

Die Eignung von Open Source Datenbanken in Vereinen

Betreuung durch Andreea Ionas, Diplomassistentin für Wirtschaftsinformatik

Universität Fribourg

Mai 2004

Abstract

Nicht nur Unternehmen, sondern auch Vereine haben eine Datenverwaltung, die sie ständig aktualisieren müssen. Im Gegensatz zu kommerziellen Organisationen beschäftigen sie meist kein ausgebildetes Personal und keine besondere Infrastruktur um die ganze Administration zu bewältigen. Die Vereine haben aus diesen zwei Gründen spezifische Bedürfnisse, was einerseits finanzielle und andererseits benutzerfreundliche Aspekte anbelangt.

In dieser Arbeit wird ein Lösungsvorschlag für eine einfache Adressverwaltung eines Vereins entwickelt. Im ersten Teil wird untersucht, welche Software sich für ein Projekt dieser Art eignet. Dann wird die Adressverwaltung ausschliesslich mit Open-Source Software realisiert und soll zur besseren Übersichtlichkeit in ein Web-Interface eingebettet werden.

Nicht zuletzt werden einige Sicherheitsaspekte betrachtet, wozu auch ein Lösungsvorschlag gemacht wird.

Keywords: Open Source Datenbanken, MySQL, PostgreSQL, Firebird, Datenverwaltung von Vereinen

INHALTSVERZEICHNIS

1	<u>EINLEITUNG</u>	3
1.1	PROBLEMSTELLUNG	3
1.2	ZIELSETZUNG	3
1.3	VORGEHENSWEISE	4
2	<u>AUSGANGSLAGE DER VEREINE</u>	4
2.1	WIE DIE VEREINE DAS PROBLEM HEUTE LÖSEN	5
2.2	BEDÜRFNISSE DER VEREINE	5
2.3	DIE SICHERHEIT DER DATEN	7
3	<u>OPEN-SOURCE PROGRAMME</u>	7
3.1	DIE GESCHICHTE VON OPEN SOURCE	7
3.2	THE OPEN SOURCE DEFINITION	8
3.3	WAS SIND OPEN-SOURCE-PROGRAMME?	9
4	<u>OPEN SOURCE DATENBANKEN</u>	9
4.1	MYSQL 4.1	10
4.1.1	VORTEILE VON MYSQL	10
4.1.2	NACHTEILE VON MYSQL	11
4.2	FIREBIRD 1.5	11
4.2.1	VORTEILE VON FIREBIRD	12
4.2.2	NACHTEILE VON FIREBIRD	12
4.3	POSTGRES SQL 7.3.4	13
4.3.1	VORTEILE VON POSTGRES SQL	13
4.3.2	NACHTEILE VON POSTGRES SQL	13
4.3	VERGLEICH	14
5	<u>LÖSUNGSVORSCHLAG EINES DATENBANKSYSTEMS FÜR EINEN VEREIN</u>	15
5.1	KONZEPT	15
5.1.1	DATENBANKKONZEPT	15
5.1.2	WEB-INTERFACE	17
5.2	WAHL DER DATENBANK	18
5.3	IMPLEMENTIERUNG	18
5.3.1	MYSQL	18
5.3.2	PHP UND GRAPHISCHE OBERFLÄCHE	19
5.4	SICHERHEIT UND SCHUTZ DER DATEN	20
6	<u>SCHLUSSFOLGERUNG UND ZUSAMMENFASSUNG</u>	21
7	<u>QUELLENVERZEICHNIS</u>	22
	<u>ANHANG</u>	I

1 Einleitung

Jeder Verein, gleich aus welchem Interessegebiet, hat eine Reihe von Verwaltungsaufgaben von der Aktualisierung der Adressbestände, Finanzen bis zur Koordination der Aktivitäten zu erfüllen. Um eine gute Übersicht zu haben und die Zusammenhänge leicht zu erkennen ist zum Teil ein grosser Verwaltungsaufwand erforderlich. In dieser Arbeit wird untersucht, wie man den Administrationsaufwand im Bereich der Adressverwaltung zeitlich und finanziell minimieren kann.

[Als Erleichterung für den Leser wird Programmcode in dieser Schrift hervorgehoben.]

1.1 Problemstellung

- Die Vereine verwalten ihre Adressen, haben dabei aber beschränkte finanzielle Mittel und wenig Informatikkenntnisse. Ein aktueller und auf eine Version reduzierter Adreßbestand ist jedoch für Informationen und Rechnungen wichtig.
- Um Mitgliederzahl konstant zu halten müssen sich Vereine der Zeit anpassen. Zum Beispiel bereitstellen einer Homepage als Informationsmittel für Mitglieder und Interessierte.
- Der Sicherheitsaspekt wird oft aus Kostengründen oder Unwissen vernachlässigt.

1.2 Zielsetzung

- Gibt es eine benutzerfreundliche, aber trotzdem sichere Lösung für Vereine?
- Gibt es geeignete Open Source Programme?
- Anforderungskatalog für Vereine und ihre Adressverwaltung
- Vergleich mit 3 Open Source Datenbanken, welche Datenbank eignet sich für ein Projekt in diesem Umfang?
- Wie gehen die Vereine mit Fragen zur Sicherheit um?

Die Fragen sollen anhand eines Lösungsvorschlags für die Pfadi Altenstein Heiden beantwortet werden.

1.3 Vorgehensweise

- Anhand einer Internetumfrage soll folgendes beantwortet werden:
Über welche Programme verfügen die Vereine bereits, wie nutzen sie ihre Adressverwaltung? Welche Bedürfnisse und Wünsche haben sie?
- Was ist Open Source und wie sehen die Anwendungsmöglichkeiten aus?
- 3 Open Source Datenbanken im Vergleich: Aussagen über die Bedienungsfreundlichkeit und wie geeignet sie für ein Projekt dieser Größe und Art ist.
- Implementierung einer geeigneten Datenbank mit einer bedienerfreundlichen Oberfläche via Web-Interface
- Vergleich mit der bestehenden Literatur und der Umfrage

2 Ausgangslage der Vereine

Alle Aussagen im folgenden Kapitel basieren auf einer Internetumfrage. Insgesamt wurden 40 Vereine aufgefordert an der Umfrage teilzunehmen, es kamen Antworten von 18 Adressverantwortlichen aus den Bereichen Jugendvereine, Musikvereine, politische Vereine und Sportvereine zurück (Abb. 1). Die Umfrage sollte zeigen wie Vereine ihre Adressen heute verwalten und welche Bedürfnisse mit dem bestehenden System nicht befriedigt werden.

Die Umfrage ist nicht repräsentativ, jedoch sind Tendenzen erkennbar, welche aufschlussreich für die weitere Arbeit sein werden.

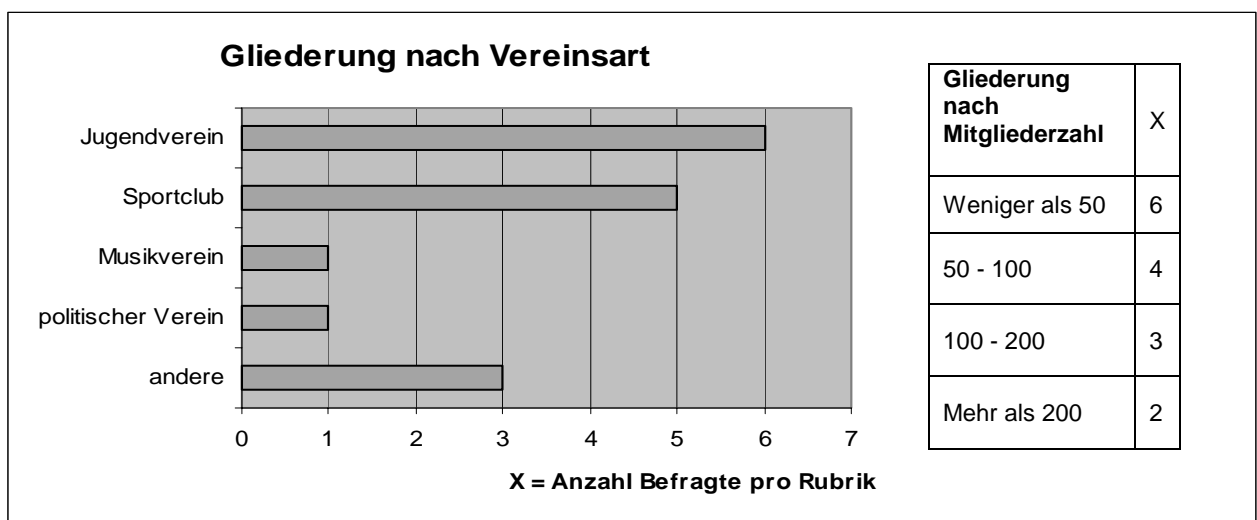


Abb. 1: „Aufteilung der Befragten“

2.1 Wie die Vereine das Problem heute lösen

Alle befragten Vereine verwalten ihre Daten elektronisch. Mehr als die Hälfte, nämlich 63 %, haben die Adressen in einer Excel-Tabelle gespeichert. Die andere Hälfte hat eine Datenbank angelegt, wobei 4 von 6 auf kommerzielle Produkte vertrauen. Nur zwei Vereine gaben an, mit Open-Source Datenbanken zu arbeiten.

