutzer, wie Privatpersonen und Firmen, die kostenlos Software zu nutzen können.

Natürlich kann man auch direkt die Datenbank mit SQL bearbeiten wie zuvor in dem Grundlagen Kapitel, aber dabei kann es sehr leicht passieren, dass man den Überblick verliert.

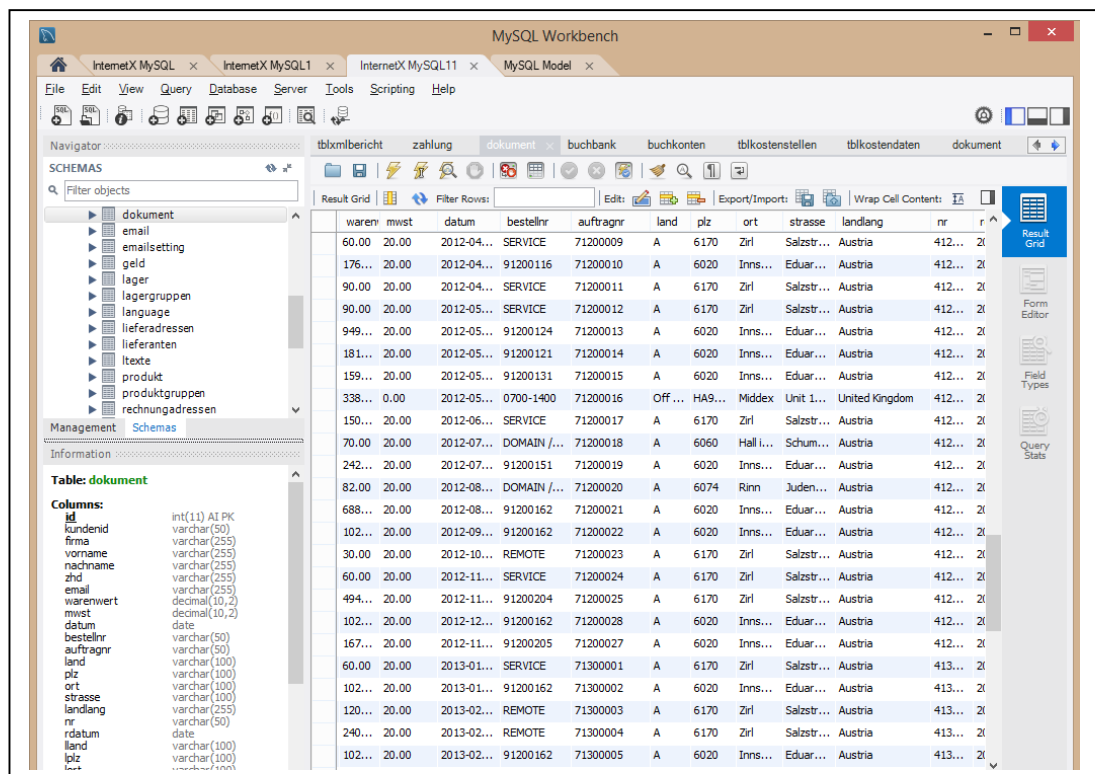


Abbildung 4-1: Übersicht von MySQL Workbench

<sup>27</sup> GPL ist die am weitesten verbreitete Software-Lizenz, welche den Endnutzern die Freiheiten garantiert, die mit ihr lizenzierte Software nutzen, studieren, verbreiten und ändern zu dürfen

Mit MySQL Workbench hat man die Möglichkeit einen sehr guten Überblick über die zu bearbeitende Datenbank zu bekommen wie man in Abbildung 4-1 sieht und es ist auch komfortable damit zu arbeiten.