Die Vereine nutzen die Datenbank bzw. die Excel-Tabellen für sehr unterschiedliche Aufgaben.

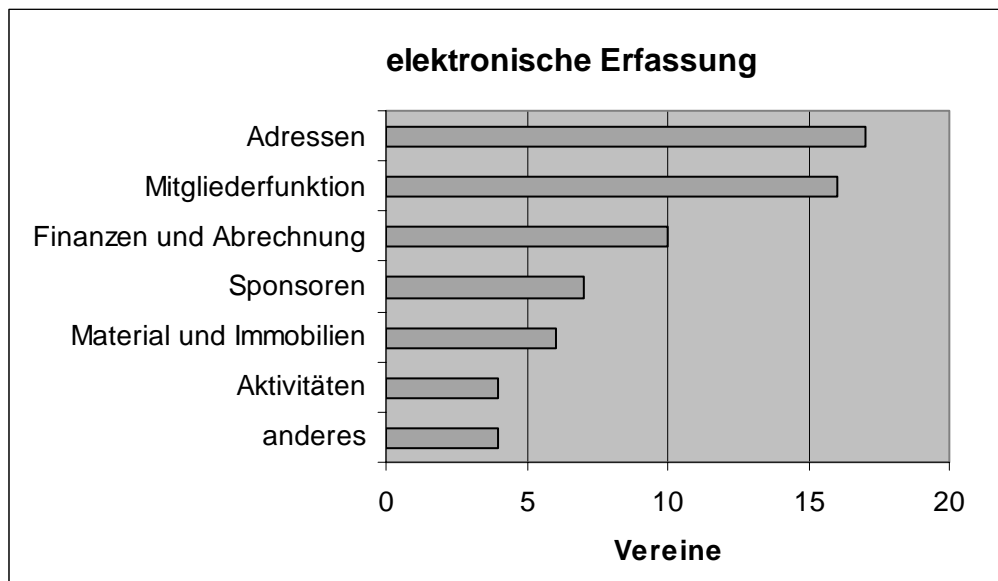


Abb. 2: „elektronische Erfassungsgebiete der Vereine“

Wie in Abb. 2 ersichtlich, ist der wichtigste Bereich der Datenerfassung die Adressverwaltung. Alle Vereine nutzen die Informatikmittel auch für andere Aufgabenbereiche wie zum Beispiel die Klassierung nach der Mitgliederfunktion. Daraus ist sofort ersichtlich ob es sich um ein Mitglied des Vorstands oder um einen Ehemaligen handelt.

2.2 Bedürfnisse der Vereine

Auch wenn die Vereine bereits viele Aufgabenbereiche auf dem Computer verwalten, gibt es immer noch einige Lücken und Verbesserungsmöglichkeiten.

Viele Vereine wünschen sich eine benutzerfreundlichere Verwaltung (Abb. 3) gegenüber Nicht-Informatikern. Die Adressbestände sollten von verschiedenen Personen bearbeitet werden können, ohne dass nachher zwei verschiedene Versionen vorhanden sind. Das ist mit den häufig benutzten Excel-Tabellen nicht möglich.

Zur Benutzerfreundlichkeit gehört, dass Zusammenhänge einfach und richtig dargestellt werden können. Ein Lösungsvorschlag wäre ein relationales Datenbanksystem mit referenzierten Tabellen.

Das Ziel dieser Arbeit ist es die Zusammenhänge einfacher darzustellen und die Benutzerfreundlichkeit zu verbessern. Ein einfaches Finanzsystem für Vereine zu erstellen ist nicht Gegenstand dieser Arbeit, obwohl es durchaus ein wichtiges und interessantes Innovationsgebiet ist.

Niemand hat den Wunsch die Änderungs-, Lösch- oder Einfügefunktion zu verbessern. Die Vereine konnten bis jetzt alle Funktionen ausführen, jedoch sollen die Funktionen jetzt benutzerfreundlicher werden.

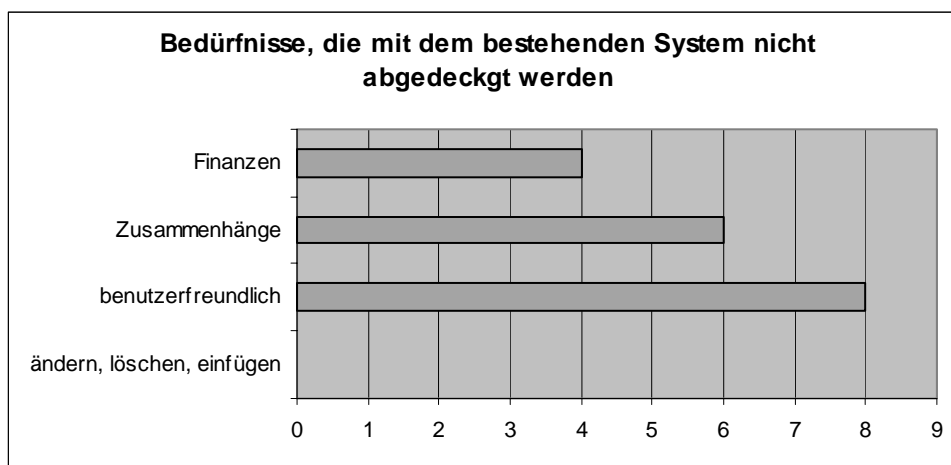


Abb. 3: „Verbesserungsmöglichkeiten“

Zum Thema Open Source schrieb in der Umfrage jemand eine interessante Bemerkung:
„Im wesentlichen hängt der Einsatz von Open Source-Software vor allem von der zuständigen Person ab, ist er ein Open-Source Fan, wird solche eingesetzt ansonsten nicht.

Für kleine Vereine ist das Argument Preis sowieso irrelevant, werden doch kurzerhand Schwarzkopien benutzt oder das Ganze läuft auf lizenzierten Geschäftscomputern (wie in unserem Fall).

Unweigerlich zieht z.B. Open Source auf Linux auch den entsprechenden Desktop nach. Das heisst, ich muss allen Personen die damit arbeiten müssen auch gleich ein Linux installieren...(ausgenommen ich hab eine Webanwendung)“

Da kommerzielle Software zurzeit grosse Investitionskosten verursachen, liegt es nahe mit Schwarzkopien zu arbeiten. Die Open Source Alternative ist nach wie vor weniger verbreitet und man benötigt die Zeit und das Interesse mit ihr die Aufgaben zu lösen. Mit der Web-Anwendung wird schlussendlich das Problem der Plattformenabhängigkeit und der Bedienungsproblemen gelöst.

2.3 Die Sicherheit der Daten

Die meisten Befragten haben um die Sicherheit und den Schutz der Daten keine Bedenken. Sie haben die Datenverwaltung in ihrem Privatcomputer oder Geschäftscomputer gespeichert und somit sind die Daten übers Internet nicht zugänglich. Ausserdem sind mehr als ein Viertel der Ansicht, dass sie in ihrer kleinen Adresskartei keine relevanten Daten verwalten. Als Absicherung bei einem Datenverlust dient eine Sicherheitskopie auf einer anderen Festplatte oder einem Ausdruck auf Papier.

Trotzdem gibt es einige Vereine, die sich Gedanken über den Bereich der Sicherheit machen. Sie ziehen zum Beispiel in Betracht, die Adressen der Verantwortlichen nicht mehr öffentlich zugänglich zu machen und/oder sie schränken den Nutzerkreis gezielt ein, um die Gefahr von Diebstahl und Missbrauch der Daten zu minimieren.

3 Open-Source Programme

Der Begriff Open-Source wird nicht überall gleich verwendet. Grundsätzlich ist zwischen Open-Source Software und Freeware zu unterscheiden. Freeware heisst, dass Programme kostenlos zur Verfügung stehen. Eine Open-Source Software zeichnet sich dadurch aus, dass das Programm und der Quellcode für alle Nutzer kostenlos zugänglich ist. So ist der Nutzer in der Lage den Quellcode zu lesen und zu verändern.

3.1 Die Geschichte von Open Source



Die Idee von freier Software begann mit der Entstehung der Computer.

An den Universitäten wurden die Programmierer für ihre Arbeit bezahlt. Jedoch die Programme wurden weitgehend frei weiterverbreitet. Erst als die Computer auch die Geschäftswelt eroberten, begannen die Programmierer ihre Produkte zu lizenzieren und erhoben eine Gebühr für jede Kopie [Perens 1999, S.172].

1984 gründete Richard Stallman die „Free Software Foundation“ und das GNU Projekt. Er entwickelte eine Lizenz für freie Software namens General Public License (GPL) mit dem Gedanken Software mit den geringsten Kosten und ohne Aufwand zu verbreiten. Über Open Source sagt Stallman: „*It has nothing to do with price. It is about freedom*“. Mit der Freiheit meint er die Möglichkeit, dass jeder Besitzer das Programm verändern und verteilen darf, die Distribution muss jedoch nicht zwingend kostenlos sein.

Zur gleichen Zeit definierte Eric Raymond Open-Source als frei zugängliche, also kostenlose Software. Die zwei Ansichten lösten eine heftige Auseinandersetzung aus.

Bis sich am 3. Januar 1998 sich wichtige Informatiker trafen, darunter auch Eric Raymond, um endlich den Begriff Open Source genauer zu beschreiben. Von nun an sollte man anhand der Open Source Definition entscheiden können, ob sich eine Software Open Source nennen darf. [O'Reilly & Associates 1999]

3.2 The Open Source Definition

Die Definition ist keine Software Lizenz, sondern zeigt nur auf, was eine Open Source Lizenz beinhalten muss. Jeder Punkt muss beachtet werden.

1. Freie Weiterverbreitung

Jeder Besitzer eines Open Source Programms, kann beliebig viele Kopien davon machen. Der Weiterverbreitung stehen keine Lizenzgebühren im Wege.

2. Quellcode

Der Quellcode muss vollständig zugänglich sein. Man muss mindestens die Möglichkeit haben den Quellcode aus dem Internet herunterzuladen. Der Quellcode muss in der Form vorliegen, damit ein Programmierer ihn verändern kann. Absichtliche Verwirrungen im Quellcode sind nicht erlaubt.

3. Auf dem Programm basierende Werke

Die Lizenz muss die Veränderung des Programms, auf dem Programm basierende Werke sowie deren Verbreitung unter den gleichen Lizenzbedingungen gestatten. Man muss den Quellcode nicht nur lesen dürfen, sondern auch die Möglichkeit haben ihn zu verändern.

4. Die Unversehrtheit des Originalcodes

Der Autor des Originalcodes kann Einschränkungen der Weiterverbreitung machen. Die Lizenz muss jedoch die Weiterverbreitung ausdrücklich erlauben, auch wenn es von der Originalsoftware einen verschiedenen Namen oder Versionsnummer tragen muss. Das heisst, dass nur Netscape ihr Programm Netscape Navigator nennen darf, während alle anderen Versionen Mozilla oder anders genannt werden.

5. Keine Diskriminierung von einzelnen Personen oder Gruppen

Die Lizenz darf keine Personen oder Gruppen diskriminieren oder von der Benutzung des Programms ausschliessen.

6. Keine Einschränkungen für bestimmte Anwendungsbereiche

Das Programm kann für kommerzielle und politische Zwecke aller Art genutzt werden. Die Lizenz setzt bei der Anwendung keine Grenzen.

7. Verbreitung der Lizenz

Mit der Verbreitung des Programms muss auch die Lizenz weitergeben werden, damit die Software wirklich frei bleibt.

8. Die Lizenz darf nicht für ein bestimmtes Produkt gelten

Es ist verboten, ein Programm so zu verändern, damit man es nur noch mit einem bestimmten Programm eines Herstellers benutzen kann.

9. Die Lizenz darf andere Software nicht beeinträchtigen

Andere Software darf zusammen mit Open Source Software nicht beeinträchtigt werden. Es darf nicht verlangt werden, dass auf einem Datenträger mit Open Source Software nicht auch kommerzielle Programme gespeichert werden können. [Perens 1999, S. 174]

3.3 Was sind Open-Source-Programme?

Open Source wurde definiert um einen gemeinsamen Namen für Lizenzen zu benutzen, der Freiheit im Sinne der Wissenschaft verkörpert [O'Reilly & Associates 1999]. Im Gegensatz zu kommerziellen Programmen, die in Form von ausführbaren Programmen verteilt werden, erfolgt die Distribution der Open Source Software oft über den Sourcecode. Nur in dieser Form sind Veränderungen am Programm möglich.

Viele Benutzer ändern das Programm nach ihren Bedürfnissen und Vorlieben. Einige Anwender werden zu Mitentwicklern, sie verbessern das Programm und testen dies auch gleich selbst aus. Die vorgeschlagenen Korrekturen und Verbesserungen werden dann von den Herstellern ins Programm aufgenommen oder der Entwickler veröffentlicht seine neue Version gleich selbst. Die Vorzüge von Open Source Programmen sind eine effiziente Problembehandlung und eine schnelle Weiterentwicklung, so dass die Programme besser und stabiler funktionieren. Das Betriebssystem Linux und der Internetbrowser Netscape wurden in den letzten Jahren immer populärer, beide Programme verfügen über eine Open Source Lizenz. Nach Bruce Perens [Perens 1999, S.171] werden dadurch sogar Programme mit restriktiveren Lizenzen verdrängt.

Die freie Weiterverbreitung ist nur dank den tiefen Materialkosten einer Kopie des Programms möglich. Bei der Herstellung von anderen Produkten z.B T-Shirts, fallen Kosten an, die mit dem Verkaufspreis gedeckt werden müssen.

4 Open Source Datenbanken

Mittlerweile gibt es Open Source Programme für fast jeden Anwendungsbereich.

In diesem Kapitel werden drei Open Source Datenbanken genauer untersucht und verglichen. Es soll aufgezeigt werden, wo die Stärken und Schwächen eines Datenbanksystems liegen und weshalb sich ein bestimmtes Produkt für Anwendungen in einem Verein eignet.

Wenn von einem Produkt gesprochen wird, bezieht sich die Aussage auf die jeweilige Version des Programms.

4.1 MySQL 4.1



MySQL ist die bekannteste und am weitesten verbreitete Open Source Datenbank. Das Produkt eignet sich für den Einsatz von kleinen und mittleren Datenbanken, vor allem im nichtkommerziellen Anwendungsbereich. Studenten und Forscher schätzen vor allem den Mehrbenutzerbetrieb, es können also mehrere Computer (Benutzer) gleichzeitig auf die Datenbank zugreifen. Die Datenbank funktioniert unter anderem auf den Betriebssystemen Linux und Windows.

MySQL erfüllt nicht ganz alle Anforderungen der Open Source Definition. Der Punkt 6 (siehe Kap. 3.2) wird nicht eingehalten, nur ältere Versionen sind kostenlos erhältlich, während aktuelle MySQL-Versionen für den kommerziellen Einsatz kostenpflichtig sind. Die Lizenzgebühren sind im Vergleich zu professionellen Produkten jedoch um einiges tiefer [Pollakowski 2003, S.17].

Um die Bearbeitung der Datensätze zu vereinfachen sind kostenlose, graphische Oberflächen erhältlich zum Beispiel MySQL-Front (bis V 2.5). So wird es möglich, ähnlich wie mit MS-Access eine Datenbank zu erstellen, Datensätze abzufragen und zu bearbeiten.

SQL, die Abfrage- und Manipulationssprache von MySQL, wird in den meisten DBMS benutzt. Die Sprache entspricht dem SQL-Standard, welcher unter anderem ISO- und ANSI-zertifiziert ist.

MySQL ist zugleich auch eine Web-Datenbank. Von einer Webseite kann man die Datenbank mit PHP kontaktieren und so einfach die Daten abrufen, auch Einfüge-, Lösch- und Änderungsvorgänge können übers Internet getätigt werden.

In der 4-er Version werden Views nur beschränkt, Stored Procedures und Triggers nicht unterstützt. MySQL kündigt aber für die Folgeversionen ab 5.0 bzw. 5.1 die Unterstützung dieser Funktionen an.