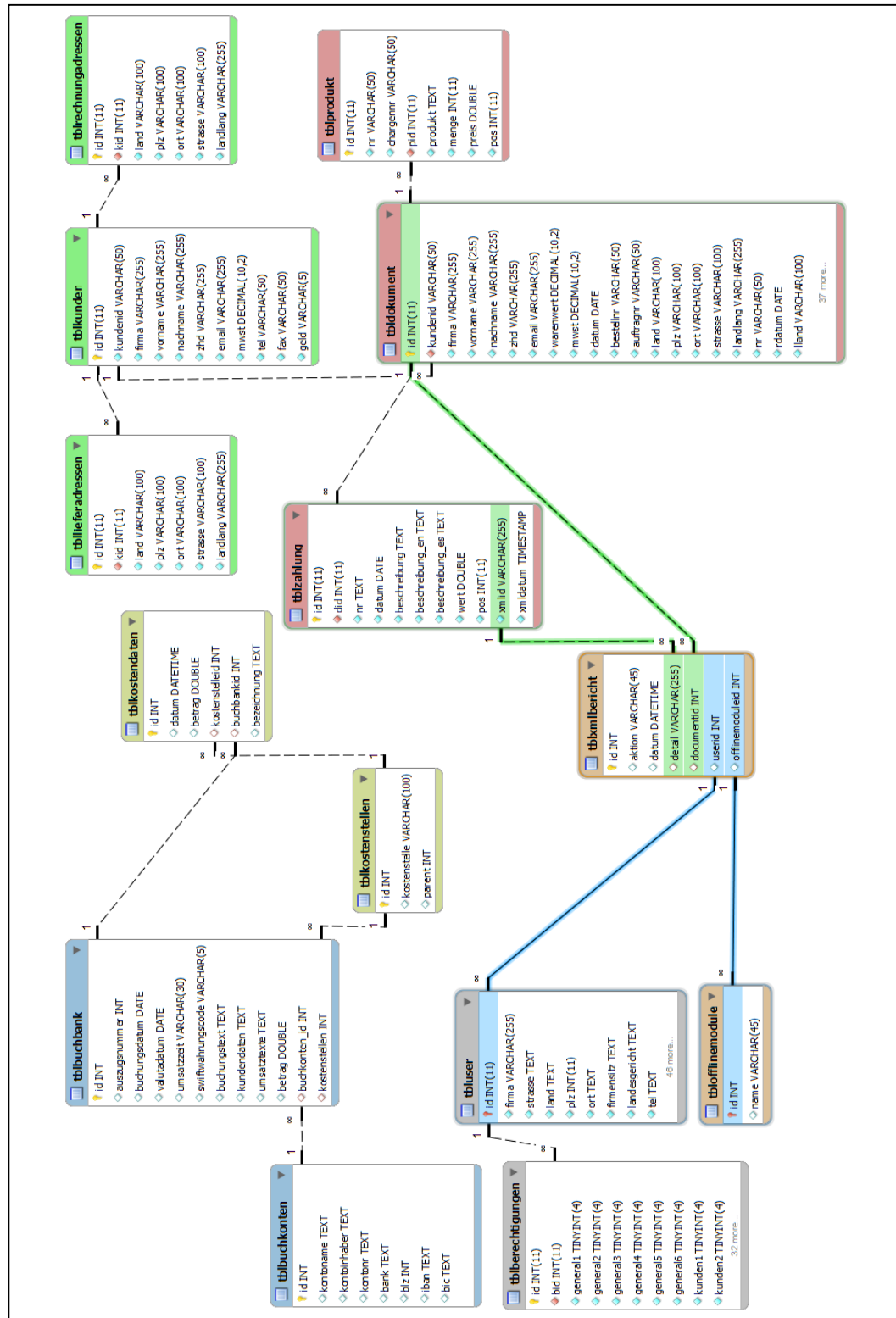


Abbildung 4-2: Übersicht der Beziehungen wichtiger Tabellen

Man hat dann in der Software auch ein Datenbankmodell angelegt (Abbildung 4-2), das man sich zuvor designet hat anhand des Systemkonzepts und der Überlegungen für die Synchronisierung und der Erläuterungen im vorhergehenden Kapitel. Dieses Modell sollte dazu dienen, die benötigte Datenbankarchitektur visuell darzustellen. Da die Visualisierung aller Tabellen zu umfangreich wäre und auch nicht mehr übersichtlich hat man sich nur auf die wichtigsten Tabellen die im direktem Zusammenhang mit den Offline-Modulen stehen entschieden.

#### 4.1.1 Hinzufügen der nötigen Metadaten

Nach dem man sich den ersten Überblick verschafft hat, hat man die Tabelle `tblxmlbericht` in der online Verwaltungssoftware Datenbank angelegt. Die Tabelle wurde mit folgendem SQL-Befehl erstellt, der durch die MySQL Workbench ausgeführt wurde.

- ```
CREATE TABLE IF NOT EXISTS `tblxmlbericht` (  
  `id` INT(11) NOT NULL AUTO_INCREMENT,  
  `aktion` VARCHAR(45) NOT NULL,  
  `datum` DATETIME NOT NULL,  
  `detail` VARCHAR(255) NOT NULL,  
  `documentid` INT(11) NOT NULL,  
  `userid` INT(11) NOT NULL,  
  `offlinemoduleid` INT(11) NOT NULL,  
  PRIMARY KEY (`id`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

Dieser Vorgang wurde dann natürlich auch in der Microsoft Datenbank durchgeführt, da ja auch im Offline-Module selber auch die verschiedenen Aktionen wie zum Beispiel ein Löschvorgang protokolliert werden muss um diesen dann auch später zu synchronisieren und in der MySQL Datenbank durchzuführen.

Hierfür muss man lediglich auf die Registerkarte Erstellen in der Gruppe Tabellen und auf Tabellen klicken. Dabei wird eine neue Tabelle in der Datenblattansicht geöffnet in der man dann auf Tabellenentwurf gehen kann und die einzelnen Metadaten hinzufügen kann. Man kann in einer Access Datenbank auch Tabellen erstellen mit einem SQL-Befehl wobei man aber nicht genau den gleichen wie bei einer MySQL Datenbank verwenden kann, da die Datentypen unterschiedlich sind zwischen den beiden Datenbanken. In Abbildung 4-3 kann man die für Microsoft Access verwendeten Datentypennamen sehen.

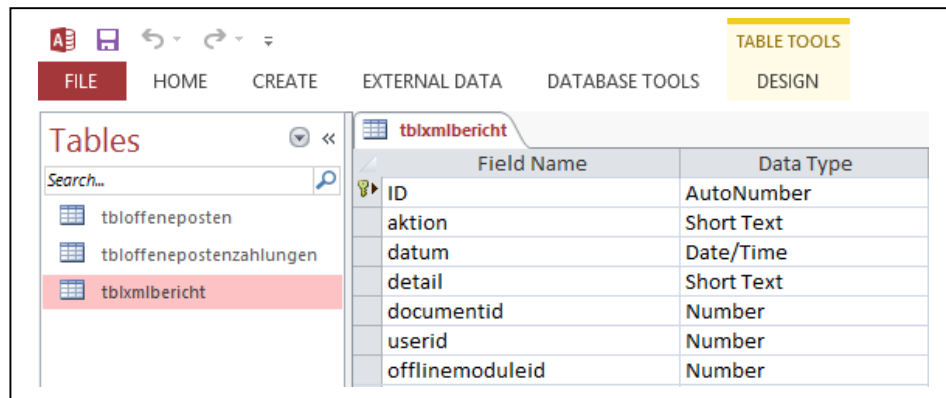


Abbildung 4-3: Tabellenentwurf mit Microsoft Access

Nachdem die komplett neue Tabelle erstellt wurde musste man dann noch einige Tabellen mit neuen Feldern versehen die durch Verwendung der XML-Synchronisierungsschnittstelle nötig wurden.

Das hinzufügen neuer Spalten oder Metadaten in bereits vorhandenen Tabellen ist immer eine etwas kniffligere Angelegenheit. Durch das hinzufügen neuer Spalten in eine Tabelle kann es nämlich möglich sein, dass man bereits bestehende SQL-Abfragen sowie SQL-Befehle die in Verwendung sind auch erweitern muss je nachdem wie sie entworfen wurden bzw. wie sie auf die einzelnen Spalten bereits zugreifen.

Glücklicherweise mussten die meisten der SQL-Befehle nicht abgeändert werden nachdem man sie einzeln kontrolliert hat und auf ihre Funktionalität in der online Verwaltungssoftware geprüft hat. Diese Aufgabe wurde relativ reibungslos mithilfe der Zend Studio Software gemeistert, da diese es ermöglicht PHP Skripte zu Debuggen und Ursachenanalyse zu betreiben. Weiterführende Informationen zum Zend Studio gibt es unter [ZeSt2014].

Die tatsächliche Erweiterung einer bestehen Tabelle, wie der Tabelle `tblzahlung`, wurde in der MySQL Datenbank mit folgenden SQL-Befehl durchgeführt:

- ```
ALTER TABLE `tblzahlung` (
 ADD `xmlid` VARCHAR(255) NOT NULL,
 ADD `xmldatum` TIMESTAMP NOT NULL,
```

Dieser SQL-Befehl wurde bei mehreren Tabellen ausgeführt sowie auch in der Microsoft Access Datenbank, wobei man dort wieder im Tabellenentwurf die Felder hinzugefügt hat, wie in Abbildung 4-4 abgebildet.

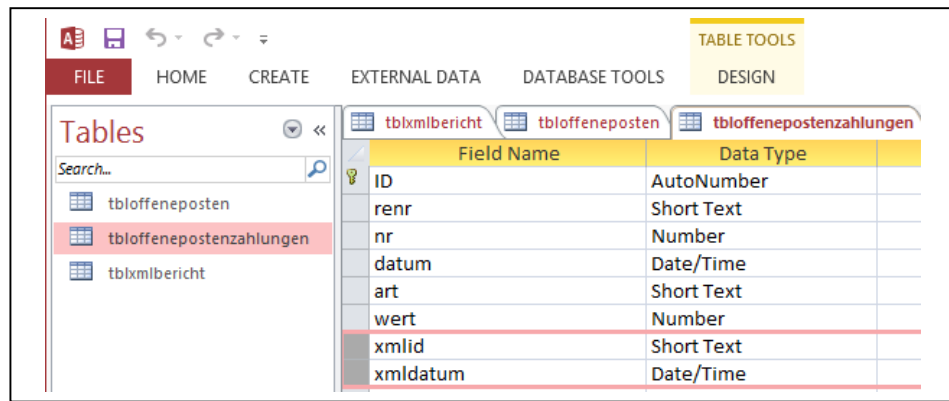


Abbildung 4-4: Tabellenerweiterung in Microsoft Access

Auch hier sieht man wieder, dass die Datentypen unterschiedlich sind und dass es in Access keinen spezifischen Datentyp `timestamp` gibt wie in MySQL.

#### 4.1.2 Ergänzung der bestehenden Datensätze

Wie von der Aufgabenstellung vorausgesetzt werden nicht alle Datensätze manuell nachgetragen mit den Werten der neuen Felder. Diese werden erst bei Bedarf durch die C#, also das Offline-Module, oder das PHP Skript, die online Verwaltungssoftware, erzeugt.

Bei der Erzeugung der Werte durch C# oder das PHP Skript kommt es schon zu Unterschiede der verschiedenen Programmiersprachen, da es nicht die gleichen Funktionen gibt. Aus diesem Grund wurde für das generieren der Werte des Feldes `xmlid` in C# eine Interpretation der PHP Funktion `microtime()` geschrieben.

Der C# Code für die Erstellung der `xmlid` Werte lautet wie folgt:

```

• DateTime dt1 = new DateTime(1970, 1, 1);
 DateTime dt2 = DateTime.Now;
 TimeSpan ts1 = new TimeSpan(dt2.Ticks - dt1.Ticks);
 String xmlid = Convert.ToInt32(ts1.TotalSeconds).ToString();

```

Dieses Code Segment gibt die Anzahl an vergangenen Sekunden zurück seit 1. Januar 1970 um 00:00:00 Uhr im Datentype `String` wie, zum Beispiel 1388534400 was das Datum 01.01.2014 00:00:00 wäre.

Dagegen gibt die PHP Funktion `microtime()` einen Wert wie folgt zurück 0.6011388534400, da bei dieser Funktion auch noch die Mikrosekunden in Sekunden angegeben die vergangen sind seit 1. Januar 1970.

Der Unterschied der zwei verschiedenen Varianten hilft sogar in diesem Verwendungszweck, da in diesem Fall somit garantiert werden kann, dass es unmöglich ist für beide Systeme jemals eine gleiche `xmlid` zu erzeugen.

Auf Grund, dass die Werte erst ergänzt werden durch die Offline-Module bzw. die XML-Synchronisierungsschnittstelle oder die Verwaltungssoftware muss man auch Rücksicht nehmen auf die verwendete Zeichenkodierung. Da durch Verwendung unterschiedlicher Zeichenkodierungen Sonderzeichen nicht richtig angezeigt werden können bzw. gespeichert werden in der Datenbank.

Die MySQL Datenbank ist in der Zeichenkodierung UTF-8<sup>28</sup> (Universal Character Set Transformation Format 8-bit), dies kann man auch erkennen bei der Erstellung der Tabelle durch folgendes SQL Segment:

- `ENGINE=InnoDB DEFAULT CHARSET=utf8;`

Microsoft Access verwendet dagegen die für Windows Standard-Kodierung ISO<sup>29</sup>-8859-1 (International Organization for Standardization).

Verschiedene Zeichenkodierungen in den Datenbanken verursachen kein Problem solange man sich bewusst ist welche Kodierung wo verwendet wird und man die entsprechenden Konvertierung vornimmt wenn nötig.

## 4.2 XML-Synchronisierungsschnittstelle

Wie schon zuvor erläutert wurde die Schnittstelle mit einer zwei Methodenfunktion entwickelt. Dabei kommt es an ob es sich um einen Datenimport oder einen Datenexport handelt. Bei dem Datenimport muss eine XML-Datei ausgelesen werden und dann in die Datenbanken importiert werden bzw. Ändern übernommen werden. Auf der anderen Seite muss beim Datenexport eine XML-Datei aus der Datenbank heraus erstellt werden.

Diese zwei Anfragearten wurden in dem Systemkonzept als Client2Server und Server2Client bezeichnet und wurden einmal mit einem PHP Skript und einmal in C# direkt im Offline-Module implementiert.

---

<sup>28</sup> UTF-8 ist die am weiteste verbreitete Kodierung für Unicode-Zeichen und hat den Status als Standard-Zeichenkodierung des Internets und damit verbundener Dokumententypen

<sup>29</sup> ISO ist die internationale Vereinigung von Normungsorganisationen und erarbeitet internationale Normen in einer Vielzahl von Bereichen

Der Zugriff auf die MySQL Datenbank vom PHP Skript heraus wird wie folgt durchgeführt:

- `$dbcnx = mysql_connect("localhost", "user", "password");`  
`mysql_select_db("database", $dbcnx);`

Von einem PHP Skript heraus eine Verbindung zu einer MySQL Datenbank zu erstellen ist sehr gut integriert in die PHP Skriptsprache und dadurch stehen auch viele brauchbare Funktionen zur Verwendung.

Zum Verbindungsaufbau für eine Microsoft Access Datenbank in C# wurde hingegen folgendes Kodeselement verwendet:

- `string s = @"Provider=Microsoft.Jet.OLEDB.4.0;`  
`Data Source=C:\Program Files\Software\database.mdb;`  
`Jet OLEDB:Database Password=password;"`  
`con = new OleDbConnection(s);`  
`con.Open();`

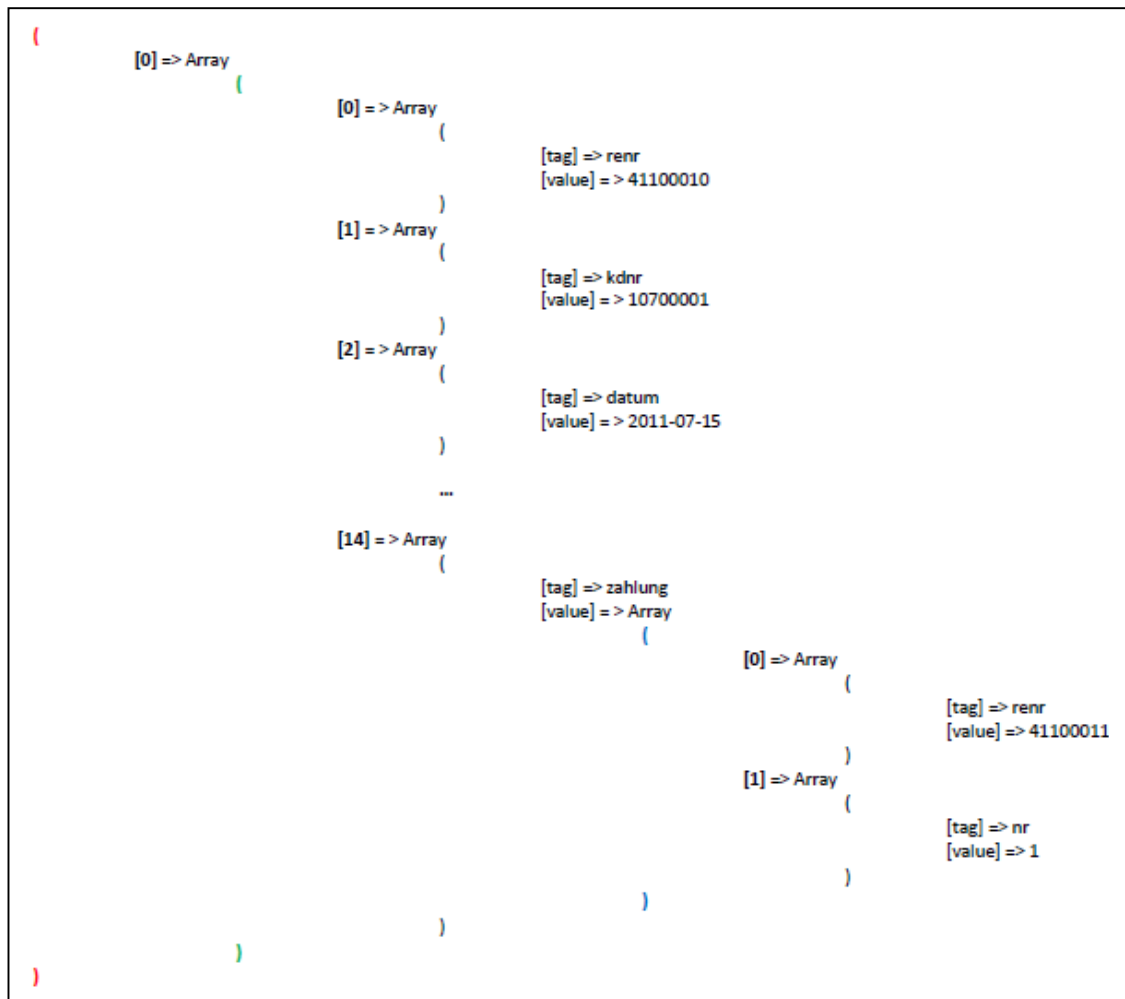
Man kann auch sehen, dass man für den Zugriff auf die lokale Microsoft Access Datenbank auch ein Kennwort braucht das in der EXE-Datei des Offline-Modules kompiliert ist, um somit das Kriterium zu erfüllen, dass man keinen direkten Zugriff auf die Rohdaten hat.

Natürlich gibt es viele verschiedene Möglichkeiten eine Verbindung in C# sowie auch in PHP herzustellen, aber die Verwendeten sind die meistverbreiteten.

#### 4.2.1 Erstellen des PHP-Skripts

Für das bessere Verständnis der Logik die in dem PHP-Skript verbaut wurde, werden die wichtigsten Elemente herausgehoben und genauer erklärt. Dabei handelt es sich vor allem um Vorgehensweise bei einem Datenimport von der Seite des Servers gesehen.

Um die erforderlichen Daten bearbeiten zu können müssen diese von der gesendeten XML-Datei in ein brauchbares Format für die Bearbeitung umgewandelt werden. Dabei hat man sich für einen Assoziierten-Array, der eine Struktur hat wie in Abbildung 4-5 dargestellt wird, entschlossen, da man dadurch die benötigte Flexibilität der Datenblöcke bekommt um die nötige Logik umsetzen zu können.



**Abbildung 4-5: Darstellung von einem Assoziierten-Array**

Nachdem die gesendete XML Datei in den Assoziierten-Array eingelesen wurde, wird dieser mittels For-Schleifen durchlaufen und mit den Datensätzen der MySQL Datenbank verglichen. Es wird dabei jeder Datensatz mit der gleichen `xmlid` miteinander verglichen und die Datensätze mit dem aktuellerem Datum im Feld `xmldatum` werden übernommen. Das ist auch so, wenn es darum geht einen ganzen Datensatz zu löschen, also wenn in der Tabelle `tblxmlbericht` im Feld `aktion` der Wert `delete` steht wird dieser zuvor auch mit dem Datum überprüft.

Im folgendem Teilsegment des PHP Skripts kann man auch erkennen wie vorteilhaft die Variante ist mit dem Array, da man damit viel einfacher und präziser auf die benötigten Werte zugreifen kann und diese mit den Werten die aus der MySQL Datenbank kommen gegenüberstellen kann. Dabei kann man von einer Zeile auf eine andere Zeile sowie auch Spalte im Array springen indem man lediglich die Nummer zwischen den Eckigen Klammern ändert. In diesem Fall wird aber die Zeile durch die For-Schleife festgelegt durch die Variable `$d` und die Spaltennummer manuell deklariert.

- ```

for($d = 0;$d < count($assoc);$d++)
{
    //get data from database from tbldocument
    $query = "SELECT * FROM tbldokument WHERE nr = '". $assoc[$d][
'value'] [0] ['value'] . "'";
    $result = mysql_query($query, $dbcnx) or die();
    $rows = @mysql_num_rows($result);

    $dbdocumentid = "";

    if ($rows != 0)
    {
        while ($line = mysql_fetch_array($result, MYSQL_ASSOC))
        {
            $dbdocumentid = $line["id"];
        }

        //loop all documents
        for($a = 0; $a < count($assoc[$d]['value']);$a++)
        {
            //loop all attributes of document
            if ($assoc[$d]['value'][$a]['tag'] == "zahlung")
            {
                //loop all zahlungen
                //get data from database from tblzahlung
                $zquery = "SELECT * FROM tblzahlung WHERE did = '".
                $dbdocumentid . "' AND xmlid = '". $assoc[$d]['value'][$a
                ] ['value'] [5] ['value'] . "'";
                $zresult = mysql_query($zquery, $dbcnx) or die();
                $zrows = @mysql_num_rows($zresult);

                if ($zrows != 0)
                {
                    while ($zline = mysql_fetch_array($zresult,
                    MYSQL_ASSOC))
                    {
                        if($zline["xmldatum"] < $assoc[$d]['value'][$a
                        ] ['value'] [6] ['value'])
                        {
                            //update zahlung in database with xml data
                            switch(utf8_decode($assoc[$d]['value'][$a
                            ] ['value'] [3] ['value']))
                            {
                                case "Überweisung":

```

In dem Teilsegment kann man auch sehen, dass die unterschiedlichen Zeichenkodierungen berücksichtigt wurden durch Verwendung der Funktion `utf8_decode()`, die dazu dient eine als UTF-8 kodierte ISO-8859-1-Zeichenkette in eine einfache ISO-8859-1-Zeichenkette zu konvertieren.

4.2.2 Implementierung in die Offline-Module

Bei der Umsetzung in die Offline-Module hat man zuerst sichergestellt, dass man eine sichere Verbindung aufbaut zum Server der Verwaltungssoftware. Dabei hat man bewährte .Net Klassen verwendet wie `WebRequest` und `WebResponse` für die Kommunikation, die es auch ermöglichen ein SSL Zertifikat in C# zu validieren.