4.1.1 Vorteile von MySQL

- MySQL ist eine, der am weitesten verbreiteten Open Source Datenbanken. Zurzeit sind viele Bücher über dieses Thema erhältlich, natürlich sind auch unzählige Informationen über das Internet abrufbar. Von Dokumentationen bis Newsgroups zu MySQL gibt es alles.
- Die Datenbank ist einfach zu installieren. Bei Linux, z.B. 'Red Hat', ist die Datenbank bereits in der Installation integriert. Unter Windows wird man mit einem Installationsassistenten Schritt für Schritt geführt.

- MySQL ist ein vollwertiges relationales Datenbankmanagementsystem.
- Dank gesetzten Indizes kann die Abfragezeit optimiert werden. Häufig abgefragte Datensätze sollten als Indexe definiert werden. Bei einer Abfrage werden nur die ersten Buchstaben, anstatt der ganze Eintrag, verglichen um einen Datensatz eindeutig zu identifizieren. Trotzdem sollte man nicht alle Merkmale als Index definieren, denn das braucht zusätzlichen Speicherplatz und Updates dauern länger.
Ein Beispiel: Eine häufige Abfrage ist der Name und der Wohnort.

```
KEY namecity (surname (10), firstname (3), city (2))
```

Jetzt werden nur die ersten 10 Buchstaben des Nachnamen, die ersten drei des Vornamen und die ersten zwei der Stadt miteinander verglichen.

- Die Popularität von MySQL ist auf das gute Zusammenspiel mit PHP zurückzuführen. Die Integration von MySQL-Anweisungen in PHP-Programme ermöglichen schnelle Abfrageergebnisse in Web-Anwendungen.

4.1.2 Nachteile von MySQL

- MySQL unterstützt keine referentielle Integrität, d.h. es wird nicht überprüft, ob zu einem Fremdschlüssel der passende Primärschlüssel wirklich existiert. Zur Hilfe kann man bereits beim definieren der Tabelle die Anweisung geben, dass die beiden Tabellen referenziert werden. z. B.

```
CREATE TABLE buch (bnr, titel, autor int REFERENCES autor.id)
```

Die Nummer des Autors wird nun mit der ID der Autorentabelle referenziert. Existiert die ID in dieser Tabelle nicht, so kann der neue Bucheintrag nicht gemacht werden.

- Mit MySQL kann man keine verschachtelten Abfragen machen.
- MySQL hat eine beschränkte Auswahl an Features, welche aber in den Folgeversionen implementiert werden sollten.

4.2 Firebird 1.5



Ende 1999 plante die Firma Borland (damals noch Inprise) ihre Datenbank Interbase aufzugeben. Auf Druck einer Interessengemeinde und dem Erfolg von Linux und anderen Open Source-Projekten, stellte die Firma den Quellcode ihrer Datenbank frei zur Verfügung.

Doch bald darauf beschloss Borland den kommerziellen Vertrieb von Interbase wieder aufzunehmen, bereits haben sie Version 6.5 und 7 herausgegeben.

Firebird 1.0 war also ein direkter Open Source-Nachfolger von Interbase Version 6. Deshalb ist es nicht erstaunlich, dass Firebird heute noch viele Ähnlichkeiten zu seinem Stammprodukt hat. Die Entwickler legen nach wie vor grossen Wert auf die Kompatibilität der beiden Produkte, zumindest soll eine funktionale Schnittstelle beibehalten werden. So ist es für eine Unternehmung problemlos möglich vom Open Source Produkt auf das kostenpflichtige Produkt von Borland zu wechseln.

Ein Administrationstool ist nicht standardmassig enthalten, jedoch sind mehrere Administrationsumgebungen zu Firebird erhältlich, einige sind kommerziell, es gibt aber auch kostenlose Produkte.

Zu Firebird gibt es einige kommerzielle Supportanbieter wie HK Software (<http://www.h-k.de>) oder IBPhoenix (<http://www.ibphoenix.com>), welche Support, Training und Beratung anbieten. Daneben gibt es wie bei anderen Open Source Datenbanken kostenlosen Support durch die Community. Die „Firebird-Community“ zeichnet sich durch ihre aussergewöhnliche Aktivität im Support-Forum aus, welches mit 1000-1500 Einträgen pro Monat belegt wird [Emser 2004]

4.2.1 Vorteile von Firebird

- Firebird ist ein vollständiges RDBMS mit einer SQL-92 konformer Abfragesprache.
- Firebird hat viele Funktionen integriert. Das DBMS unterstützt Stored Procedures, Trigger und referentielle Integrität. Die Funktionen können mit UDF erweitert werden.
- Dank einem ausgeklügelten System ist es möglich, Back-ups während der Betriebszeit durchzuführen.
- Spiegelung der Datenbasis.

Die Daten werden im laufenden Betrieb in einer zweiten Datenbank parallel zur ersten nachgeführt. Firebird ist in der Lage mehrere solcher Datenbankspiegel gleichzeitig zu führen.

4.2.2 Nachteile von Firebird

- Das Firebird-RDBMS ist nicht leicht bedienbar, vor allem ohne Benutzeroberfläche wird es schwierig den Umgang mit Firebird schnell zu erlernen. Um ein Projekt mit einer Firebird-Datenbank starten zu wollen, braucht es mindestens einen erfahrenen Entwickler.
- Firebird ist auch im Bereich der Web-Applikationen geeignet. Wer allerdings keinen eigenen Server ins Netz stellen kann, wird Mühe haben einen Web-Hoster mit der Konfiguration für eine Firebird-Datenbank zu finden. Oft bleibt keine andere Wahl als zum Erstellen von Web-Applikationen auf MySQL zurückzugreifen [Emser 2004].

4.3 PostgreSQL 7.3.4



PostgreSQL ist eines der leistungsstärksten Open Source DBMS. Die Entwicklung von PostgreSQL wird von einer Forschergruppe koordiniert, welche nicht von einer bestimmten Firma kontrolliert wird. Das heutige PostgreSQL ist auf die Arbeit von unzähligen Studierenden und Programmierern zurückzuführen, die unter der Leitung des Professors Michael Stonebraker an der Universität von Berkeley, Kalifornien arbeiteten.

Der ursprüngliche Name der Software war Postgres. Er wurde 1996 auf PostgreSQL geändert, als ein Jahr zuvor die SQL-Funktionalität die PostQUEL-Abfragesprache ersetzte. [Momjian 2004]

PostgreSQL bietet viele Funktionen, die auch bei den kommerziellen DBMS vorhanden sind. Von den Open Source Datenbanken kommt PostgreSQL den kommerziellen Produkten am nächsten. Gerade wegen dem grossen Funktionsumfang ist die Datenbank vor allem für mittlere und grosse Projekte geeignet. Um alle Möglichkeiten auszuschöpfen braucht es jedoch schon einige Informatikkenntnisse.

4.3.1 Vorteile von PostgreSQL

- Im Vergleich zu MySQL ist PostgreSQL schneller und leistungsfähiger bei zeitgleichen Zugriffen und komplexen Abfragen. Die Geschwindigkeit ist vor allem bei grossen Projekten relevant, bei sehr kleinen Projekten kann PostgreSQL sogar langsamer sein.
- PostgreSQL bietet eine umfassendere Unterstützung des SQL-Standards, sowie generell mehr Features als MySQL. Zum Beispiel Views, Stored Procedures, Trigger.
- PostgreSQL unterstützt referentielle Integrität.

4.3.2 Nachteile von PostgreSQL

- Für PostgreSQL ist eine Unix-Umgebung notwendig. Das heisst, PostgreSQL läuft problemlos auf Linux. Unter Windows ist PostgreSQL in der Cygwin-Umgebung verfügbar, welche eine Unix-Umgebung simuliert. Windows-kompatible Versionen werden auch von einigen kommerziellen Firmen angeboten [The PostgreSQL Global Development Group 2004].
- Eine Datenbank in einer Cygwin-Umgebung zu benutzen, ist ziemlich aufwendig und überhaupt nicht benutzerfreundlich.

4.3 Vergleich

Um eine Datenbank zu installieren, braucht man einiges Informatikwissen. Meist ist die Installation schwieriger als bei einem kommerziellen Produkt.

Den Open Source Datenbanken wird nachgesagt, dass es an einem genügenden Support mangle oder dieser überhaupt nicht vorhanden sei und sie sich deshalb nur für kleine Anwendungen eignen. Doch die Vielfalt der Informationen und Hilfe für Open Source-Produkte im Internet, in Büchern oder durch Spezialisten ist genau so gross, wie bei kommerziellen Produkten. Somit sind auch erste Installationsschwierigkeiten zu überwinden.