```
• WebRequest request = WebRequest.Create("https://secure.comtraq.com/[REDACTED]");

// Validation of SSL Certificate
ServicePointManager.ServerCertificateValidationCallback +=
new System.Net.Security.RemoteCertificateValidationCallback(ValidateRemoteCertificate);

request.Method = "POST";

string postData = "[REDACTED]";

byte[] byteArray = Encoding.UTF8.GetBytes(postData);

request.ContentType = "application/x-www-form-urlencoded";
request.ContentLength = byteArray.Length;

Stream dataStream = request.GetRequestStream();
dataStream.Write(byteArray, 0, byteArray.Length);
dataStream.Close();

WebResponse response = request.GetResponse();

dataStream = response.GetResponseStream();
StreamReader reader = new StreamReader(dataStream, Encoding.UTF8);
string responseFromServer = reader.ReadToEnd();

MessageBox.Show(responseFromServer.ToString());
```

Im oberen C#-Code kann man auch wieder sehen, dass die Zeichenkodierung berücksichtigt wird durch die Funktion `Encoding.UTF8` wie auch in dem PHP Skript. Die Validierung des SSL Zertifikates wird durch das Unterprogramm `ValidateRemoteCertificate` durchgeführt. Durch die schwarzen Balken werden die genauen Variablendeklarationen und die exakte URI des verwendeten Webservices ausgeblendet.

Nachdem die Verbindung geprüft und hergestellt wurde sowie die Authentifizierung, werden nun die übermittelten Daten aufgearbeitet. Damit meint man, dass die Daten auch wie in PHP in eine Struktur gebracht werden müssen mit der man gut weiterarbeiten kann. Dabei wird in C# ein sogenanntes `DataSet` verwendet das eine vergleichbare Struktur hat wie ein Assoziierter-Array. Man könnte die Struktur eines `DataSet` durchaus mit der einer relationalen Datenbank vergleichen.

Auch hier werden dann die einzelnen Zeilen mithilfe mehrerer Schleifen durchlaufen und die entsprechenden Bedingungen angewendet die für die Server2Client Anfrageart notwendig sind.

Nicht wie im PHP Skript wird hier eine ForEach-Schleife verwendet um die Datensätze zu durchlaufen, wie man auch folgend sehen kann:

- ```
foreach (DataRow drow in xmlsync.Tables["document"].Rows)
{
 bool found = false;
 //loop all documents in local database
 foreach (DataRow dbrow in tblOffenePostenDT.Rows)
 {
 if (dbrow.ItemArray[0].ToString() == drow.ItemArray[0].ToString())
 {
 //loop all attributes for document except renr
 dbrow[1] = drow.ItemArray[1]; //kdnr
 dbrow[2] = drow.ItemArray[2]; //datum
 dbrow[3] = drow.ItemArray[3].ToString().Replace(".",
CultureInfo.CurrentCulture.NumberFormat.NumberDecimalSeparator); //betrags
 dbrow[4] = drow.ItemArray[4]; //firma
 dbrow[5] = drow.ItemArray[5]; //vorname
 dbrow[6] = drow.ItemArray[6]; //nachname
 dbrow[7] = drow.ItemArray[7]; //zhd
 dbrow[8] = drow.ItemArray[8]; //email
 dbrow[9] = drow.ItemArray[9]; //landkurz
 dbrow[10] = drow.ItemArray[10]; //plz
 dbrow[11] = drow.ItemArray[11]; //ort
 dbrow[12] = drow.ItemArray[12]; //strasse
 dbrow[13] = drow.ItemArray[13]; //landlang

 tblOffenePostenBS.EndEdit();

 int read = 0;
 read = DownloadFile("https://secure.comtraq.com/");
 found = true;
 }
 }
 if (found == false)
 {
 DataRow ndr = tblOffenePostenDT.NewRow();
 ndr[0] = drow.ItemArray[0];
 ndr[1] = drow.ItemArray[1];
 ndr[2] = drow.ItemArray[2];
 ndr[3] = drow.ItemArray[3].ToString().Replace(".",
CultureInfo.CurrentCulture.NumberFormat.NumberDecimalSeparator);
 ndr[4] = drow.ItemArray[4];
 ndr[5] = drow.ItemArray[5];
 ndr[6] = drow.ItemArray[6];
 ndr[7] = drow.ItemArray[7];
 ndr[8] = drow.ItemArray[8];
 ndr[9] = drow.ItemArray[9];
 ndr[10] = drow.ItemArray[10];
 ndr[11] = drow.ItemArray[11];
 ndr[12] = drow.ItemArray[12];
 ndr[13] = drow.ItemArray[13];

 tblOffenePostenDT.Rows.Add(ndr);

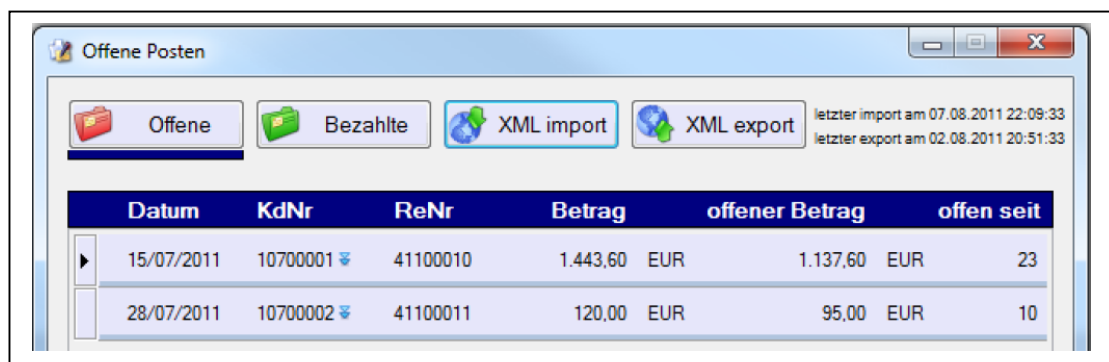
 int read = 0;
 read = DownloadFile("https://secure.comtraq.com/");
 }
}
```

Durch Verwendung der Microsoft Access Datenbank bei den Offline-Modulen muss auch darauf geachtet werden, dass man bei den SQL-Befehlen auch die gleiche Formatierung verwendet wie auf dem Computer auf dem die Anwendung ausgeführt wird. Da es zum Beispiel bei Zahlen mit Kommastellen leicht zu Schwierigkeiten kommen kann.

Man muss also wenn man den Wert 1.000,00 bei einem Computer mit US Amerikanischer Formatierung in die Access Datenbank schreiben muss, das Format auch entsprechend im SQL-Befehl umgesetzt sein. Dies wird mit der .NET Klasse `CultureInfo` gelöst, dabei werden die Regionalen Einstellungen von dem jeweiligen Computer ausgelesen. Dadurch wird auch sichergestellt, dass die Anwendung reibungslos auf mehrsprachigen Windows-Betriebssystemen läuft.

Die Bedienung der XML-Synchronisierung wurde gezielt einfach gehalten in den Modulen mit jeweils einen Knopf zum Importieren und Exportieren der Daten. Dabei braucht der Anwender aber keine Acht geben in welcher Reihenfolge die Synchronisierung durchgeführt wird, also ob zuerst ein Export oder ein Import gemacht wird.

In Abbildung 4-6 kann man das GUI des Offene Posten Modules sehen und auch das Datum der letzten Synchronisierung.



**Abbildung 4-6: Anzeige vom letzten Import bzw. Export Vorgang**

Um dem Anwender eine bessere Übersicht über den Fortschritt des Datenaustausches zu bieten wurde auch das Steuerelement `ProgressBar` hinzugefügt, die nur angezeigt wird wenn man den XML-Import oder XML-Export Vorgang betätigt hat wie man in Abbildung 4-7 sehen kann.

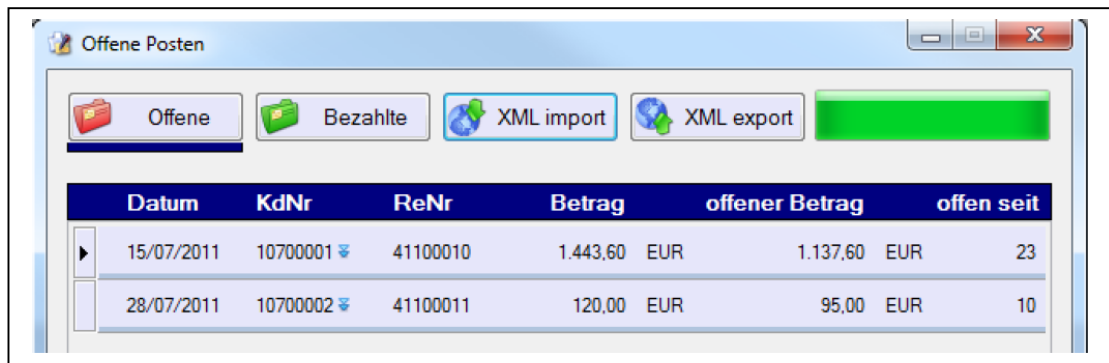


Abbildung 4-7: Fortschrittsanzeige mit dem Steuerelement ProgressBar

## 5 Ergebnisse und Ausblick

Im abschließenden Kapitel werden die bisher gewonnenen Ergebnisse zusammengefasst und eine Bewertung der Leistung aus Sicht des Autors vorgenommen. Ein Ausblick zeigt Weiterentwicklungspotenziale auf.

### 5.1 Ergebnisse

Das Diplomarbeitsthema ergab sich aus einer Anforderung der Comtraq GmbH, die Funktionalitäten der firmeninternen Verwaltungssoftware bereitzustellen mit sogenannten Offline-Modulen. Diese Offline-Module welche für die Kommunikation mit der firmeninternen Verwaltungssoftware über eine XML-Schnittstelle kommunizieren schafften die Grundlage dieser Diplomarbeit.

Nach der Findung des Themas wurden die Grundlagen der Datenbanken und Ihre Standards erläutert. Dabei wurden verschiedene Datenbankmodelle vorgestellt und ihre grundlegende Funktion erklärt. Anschließend wurde für das Aufgabenverständnis, die Verwaltungssoftware mit einigen Funktionen erklärt.

Anschließend wurde das Systemkonzept für die ganze Datenkommunikation über die XML-Synchronisierungsschnittstelle vorgestellt. Dazu wurde auch die Datenbankarchitektur beziehungsweise Datenbankerweiterungen aus softwaretechnischer Sicht erläutert. Weil die erfolgreiche Umsetzung des Projektes die Verwendung verschiedener unterstützender Werkzeuge voraussetzt wurden diese technischen Hilfsmittel kurz angeführt und beschrieben. Aufgrund der ständigen Weiterentwicklung wurde auch das Konzept für die Softwareaktualisierung vorgestellt durch die Verwendung des ClickOnce-Systems.

Bei den verwendeten Datenbanken wurde auf den vorhanden aufgebaut und dabei auch keine zusätzlichen Lizenzkosten verursacht. Sowie auch bei den Programmiersprachen und technischen Hilfsmittel war es nicht nötig neue Anschaffungen zu tätigen.

Abhängig von dieser Entscheidung wurde für die Offline-Module die Programmiersprache C# mit einer Microsoft Access Datenbank verwendet und der Webservice wurde in PHP mit der MySQL Datenbank entwickelt. Als grundlegendes Sicherheitskonzeptes wurde ein Datenaustausch über HTTPS auf Basis einer Benutzer-Authentifizierung verwendet.

Zur Implementierung der Änderungen wurde die Software MySQL Workbench für die Datenbankänderungen sowie für die Analyse verwendet. Bei der Entwicklung