Die 3 Produkte MySQL, Firebird und PostgreSQL setzen unterschiedliche Schwerpunkte. MySQL verdankt seine Popularität hauptsächlich den Webanwendungen im Zusammenhang mit der LAMP-Konfiguration (Linux, Apache, MySQL, PHP).

Firebird und PostgreSQL unterstützen dafür wichtige Funktionen wie Views, Stored Procedures, Trigger und referentielle Integrität. Die beiden Produkte zeichnen sich als volle RDBMS aus, wo SQL-92 als Standard gilt.

Firebird ist als Unternehmensdatenbank tauglich, wobei es aber durchaus geschulte und erfahrene Administratoren benötigt, um die Datenbank zu verwalten. Firebird bietet viele Möglichkeiten, wenn man mit den Eigenheiten des DBMS umzugehen weiss. Ein grosser Vorteil für die Benutzer ist die Kompatibilität von Firebird zu Interbase und die Möglichkeit, dass sie folglich die Borland-Werkzeuge benutzen können.

PostgreSQL bietet sehr viele Features und kann komplizierten Abfragen schnell bearbeiten. Dieses DBMS eignet sich deshalb eher für grosse Datenbanken zum Beispiel für wissenschaftliche Projekte.

Ein Verein legt meist eine eher kleine Datenbank mit einer relativ einfachen Struktur an. Dem Verein ist es wichtig, die Datenbank einfach verwalten zu können. Aus diesen Gründen eignet sich für Vereine, die meist nur ihre Adressen auf der Datenbank verwalten eine MySQL-Datenbank.

Weitere Gründe sprechen für die MySQL-Datenbank: Falls die Datenbank ins Internet integriert werden soll, bietet MySQL eine Ideale Voraussetzung. Die Einbindung der Datenbank ist mit PHP einfach vorzunehmen und viele Web-Hoster bieten eine Konfiguration für MySQL an. Mit den vielen Dokumentationen findet man einen leichten Einstieg in MySQL und PHP.

5 Lösungsvorschlag eines Datenbanksystems für einen Verein

Das Datenbanksystem soll ein Lösungsvorschlag für einen Verein sein, der ähnliche Strukturen wie die Pfadi Heiden aufweist. Es handelt sich um einen Jugendverein mit etwa 80 aktiven Mitgliedern. Der Verein möchte mit der Datenbank die Adressen verwalten, wobei man die Adressen nach Stufe und Funktion ordnen können soll. Siehe auch die Bedürfnisse der Vereine Kapitel 2.1.

Der Vorteil eines Datenbanksystems im Gegensatz zu einer „Dateikartei“ z. B. Excel, ist der zentrale Speicherort. So existiert nur eine aktuelle Version des Adressbestands, auf welchen mehrere Benutzer zugreifen können, aber nur eine Person die Berechtigung für den Löschvorgang hat.

5.1 Konzept

Um ein Konzept zu entwerfen, muss man die Anforderungen an das Datenbanksystem bestimmen. Danach macht man sich Gedanken über die Datenbankmodellierung. Zum Schluss wird die Datenbank in eine benutzerfreundliche Oberfläche gebettet.

Anforderungskatalog

- Die Pfadi möchte ihre Adressen mit einem Datenbanksystem verwalten
- Die Adressen können nach Funktion und Stufe geordnet werden. Es ist auch möglich nur einzelne Daten abzufragen.
- Bedienerfreundliche Oberfläche mit Eingabemaske zum Einfügen, Ändern und Löschen
- Die Datenbank ist übers Internet zugänglich, damit jeder Verantwortliche immer die aktuelle Version zur Verfügung hat.
- Über die Internetschnittstelle kann man pdf-files für Adresstiketten erstellen.
- Die wichtigsten Sicherheitsaspekte sollen berücksichtigt werden.
- Um die Verwaltung des Informatiksystems möglichst einfach zu halten wird Web-Server wie bei der Homepage verwendet.

5.1.1 Datenbankkonzept

Im nächsten Schritt wird ein Datenmodell nach den Ansprüchen des Vereins erstellt. Das Datenmodell zeigt die Struktur und den Aufbau der Datenbank auf. Das Datenmodell muss gründlich durchdacht sein, um spätere Änderungen, welche einen grossen zusätzlichen

Aufwand bedeuten, zu verhindern. Zur Darstellung der Beziehungen wird wie in Abbildung 3 ein einfaches ERM (Entity-Relationship-Modell) gezeichnet.

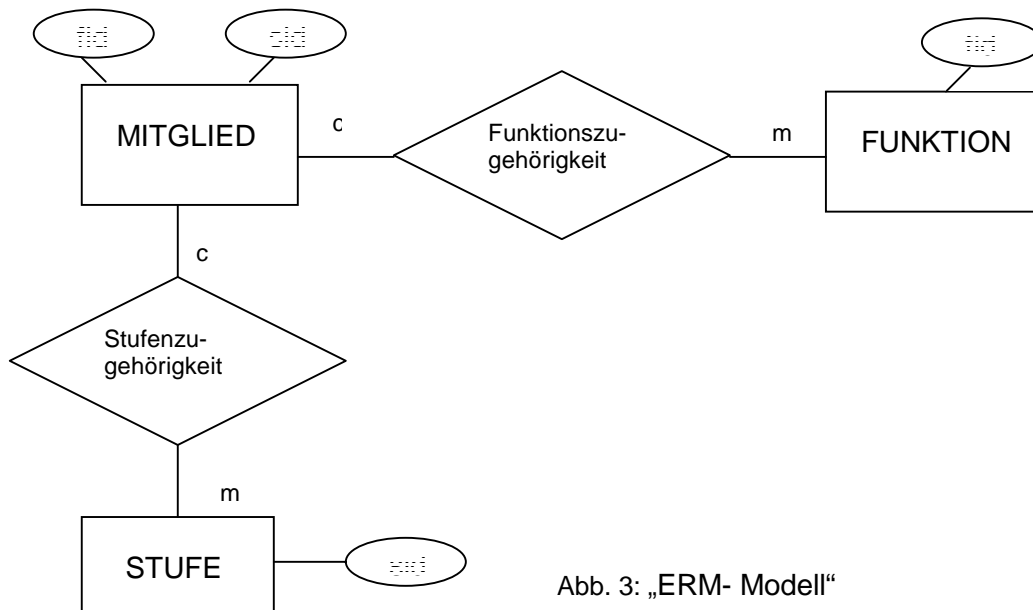


Abb. 3: „ERM- Modell“

Die Beziehungstabellen können weggelassen werden, da es sich um eine (1-m)-Beziehung handelt. Die Fremdschlüssel werden einfach der Entitätsmenge Mitglied angehängt. Aus der Graphik ist ersichtlich, dass der Entitätsmenge MITGLIED die zwei Fremdschlüssel der STUFE (sid) und der FUNKTION (fid) angehängt wurden. Die anderen Merkmale wurden zur besseren Übersichtlichkeit weggelassen. Am Ende enthält die Datenbank nur 3 Beziehungstabellen und keine Beziehungstabelle.¹

Für die Adressverwaltung werden folgende Tabellen benötigt:

MITGLIED: Diese Tabelle enthält alle Angaben über die Mitglieder. Dabei sind alle Aktiven, Komiteemitglieder, Ehemalige und Pfadzeitungs-Abonnenten darin enthalten.

Die Tabelle enthält folgende Merkmale:

Vorname, Name, Vulgo, Adresse, Wohnort, Telefon und Geburtstag

FUNKTION: Einige Mitglieder übernehmen eine spezielle Tätigkeit:

Venner/in: Sie führen eine Gruppe von Jugendlichen

Leiter/in: Sie helfen eine Stufe zu führen

Stufenleiter/in: Sie sind für eine ganze Stufe verantwortlich

Abteilungsleiter: Sie sind für die ganze Abteilung verantwortlich

Die Tabelle hat zwei Merkmale Funktions-ID (fid) und Bezeichnung.

¹ Eine ausführliche Dokumentation über Datenmodellierung ist zu finden unter :

Meier, Andreas. Relationale Datenbanken, Leitfaden für die Praxis. 5. Auflage. Berlin, Heidelberg, New York. 2004. Kapitel 2

STUFE: Aus dieser Tabelle ist ersichtlich welche Stufen und Mitgliedergruppen es in der Abteilung gibt.

1. Stufe: Kinder von 7-11 Jahren mit Leitern
2. Stufe: Jugendliche von 11-16 Jahren mit Leitern
3. Stufe: Jugendliche von 16-18 Jahren mit Leitern
4. Stufe: Jugendliche ab 18 Jahre

Die Tabelle hat zwei Merkmale Stufenid (sid) und Bezeichnung.

5.1.2 Web-Interface

Ohne graphische Oberfläche müsste der Benutzer die SQL-Befehle direkt in die Kommandozeile eingeben. Der Anwender hat in der Regel keine SQL-Kenntnisse und möchte diese nicht über die Tastatur eingeben, lieber möchte er die Abfrage durch einige wenige Mausklicks erledigen. Mit dieser graphischen Oberfläche wird man der Forderung einer benutzerfreundlichen Adressverwaltung gerecht.

Die Oberfläche der Adressverwaltung soll, wie in Abb. 4 aufgebaut sein.

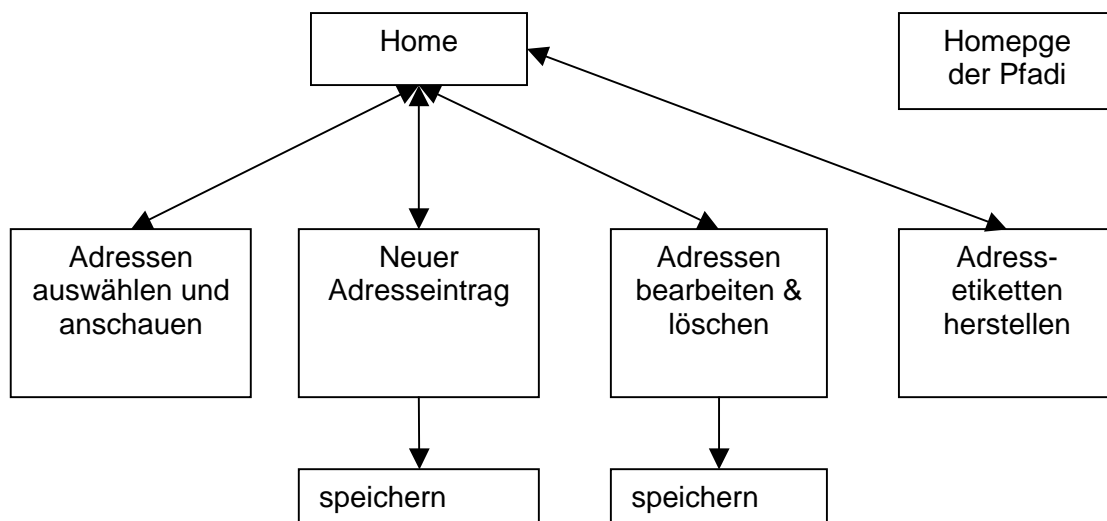


Abb. 4: „Aufbau der Benutzeroberfläche“

Weshalb keine feste Verbindung zur Homepage der Pfadi besteht, wird im Abschnitt 5.4 „Sicherheit und Schutz der Daten“ genauer erläutert.

5.2 Wahl der Datenbank

Zu Beginn standen 3 verschiedene Open Source-Datenbanksysteme zur Auswahl, deren Vor- und Nachteile im Kapitel 4 erläutert sind. Obwohl alle drei Programme vollständige RDBMS sind, fiel die Wahl eindeutig auf MySQL. Denn nur die MySQL-Datenbank ist so einfach in ein Web-Interface einzubauen, was für eine bedienerfreundliche Oberfläche unbedingt nötig ist. Die fehlenden Features bei MySQL stellen für einen Verein kein Hindernis dar, da es sich um eine kleine Datenbank mit nur drei Tabellen und einfachen Verknüpfungen handelt. Für die Entscheidung spielte zuletzt noch ein anderer Faktor eine wichtige Rolle. Der Web-Server der Pfadi unterstützt einzig MySQL-Datenbanken. Da bereits ein Abonnement für die Homepage mit diesem Anbieter läuft, steht dem Verein kostenlos eine MySQL-Datenbank zur Verfügung.

5.3 Implementierung

Nun wird die Datenbank in MySQL erstellt und in ein Web-Interface eingebettet, damit die Adressverwaltung auch für Nicht-Informatiker leicht zu bedienen wird. Die Herausforderung besteht darin, die ausgewählte Software zu installieren und das Konzept praktisch umzusetzen.

5.3.1 MySQL

Die Datenbank soll mit MySQL erstellt werden, welches unter www.mysql.com heruntergeladen werden kann. Um die Verwaltung der Datenbank zu vereinfachen, gibt es ein Administrationstool „MySQL-Front“, das unter www.mysqlfront.de zur Verfügung steht. Leider war die Software nur bis zur Version 2.5 als Free Ware erhältlich. Ab Version 3.0 gibt es die Software nur noch für 30 Tage ohne Einschränkungen gratis zum herunterladen.

Mit MySQL-Front wird es einfach die Datenbank zu verwalten. Die Abbildung 5 zeigt die Arbeitsoberfläche von MySQL-Front. Links sind alle Datenbanken mit ihren Tabellen aufgelistet. Im Fenster rechts kann man die ausgewählte Datenbank bearbeiten, dabei ist es möglich zwischen verschiedenen Ansichten wechseln. Die Änderung kann direkt in der Tabelle eingegeben oder das passende Query in der Eingabezeile ausgeführt werden. Im unteren Teil des Fensters sind alle eingegebenen SQL-Kommandos sichtbar.

Die Kommandos erscheinen, egal ob man die Änderung als SQL-Befehl eingibt oder sie in der Tabelle vornimmt.