der Offline-Module wurde Microsoft Visual Studio und für das PHP Skript wurde die Entwicklungsumgebung Zend Studio eingesetzt.

Zum Anschluss erfolgten die Tests der Datensynchronisierung über unterschiedliche Internet-Anbindungen. Dabei konnten die einzelnen Aufgabenstellungen geprüft werden beziehungsweise durchgeführt werden.

## **5.2 Bewertung der Arbeit**

Im Rahmen der Diplomarbeit ist es gelungen alle geforderten Kriterien zu erfüllen. Des Weiteren konnten die Grundlagen von Datenbanken, SQL und ihre Standards erläutert werden.

Zusammenfassend lässt sich sagen, dass durch die kontinuierliche Zunahme der Mobilen Mitarbeiter, vor allem deren im Ausland, sich die Firma auf dem strategisch richtigen Wege befindet. Durch die Verwendung von Offline-Module mit der Kombination der Synchronisierungsschnittstelle lassen sich nicht nur teure Internetkosten (Roaming-Gebühren) sparen, sondern sie können praktisch diesen Lösungsansatz beziehungsweise Schnittstelle überall einsetzen.

Den größten Vorteil bringt aber mit Sicherheit das im Rahmen der Diplomarbeit erstellte Systemkonzept für die Zusammenarbeit der XML-Synchronisierungsschnittstelle und der verschiedenen Datenbanken. Durch diesen Schritt ist ein guter Grundstein gelegt worden um den Gedanken des Offline-Arbeiten voranzutreiben. Zusätzlich möchte ich noch darauf hinweisen, dass auch andere Firmen die Möglichkeit anbieten für ihre Online Produkte einen sogenannten Offline-Mode oder Offline-Cache zu verwenden um das Produkt ohne Internetzugang zu verwenden.

## **5.3 Ausblick**

Um eine zukunftsorientierte Lösung zu erstellen, sollte der Webservice in mehrere Bereiche unterteilt werden. Da jedes Offline-Module seine eigene Logik braucht und alle in einem Webservice stecken kann es bei der Erweiterung der Modulanzahl dazu führen, dass die Reaktionszeiten zu stark anwachsen. Hierbei könnte die Unterteilung der Verwaltungssoftware auch im Webservice übernommen werden.

Man sollte die Sicherheitsvorkehrungen weiter entwickeln und auch die Datensicherheit am lokalen Computer der Anwender überprüfen. Unter diesem Aspekt könnte die lokale Datenbank verschlüsselt werden, dabei muss man natürlich auf die Leistung achten, dass das System nicht zu langsam wird.

Ein weiterer interessanter Punkt wäre die Weiterentwicklung zur Unterstützung verschiedener Datenbanken in den Offline-Modulen. Dies könnte von Bedeutung werden bei der Implementierung von Modulen mit einem größeren Datenaufkommen.

# Literatur

- [HaMe2001] Harold, Elliotte Rusty; Means, W. Scott: XML in a Nutshell – 1. Auflage: O'Reilly & Associates Verlag, 2001
- [GeFr2011] Geisler, Frank: Datenbanken Grundlagen und Design – 4. Auflage: Huthig Jehle Rehm GmbH Verlag, 2011
- [ElSh2005] Elmasri, Ramez; Navathe Shamkant B.: Grundlagen von Datenbanksystem - 3. Auflage: Addison-Wesley Verlag, 2005
- [BoAx2006] Borntträger Axel: MySQL 5. Für Professionals - 1. Auflage: Franzis Verlag, 2006
- [BaTh2010] Bartosch, Oliver; Throll Marcus: Einstieg in SQL - 4. Auflage: Galileo Computing Verlag, 2010
- [FePri2002] Feuerstein Steven, Pribyl Bill: Oracle PL/SQL Programming -3. Auflage: O'Reilly & Associates, 2002
- [KüAn2011] Kühnel Andreas: Visual C# 2010 Das umfassende Handbuch – 5. Auflage: Galileo Computing Verlag, 2011
- [MyWo2014] MySQL Workbench 6.1: ist ein konsolidiertes visuelles Werkzeug für Datenbankarchitekten, Entwickler und DBAs -  
URL: <http://www.mysql.com/products/workbench/>, 2014
- [ZeSt2014] Zend Studio 8: stellt professionellen PHP-Entwicklern die Tools zur Verfügung, Probleme schneller zu beheben. -  
URL: <http://www.zend.com/de/products/studio> , 2014

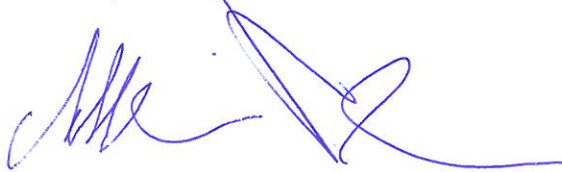
# Eidesstattliche Erklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Literatur und Hilfsmittel angefertigt habe.

Stellen, die wörtlich oder sinngemäß aus Quellen entnommen wurden, sind als solche kenntlich gemacht.

Diese Arbeit wurde in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegt.

Innsbruck, den 20. Mai 2014



Ing. Matthias Thumer