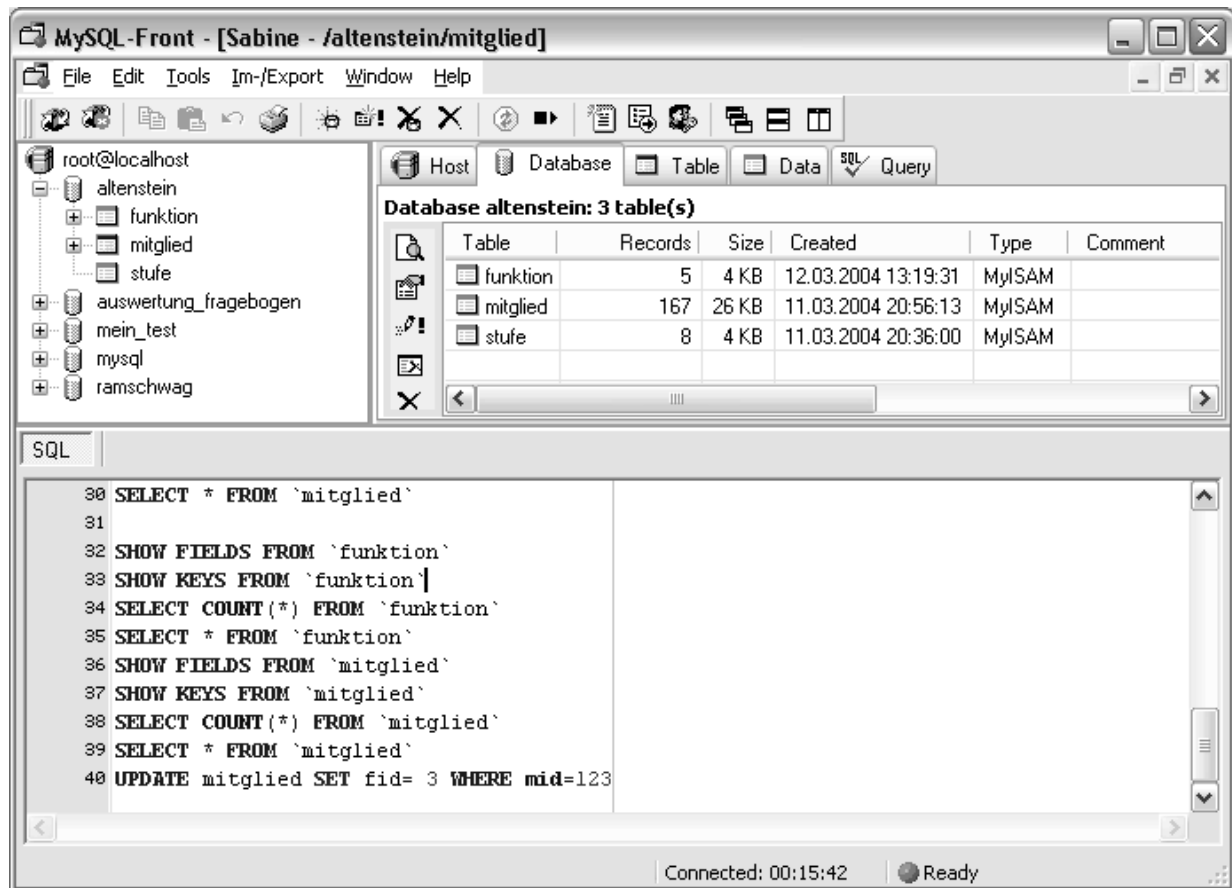


Abb. 5: „MySQL-Front“

5.3.2 PHP und graphische Oberfläche

Um dynamische Websites zu erstellen benutzt man im Zusammenhang mit MySQL meistens die Programmiersprache PHP. Dank PHP kann man direkt von einer Website aus die MySQL-Datenbank kontaktieren und Abfragen übers Web tätigen. Die Programmiersprache PHP kann unter www.php.net heruntergeladen werden. Damit PHP auf einem privaten Computer funktioniert, muss man zusätzlich einen Web-Server installieren. Der am häufigsten benutzte Open Source Server heisst Apache und steht unter www.apache.org zur Verfügung. Diese Softwarekombination ergibt die bekannte LAMP-Konfiguration. LAMP sind die Anfangsbuchstaben von **L**inux – **A**pache – **M**ySQL – **P**HP, wobei das Programmpaket genauso gut mit den Betriebssystemen Windows und OS funktioniert.

Die Oberfläche der Adressverwaltung wird teils mit HTML und teils mit PHP programmiert, da nicht die gesamte Seite, sondern nur die Datenbankabfragen und die zu erstellenden Adresstketten dynamisch sind.

Mit PHP kann man nicht nur eine Datenbank kontaktieren, es gibt noch eine Menge anderer Module zur Erweiterung der Sprache. Die gewünschten Module müssen bei der Installation aktiviert werden, damit nur die benötigten Komponenten installiert werden und kein unnötiger

Speicherplatz belastet wird. Eines dieser Module ermöglicht die Erstellung von pdf-Files, welches im Fall dieses Projekt für die Anfertigung von Adresstiketten geeignet ist.

Am Ende dieser Arbeit ist ein Auszug aus dem Programmcode zur Kontaktierung der Datenbank zu finden.

5.4 Sicherheit und Schutz der Daten

In der Datenbank sind alle Mitgliederadressen und ihre persönlichen Angaben gespeichert. Die Daten sollen auf keinen Fall öffentlich zugänglich sein, um unberechtigten Zugriff und missbräuchliche Veränderung oder Verlust der Daten zu verhindern. Speziell im Fall eines Jugendvereins, wie die Pfadi, sind diese Massnahmen im Sinne des Kinder- und Jugendschutzes. Der Verein will in keinem Fall eine Datenquelle für Kriminelle (Pornographie, Kindesmissbrauch) sein. Ein Interessierter könnte zum Beispiel veröffentlichte Fotos von Aktivitäten mit den Adressen unseres Vereins in Zusammenhang bringen und so das Opfer leicht auffinden.

Die gespeicherten Zugangsdaten der Datenbank im Script der Benutzeroberfläche sind daher ein gewisses Problem. Da man im Internet nur das ausgeführte PHP-Script welches HTML-Code ausgibt sieht, sind die im PHP-Code gespeicherten Zugangsdaten nicht ersichtlich. Ausserdem muss sich der Verein auf die Stabilität des Web-Servers und die Ehrlichkeit der Benutzer verlassen, was ein gewisses Risiko darstellt.

Der Datenschutz (Schutz vor Missbrauch der Daten) und die Datensicherheit (Schutz vor Veränderung und Verlust der Daten) soll durch mehrere Punkte gewährleistet werden.

1. Die Adressverwaltung ist von der öffentlichen Homepage abgegrenzt. Es gibt keinen Link zur Adressverwaltung. Aussenstehende wissen daher nichts von der Internetdatenbank und werden daher nicht zur Manipulation verlockt.
2. Der Zugang ist nur mit einem gültigen Passwort möglich. Die Daten können nur nach einem erfolgreichen Login eingesehen und bearbeitet werden. Nur wenige Mitglieder besitzen das Recht, mit den Daten zu arbeiten.
3. In regelmässigen Abständen wird eine Sicherheitskopie der Datenbestände erstellt. Die Kopie kann auf dem privaten Computer des Adressverantwortlichen gespeichert werden. Eine Sicherheitskopie ist notwendig falls der Web-Server eine Panne haben sollte oder jemand versehentlich etwas Falsches macht und dadurch alle oder einen Teil der Daten verloren gingen.

Beim Befolgen dieser 3 Punkte sollte die Datenbank und ihre Daten genügend geschützt sein. Es ist jedoch nicht möglich einen Hacker-Angriff auszuschliessen, aber weitere Sicherheitsmassnahmen bedeuten einen finanziellen Aufwand.

6 Schlussfolgerung und Zusammenfassung

Aus dem Resultat der Umfrage ist zu lesen, dass die Vereine ein Bedürfnis zur Verbesserung der Administration haben. Das Bedürfnis ist nicht nur auf die Adressverwaltung beschränkt, sondern viele Vereine wünschen sich auch eine Verbesserung im Bereich der Finanzen oder im Sicherheitsmanagement. Im Vordergrund steht jedoch die Bedienerfreundlichkeit der Software und das so auch Zusammenhänge einfacher dargestellt werden können.

Da sich Open-Source Programme besonders für kleine und mittlere Projekte eignen und immer ein grösseres Benutzersegment abdecken, wurde eine geeignete Software gesucht. Open Source Software wird kostenlos verteilt, was vor allem für finanzschwache Vereine einen grossen Vorteil ist.

Für die Umsetzung der Adressverwaltung der Pfadi wurde die Open-Source Datenbank MySQL und PHP verwendet. Mit PHP kann MySQL am einfachsten in eine Web-Interface eingebettet werden. Um eine PostgreSQL- oder Firebird-Datenbank einzubetten braucht es kompliziertere Operationen und es ist schwieriger einen Web-Server mit einer PostgreSQL- bzw. Firebirdkonfiguration zu finden.

Dank der Verwaltungsoberfläche im Internet wird die Adressbearbeitung einfacher und auf eine aktuelle Version reduziert. Der Verein hat einen zusätzlichen Nutzen, da jeder Verantwortliche Zugriff auf die Datenbank hat und gleich selbst die gewünschten Adressen auf Etiketten ausdrucken kann. Um die Datenbank und die Oberfläche zu erstellen, braucht es hingegen einen grösseren Aufwand und mehr Kenntnisse über die Programme.

Ein erstes Ziel, das Finden einer geeigneten Open-Source Software für eine Adressverwaltung, wurde erreicht. Es liegt nun eine geeignete Kombination der Programme und ein Prototyp einer Adressverwaltung mit einer Arbeitsoberfläche vor. Bereits erhielt ich eine Anfrage von zwei weiteren Vereinen, der sich eine ähnliche Datenverwaltung wünscht. Der Prototyp kann weiter entwickelt und angepasst werden. Eine Erweiterung für die Pfadi ist zum Beispiel eine Agenda, in der alle ihre Aktivitäten in einer Tabelle speichern, damit man alles nach dem Datum oder nach Aktivitäten der verschiedenen Stufen sortieren kann. Ausserdem vermietet die Pfadi regelmässig ihr Material (Zelte, Tipi, Kochtöpfe, etc...). In einer Tabelle für die Vermietungen könnte man das ausgeliehene Material verwalten. So kann man sofort bestimmen, ob das Material zur Vermietung frei ist oder ob es der Verein selbst benötigt.

Im Bereich der Vereinsverwaltung liegt weiterhin viel Potential zur Verbesserung. Ein Verein kann mit einem Datenbanksystem die Administration viel einfacher bewältigen, als das mit einer Excel-Tabelle möglich ist.

7 Quellenverzeichnis

Literatur

[Perens 1999]: Perens, Bruce: The Open Source Definition. In: Open sources: voices from the open source revolution. DiBona, Chris / Ockman, Sam
1. Auflage. Beijing/ Cambridge/ Köln. S. 171-188

[Pollakowski 2003]: Pollakowski, Martin: Grundkurs MySQL und PHP: So entwickeln Sie Datenbanken mit Open-Source-Software. 1. Auflage.
Braunschweig/Wiesbaden.

Internet

[Emser 2004]:Emser, Frank: Firebird Datenbank – relationales Open-Source-Datenbanksystem FAQ.
Abrufbar im Internet. URL:
http://www.firebird-datenbank.de/Open-Source_Datenbank_Firebird_FAQ.html
Stand: 14.04.04

[Mojian 2004]: Momjian, Bruce: deutsche Uebersetzung von Barwick Ian (2004). Häufig gestellte Fragen (FAQ) zu PostgreSQL. Abrufbar im Internet. URL:
http://www.postgresql.org/docs/faqs/FAQ_german.html
Stand: 06.04.04

[O'Reilly & Associates 2004]: O'Reilly & Associates: deutsche Übersetzung von Snoopy & Martin Müller (1999). Open Source kurz & gut.
Abrufbar im Internet. URL:
http://www.oreilly.de/german/freebooks/os_tb/os_tb_1.htm#HEADING1-7
Stand: 29.03.04

[The PostgreSQL Global Development Group 2004]: The PostgreSQL Global Development Group: PostgreSQL: Das weltweit fortschrittlichste Open Source Datenbanksystem. Abrufbar im Internet. URL:
<http://advocacy.postgresql.org/advantages/?lang=de>
Stand: 06.04.04

Anhang

Abkürzungen

DBMS	Datenbankmanagementsystem
LAMP	Linux, Apache, MySQL, PHP
PHP	Hypertext Preprocessor
RDBMS	relationales Datenbankmanagementsystem
SQL	Structured Query Language
UDF	User Defiened Functions

Umfrage

Fragen zum Verein

Name des Vereins:

Art des Vereins:

Anzahl Mitglieder:

Fragen zur Datenerfassung

Wie werden die Daten (Adressen) erfasst? schriftlich elektronisch

Nur beantworten, wenn die Daten elektronisch erfasst werden:

Wie werden die Daten erfasst?

- Excel oder anderes Tabellenprogramm
- Microsoft Acces
- Open Source Datenbank
- andere Datenbank

Wie oft gibt es Schwierigkeiten mit der Verwaltung der Daten?

- nie
- manchmal
- oft

Was wird elektronisch erfasst?

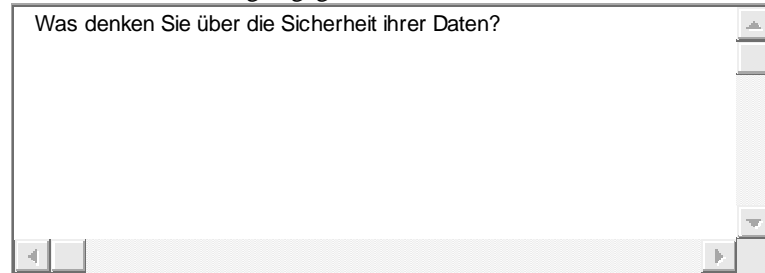
- Adressen
 - Mitgliederfunktionen und andere Angaben über Mitglieder
 - Finanzen und Abrechnungen
 - Sponsoren
 - Aktivitäten
 - Material und Immobilien
 - anderes
-

Fragen zur Datenverwaltung

Wieviel Stunden benötigen Sie für die Verwaltung Ihrer Daten pro Monat?

Haben Sie Befürchtungen gegenüber Sicherheitslücken?

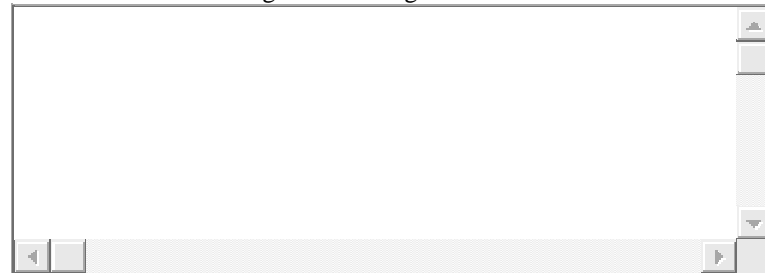
Was denken Sie über die Sicherheit ihrer Daten?

A text input field with a light gray border and a vertical scrollbar on the right side. The text "Was denken Sie über die Sicherheit ihrer Daten?" is at the top. The field is currently empty.

Welche Bedürfnisse können mit Ihrem bestehenden System nicht abgedeckt werden?

- Ändern, Löschen, Einfügen
- Finanzen
- Zusammenhänge einfach darstellen
- benutzerfreundlich gegenüber Nicht-Informatikern
- andere

Hier ist Platz für allfällige Bemerkungen

A large text input field with a light gray border and a vertical scrollbar on the right side. The text "Hier ist Platz für allfällige Bemerkungen" is at the top. The field is currently empty.

Programmcode

1. Programmcode zur Kontaktierung der Datenbank und geordnete Ausgabe

```
<?php

//Datenbank kontaktieren
mysql_connect("localhost","user","password");
function send_sql($sql)
{
    $db="database";
    if (! $res=mysql_db_query($db, $sql))
    {
        echo mysql_error();
        exit;
    }
    return $res;
}

. . .
// Abfrage
$sql= "
    SELECT
        `mitglied`.`vulgo`,    `mitglied`.`vorname`,
        `mitglied`.`name`,    `mitglied`.`adresse`,
        `mitglied`.`plz`,     `mitglied`.`ort`,
        `mitglied`.`telefon`, `mitglied`.`mobile`,
        `mitglied`.`email`,   `mitglied`.`geburtstag`,
        `mitglied`.`sid`,     `mitglied`.`fid`
    FROM
        `mitglied`
    WHERE
        1
    ORDER BY `mitglied`.`name`
";

$result          = send_sql($sql);
$resultate_anz  = mysql_num_fields($result);
$resultate_row   = mysql_num_rows($result);
$field = 0;
$rows = 0;
while ($row = mysql_fetch_array($result,MYSQL_NUM))
{
    foreach ($row as $elem)
    {
        if (empty($elem))
            $elem = "&nbsp;";
        $resultate[$rows][$field] = $elem;
        if ($field == $resultate_anz-1)
        {
            $field=0;
            $rows++;
        }
        else
        {
            $field++;
        }
    }
}

//Ausgabe
echo " <table border = '1' bordercolor = 'black'>";
echo " <tr><td><b>Vulgo</b></td>
    <td><b>Vorname</b></td>
    <td><b>Name</b></td>
    <td><b>Adresse</b></td>
    <td><b>PLZ</b></td>
    <td><b>Ort</b></td>
    <td><b>Telefon</b></td>
```

```
        <td><b>Mobile</b></td>
        <td><b>E-Mail</b></td>
        <td><b>Geburtsdatum</b><br> jjjj-mm-dd</td></tr>";

for($i = 0; $i < $resultate_row; $i++)
{
    echo "<tr><td>" . $resultate[$i][0] . "</td>
        <td>" . $resultate[$i][1] . "</td>
        <td>" . $resultate[$i][2] . "</td>
        <td>" . $resultate[$i][3] . "</td>
        <td>" . $resultate[$i][4] . "</td>
        <td>" . $resultate[$i][5] . "</td>
        <td>" . $resultate[$i][6] . "</td>
        <td>" . $resultate[$i][7] . "</td>
        <td>" . $resultate[$i][8] . "</td>
        <td>" . $resultate[$i][9] . "</td></tr>";
}
echo "</table>";
?>
```

2. pdf-Adressetiketten erstellen

```
<?php

// SQL-Abfrage
. . .
//Ausgabe pdf
for ($k=0; $k < $resultate_row; $k=$k + 10){

// Beginn einer Seite
    pdf_begin_page($pdf, 595.3, 842);
    pdf_set_font($pdf, "Helvetica", 12, "host");

    // 1. Spalte
    $h=0;
    for($i = $k; $i < (10+$k) and $i < $resultate_row; $i++)
    {
        $h++;
        $pos=($h * (-80)) + 800;
        pdf_show_boxed ($pdf,
            "" . $resultate[$i][0] . "
            " . $resultate[$i][1] . " " .
            $resultate[$i][2] . "\n
            " . $resultate[$i][3] . "\n" .
            $resultate[$i][4] . "
            " . $resultate[$i][5] . "
            ", 50, $pos, 180, 70, "left");
    }
    // 2. Spalte
    . . .

// Ende einer Seite
    pdf_end_page($pdf);
}

// Ende des Dokuments
pdf_close($pdf);

// Ausgabe des Pdf
$data = pdf_get_buffer($pdf);

// Angaben, damit der Browser die Datei als pdf interpretiert
header("Content-type: application/pdf");
header("Content-Disposition: inline; filename=downloaded.pdf");
header("Content-length: " . strlen($data));

echo $data;
?>
```